Automatic Synchronization of Music Data

Vlora Arifi, Michael Clausen, Frank Kurth, and Meinard Müller
Institut für Informatik, Abt. III
Universität Bonn
Römerstr. 164
D-53117 Bonn, Germany
{*arifi, clausen, kurth, meinard*}*@cs.uni-bonn.de*

Abstract

Digital music libraries typically contain the same piece of music in different data formats such as the score, some MIDI-files, and several interpretations by various musicians in form of CD recordings. Inhomogeneity and complexity of such music data make content-based browsing and retrieval in digital music libraries a difficult task with many yet unsolved problems. One important step towards a solution are synchronization algorithms which automatically link data streams of different data formats representing a similar kind of information. In this article we introduce some algorithms which automatically link any two data streams given in score-, MIDI- or CD-format representing the same polyphonic piano piece. This involves some steps for extracting note parameters such as onset times and pitches from CD-data. These discrete parameters makes the physical audio data comparable to the purely symbolic score data. In this context, we also give a summary and specification of further problems in computational musicology related to the discussed synchronization problems.

1 Introduction

Modern digital libraries contain many different kinds of information such as textual, visual, and audio data. Among this multi-media based information, music data constitutes a particularly difficult case: music information is represented in various data formats which, depending on the application, fundamentally differ in their respective structure. For example, the *score* — encoded in a formal language and depicted in a graphical-textual form — gives a specification of what we commonly refer to the "piece of music". In a score the notes are fixed by parameters such as pitch, relative onset times, and note durations. The global tempo and local tempo variations are specified by textual notions such as allegro or moderato and accelerando or ritardando. Similarly, loudness and dynamics are described by terms such as piano, forte, crescendo, or diminuendo. Hence the score is just a description of the piece of music which leaves a lot of room for various interpretations not only concerning the tempo and dynamics but also concerning the notes itself — one just may think of vague notations such as trills, arpeggios, or grace notes. Fig. 1 shows some score representation of the beginning of the *Aria con Variazioni* by J. S. Bach, BWV 988. These $4\frac{1}{3}$ measures will serve as running example throughout this article and will simply denoted as Aria.

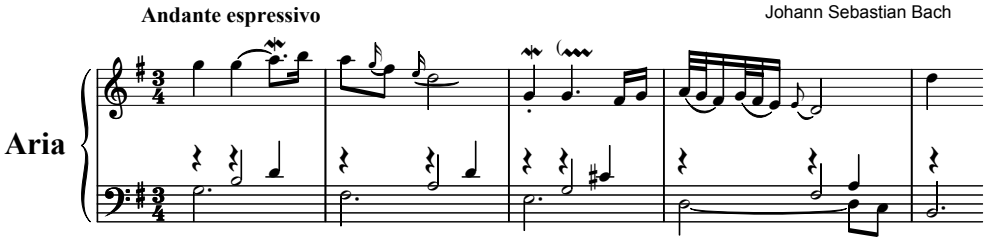


Figure 1: Beginning of the *Aria con Variazioni* by J. S. Bach, BWV 988.

From a physical point of view, the musician, by means of his voice or instrument, generates an *audio signal*. This audio signal has the form of a sound wave emerging at its source and spreading through the air. Graphically, such an audio signal may be represented by its *waveform* which depicts the amplitude of the air pressure over the time (see Fig. 2 for an example). The *PCM-format* (Pulse Code Modulation), as used for CD-recordings, is a discretized and encoded version of such a waveform.

In the following, we do not distinguish between the waveform and its PCM-representation.

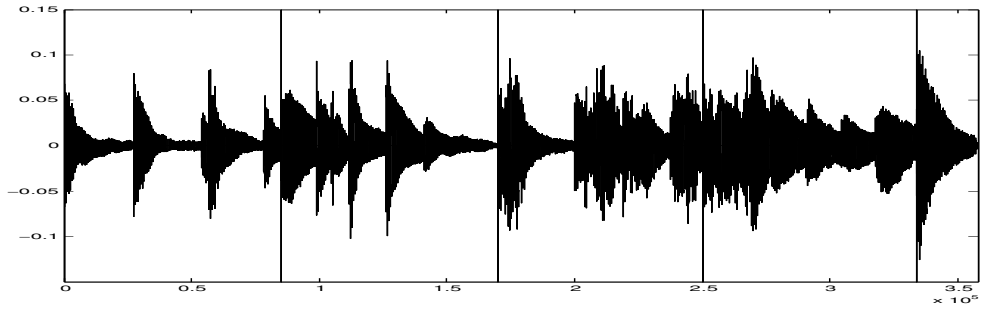


Figure 2: Waveform representation of an Aria interpretation.

As opposed to the score representation, the waveform encodes all information needed to reproduce the acoustic realization of the specific interpretation of some piece of music with all its temporal, dynamical, and tonal micro deviations making music alive. However, in the waveform note parameters such as onset times, pitches, or note durations are not given explicitly. Even worse, considering that even a single note of the score becomes a complex sound when played on some instrument — not only does it consist of several harmonics but also of noise components and vibrations — one might guess how hopeless it is to determine note parameters from the waveform of some polyphonic orchestral piece.

The *MIDI-format* (see, e.g., Selfridge-Field (1997)) may be thought of as a hybrid of the score- and the waveform-based PCM-format: it can encode all relevant content-based information on the notes of the score as well as agogic and dynamic niceties of some specific interpretation. However, MIDI is quite limited especially in modeling the timbre of a sound. MIDI data streams are very often visualized by the piano roll representation as shown in Fig. 3.

From the above discussion it should be clear that what we simply refer to as a "piece of music" is something rather complex due to different formats as well as various realizations. For example, a digital music library may contain, for one and the same piece of music, the score given in Capella or Score format, some MIDI-files, and several interpretations by various musicians in form of CD recordings which may differ considerably, e.g., in tempo or dynamics. Such inhomogeneity and complexity

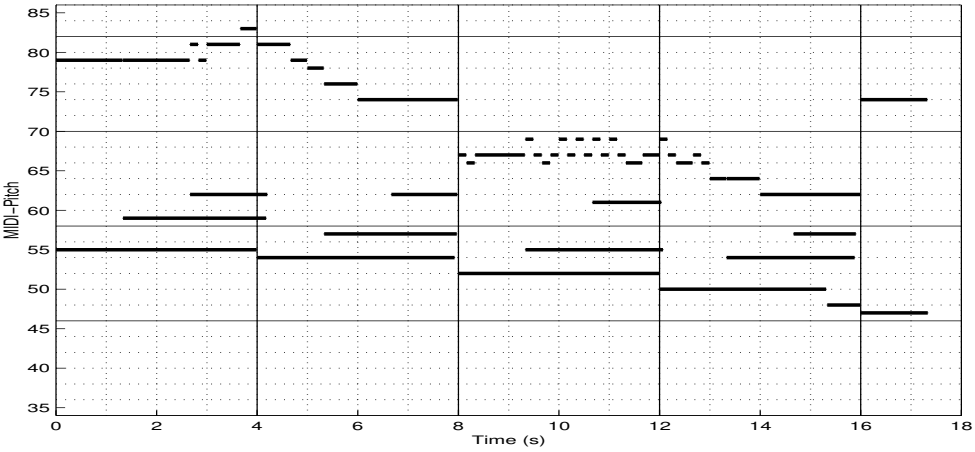


Figure 3: Piano-roll representation of an Aria interpretation.

make content-based browsing and retrieval in digital music libraries a difficult task with many yet unsolved problems (see Section 2).

One important step towards a solution are synchronization algorithms which automatically link data streams of different data formats representing a similar kind of information. In particular, in the framework of audio by *synchronization* we mean some procedure which, for a given position in some representation of a given piece of music (e.g., given in score format), determines the corresponding position within some other representation (e.g., given in PCM-format). Such synchronization algorithms have applications in many different scenarios: following some score-based music retrieval, linking structures can be used to access some suitable audio CD accurately to listen to the desired part of the interpretation. A further application is the automatic annotation of a piece of music in different data formats as a basis for content-based retrieval. As another example, musicologists can use synchronizations for the investigation of agogic and tempo studies. Furthermore, temporal linking of score and audio data can be useful for automatic tracking of the score position during a performance.

In this article we concentrate on the three representative data formats mentioned above: the purely symbolic *score format*, the physically based *PCM-format*, and the semi-symbolic *MIDI-format*. Depending on the formats of the two data streams to be synchronized we speak of Score-

to-MIDI (SM) synchronization, Score-to-PCM (SP) synchronization, or MIDI-to-PCM (MP) synchronization, etc. We refer to Fig. 4 for an illustration of the various synchronization problems.

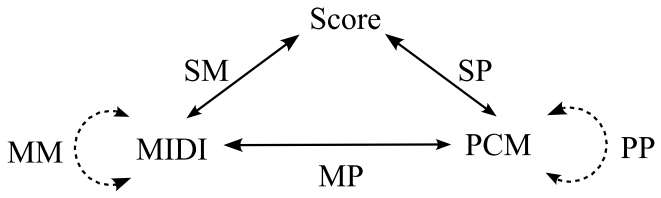


Figure 4: Overview on various synchronization problems.

Especially, SP- and MP-synchronization constitute a difficult problem since the waveform-based PCM-format does not contain any explicit information on the notes. Here, in a preprocessing step, one first has to extract information such as note onsets and pitches from the PCM-recording in order to make it comparable to other symbolic score-like representations and hence algorithmically usable for the actual synchronization. However, as will be summarized in Section 2, the extraction of note information from the waveform of polyphonic music constitutes an extremely difficult problem which is solved only for a few special cases. Especially, in the most general case of orchestral music the extraction problem and not to mention the transcription problem are largely open problems which seem to be unfeasible. In our research, we have concentrated on polyphonic piano music. In contrast to many other research projects we do not restrict ourselves to PCM-data generated by MIDI-pianos. Instead we allow PCM-data generated by any acoustic piano, e.g., audio data from a CD containing piano music. This in general extremely complex data leads to many erroneous extraction results which would not be acceptable when treated as a transcription of the original piece of music into some score-like format. However, the extracted data — even from very complex piano pieces — is of sufficient quality to be used in the proposed synchronization algorithms.

The rest of this article is organized as follows. In Section 2, we review several problems in computational musicology which are intimately

related to our synchronization problem and give links to some recent literature. One goal is to give a clear specification of these problems which are not very well marked-off in the existing literature. In Section 3, we summarize our system for the extraction of musically relevant parameters from PCM-data streams. The actual synchronization algorithms are described in Section 4. One additional goal is to demonstrate how problems in computational musicology may be modeled by using a solid mathematical language. We first give a mathematical definition of fundamental notions such musical and physical onset times, tempo functions, and time flow and then give a rigorous specification of our synchronization problems. In Subsection 4.1, we describe a suitable data format for the score data stream introducing the concept of fuzzy-notes which handles ambiguities such as trills or arpeggios in the score. Crucial for the synchronization algorithms, being based on dynamic programming, is our carefully designed cost function which will be described in Subsection 4.2. Since this subsection is of rather technical nature, we hope that the example in Subsection 4.3 will help the mathematically non-experienced reader to understand the main underlying ideas. Finally, Section 5 contains a summary of our experiments, and Section 6 gives some concluding remarks and an outlook to future work,

2 Synchronization and Related Problems

To define and classify the synchronization problem it is helpful to delimit this problem to related or complementary problems in computational musicology such as segmentation, extraction, and music transcription. In this section we give a short summary of these problems and give links to some of the relevant literature.

In audio signal processing the notion of *segmentation* is not very well specified and is used in many different settings. Generally, by segmentation one means the temporal partition of some acoustic signal into logically coherent sections. Segmentation of some piece of music may be based on purely content-based criteria, such as the partition of a sonata into its movements, or the partition of the first movement into exposition, development, recapitulation, and coda. Also the partition of a piece of music into measures may be considered as a segmentation. In

audio signal processing, however, the segmentation problems deal with the decomposition into more elementary units. For example, in case of a speech signal the segments are often chosen to be the phonemes, whereas in case of monophonic music they may be chosen to be the notes. In the latter case, the segmentation boundaries consist of the initial and end points of the corresponding notes (see, e.g., Raphael (1999)). In this article, we typically deal with audio signals of polyphonic music given in PCM-format. Even in the case of a single instrument — in the following, we concentrate on the piano — it is not any longer clear how to define the segmentation boundaries in the polyphonic scenario in a meaningful way. In this case the corresponding audio signals are complex sound mixtures of just played and abating notes where, in addition, each realization of a note itself constitutes a complex sound. In combination with resonance effects, this intermingling and superposition of sounds results in surprising phenomenons, e.g., a sudden increase of the intensity of certain harmonics even though no new notes are played (see, e.g., Blackham (1998)). Due to such phenomenons one needs a more general notion of segmentation. Segmentation boundaries are defined to be points of time where sudden changes in the spectrum as well as in the energy of the signal occur. Then within a segment the signal may be assumed to be quasi-periodic and the segmentation boundaries are candidates for the onset times of notes.

The segmentation problem may be considered as part of the more general *extraction problem*. In the scenario of audio signal processing one tries to extract a suitable set of parameters from some waveform-based representation of an audio signal which somehow describes the musical content of the underlying piece of music. Among these parameters are not only the above mentioned candidates for onsets but also spectral parameters for the description of pitch, parameters for the description of note duration or loudness, and so on. In this context, *beat-* and *tempotracking* can be considered as special cases of the extraction problem where temporal and dynamical parameters are suitably interpreted (see, e.g., Cemgil et. al. (2000) or Goto (2001)), or repeating spectral patterns are used (see Foote et. al. (2001)). For the task of *automatic accompaniment* spectral parameters are used to determine the pitch (see, e.g., Dannenberg et. al. (1988) or Raphael (1999)), sometimes in combination with energy parameters (see Cano et. al. (1999)). At this point we

mention that in case of complex polyphonic music the extraction of the pitch from the corresponding spectral parameters constitutes a more or less unsolved problem. One reason is that in some mixture of sounds the harmonics cannot be uniquely assigned to the single notes. Possible approaches towards a solution of the pitch extraction problem are based on the usage of note templates (see, e.g., Bobrek et. al. (1998) or Ortiz-Berenguer et. al. (2002)) or the usage of additional score information (see Scheirer (1995)). In our approach, described in Section 3, we use a similar technique as described in Bobrek et. al. (1998).

By *music transcription* one roughly means the transcription of some music recording into score notation. In other words, from the audio signal the score data such as notes, instruments, measure, and annotations for tempo and dynamics are to be determined. Hence, transcription and extraction seem to be similar problems which, however, differ in one essential point. The score data are "standardized" parameters which allow room for various interpretations such as local tempo variations and timbre manipulations, whereas the extracted note parameters of some music signal include the score information as well as all specific features coming from the particular interpretation of the underlying recording. Hence, for the *transcription problem* the following approach seems to be feasible: in a first step one extracts note parameters such as onset times, pitch, and note duration; in a second step, this data must be "reassessed" by suitable quantization and normalization methods. As mentioned above the extraction of note information from the waveform, not to mention the transcription problem, is a largely open problem which seems to be unfeasible in the most general case of orchestral music. Special cases of music transcription are discussed, e.g., in Cemgil et. al. (2000).

Finally, *synchronization* may be generally defined as the temporal coordination of two independent processes when coming into interaction. For example, in multimedia-based applications several data streams of various formats such as video, audio, and text have to be synchronized for simultaneous reproduction. In the scenario of this article the *synchronization problem* refers to temporal synchronization of two different realizations or variations of the same piece of music which are given as two independent data streams in possibly different data formats. The synchronization is realized by some suitable linking structure. As an example, one data stream could be some CD-recording of a piece of mu-

sic, whereas the other one could be the score data represented in some suitable digital format. In this case, the *synchronization problem* can be considered to be *complementary* to the *music transcription*. While in music transcription there is only *one* data stream from which the score parameters have to be determined, in the synchronization problem one starts with *two* data streams. Furthermore, the goal of music transcription is — in order to get "pure" score data — to "iron out" the deviations in the extracted data resulting from the specific underlying interpretation. On the other hand, the goal of synchronization (particularly of SP-synchronization) is to pick up just these local time deviations in the underlying interpretation of the the PCM-data stream to accomplish the linking with the uninterpreted score data stream.

There are various problems intimately related to the synchronization problem. In the following we quickly summarize some of the recent approaches which are partially based on similar techniques as discussed in this article. However, the discussed problems are somewhat different to our scenario. In the problem of *automatic accompaniment* one typically has a solo part played by some musician which is to be accompanied by a computer system in real time. Dannenberg et. al. (1984, 1988, 1994) describe one of the first algorithms for automatic music accompaniment reducing the synchronization problem to an LCS (longest common subsequence) problem which is solved using dynamic programming. Vercoe (1984) has developed a system for automatic accompaniment of a transverse flute. Raphael (1999, 2001) has developed a system for automatic musical accompaniment of an oboe based on Hidden Markov Models. Desain et. al. (1997) describe some general sequential and tree-based score-performance matching algorithms. A similar problem addresses Large (1993) using dynamic programming to study music production errors. In all of those approaches the data streams involved in the synchronization problem either explicitly contain score-like note parameters or only consist of monophonic music so that the note parameters are comparatively clean and error free. In our scenario, however, we also allow PCM-data of complex polyphonic piano music where there are no such explicit and clean parameters.

Finally, we want to mention the comprehensive book by Mazzola (2002) who gives, among many related topics, a detailed account on local tempo variations resulting from expressiveness in performances.

3 Extraction of Note Parameters

In this section we describe our system for extracting note parameters from the PCM-data stream. The synchronization algorithms which are discussed in Section 4 only use the onset times and pitches of the note objects. One could also think of exploiting note durations or parameters on the dynamics. However, the restriction to onset times and pitches is sufficient for obtaining good synchronization results. This is also in accordance with our experience that a staccato version of a piece of music contains most of the characteristic information needed for an identification of that piece.

Our extraction algorithms in the most parts use established techniques from audio signal processing, which will not be discussed in detail. Our main contributions outlined below are a refined template matching algorithm for polyphonic pitch extraction and a two-step algorithm for note onset detection.

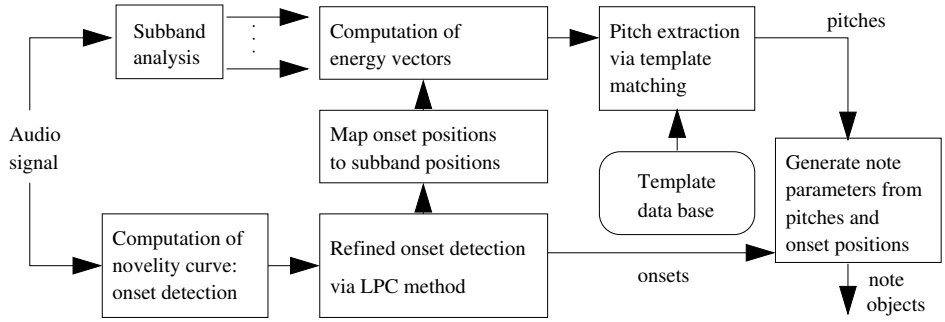


Figure 5: Diagram of the feature extraction algorithm.

The extraction of the note parameters is performed in two stages. First, the onset times are estimated using suitable attack detection algorithms. Ideally, this yields a sequence of time intervals or segments, each containing a note event of the underlying piece of music. In this, a *note event* denotes a single note, a superposition of several simultaneously sounding notes (with possibly different onset times), or chords. In a second step, we examine each of those note events and try to determine the pitches of all musical notes contained in the event.

Fig. 5 gives an overview on the components of our feature extraction algorithm which will now briefly discussed. Several methods have

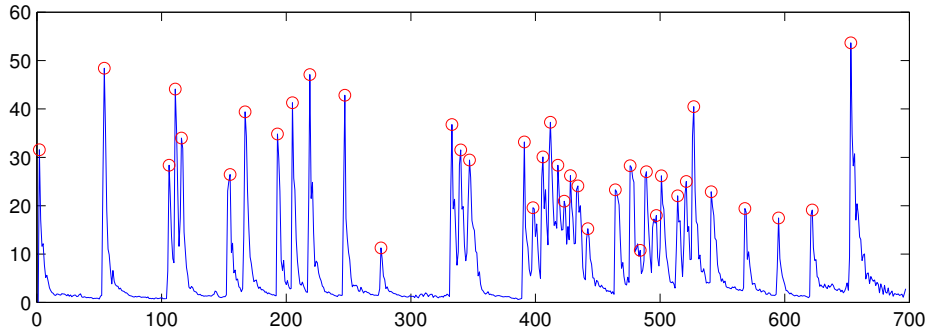


Figure 6: Novelty curve extracted from some interpretation of the Aria in Fig. 1.

been proposed for detecting attacks or onsets of musical notes. Those may generally be classified into frequency- and time-domain approaches. While investigating frequency-domain approaches such as linear prediction, where changes in the frequency-contents of successive time-frames are tracked, we found that several notes with relatively low energy (mainly repetitions of notes and chords) could not be detected. Similary, pure time-domain approaches, such as creating coarse signal-approximations using sliding windows, resulted in inferior detection results with passages containing strong variations in dynamics.

Hence, we employed a compromise between time- and frequency-domain approaches for attack detections. For this, we used novelity curves (similarly to Foote (2000)), which roughly describe an approximation of the change of a signal's frequency contents in time by summing over the absolute changes of the last two steps' magnitude spectra. Fig. 6 shows the novelity curve extracted from some interpretation of the Aria from Fig. 1.

As the time-resolution resulting from this procedure is relatively coarse, a postprocessing is performed based on the concept of linear prediction. Using a method proposed by Foster et al. (1982), this step refines the estimated coarse onset positions to a reasonable time resolution of up to 10 ms. In Fig. 6, the peaks corresponding to the refined onset times are marked by circles. Fig. 7 shows the detected onset times in the context of the physical waveform corresponding to the particular interpretation of the Aria.

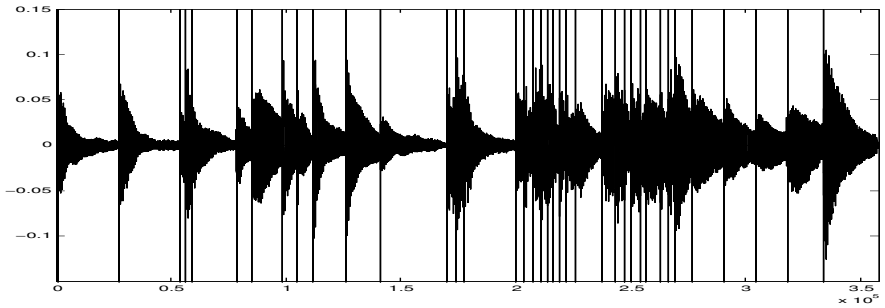


Figure 7: Detected onset times in the context of the physical waveform corresponding to the particular interpretation of the Aria.

As a second stage, polyphonic pitch detection is perfomed using a template matching algorithm. This algorithm is subsequently applied to each time segments between two successive detected attack positions. The idea of template matching is to successively compare the target interval (in a suitable representation) with a collection of pre-trained templates (the template data base). In our case, the template data base consists of one template for each possible musical note (intuitively it corresponds to the keyboard of the piano). Then, the time interval is assigned those notes whose templates yield a "good" match w.r.t. some suitable criterion.

In our setting, the templates are created using fingerprints of the frequency-energy distributions of the single notes. Technically, we use a tree structured filterbank as proposed by Bobrek et al. (1998) to transform the signal into a subband representation. The tree structure of the filterbank — and hence the frequency ranges of the subbands — are chosen such that the fundamental frequency of at most one piano note (well-tempered tuning) falls into one subband. This guarantees that in the template matching algorithm templates can be uniquely assigned to energy vectors. For further details on the filter bank tree structure we refer to Arifi (2002) and Bobrek et al. (1998).

Our templates were recorded using a Yamaha GranTouch E-Piano. We also evaluated templates generated by two acoustic pianos (Steinway and Schimmel). However, the E-Piano's templates turned out to be the most robust for our purposes.

Our template matching algorithm works as follows: starting with an

initial energy vector of the signal on the time segment in question, the algorithm roughly tries to select a template from the TDB which optimally fits the energy vector w.r.t. a certain criterion (see below). If successful, a corresponding energy fraction is subtracted from the energy vector to yield a modified vector, which is used to recurse the procedure until the remaining residual energy falls under some lower bound. We used the following criterion for selecting a template which optimally fits an energy vector $E \in \mathbb{R}^M$: find the lowest subband index k such that $E(k) \geq (c/M) \sum_{i=1}^{M} |E(i)|$ (for some suitable prior constant c). If such a k exists, search the TDB for a template S_k with fundamental frequency in this subband. If E contains S_k to a significant extent, select S_k as a matching template. Bobrek et al. (1998), instead of using the lowest significant subband, choose the subband containing the highest energy component for selecting the template. However, in our experiments this criterion turned out to yield several octave interval errors in pitch detection, especially when applied to recordings from acoustic pianos.

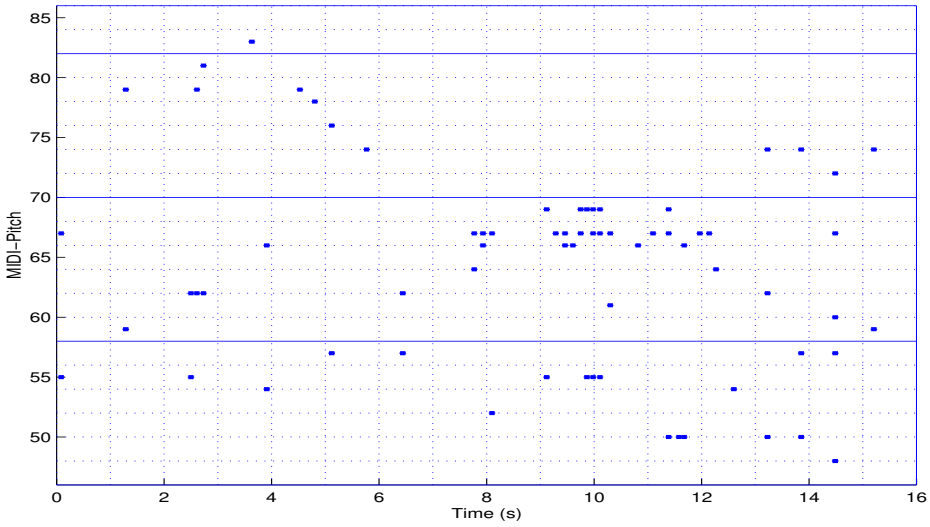


Figure 8: Piano roll representation on the note parameters extracted form the interpretation of the Aria.

Fig. 8 shows the piano roll representation of the note parameters extracted from the interpretation of the Aria using the proposed algorithms. The explicit parameters of the first two measures can also be found in

Table 2. As explained before, note durations are not included in our extraction process.

To conclude this section we note that our algorithms require a careful choice of parameters (e.g., thresholds for template extraction, peak picking, or the minimum inter-onset interval). For a detailed discussion we refer to Arifi (2002).

4 Synchronization Algorithm

Before we describe the actual synchronization algorithms in this section we first have to further specify the synchronization problems depicted in Fig. 4. To this means we dwell on notions of tempo, time flow, and synchronization following in part Mazzola (2002).

Some annotation such as "M.M. quarter note = 100" at the beginning of some score gives the instruction to play the piece of music in uniform tempo of 100 quarter notes per minute. Instead of "quarter-note" one also could choose a different base unit such as "quaver" or "half-note". In general we specify the *uniform tempo* T by the formula

$$T = \frac{\#\text{beats}}{\text{minute}} \ .$$

Independent of such a tempo annotation one can allocate each metric position within any measure of some score a *musical onset time*: Suppose there are b beats per measure and the position in question is the kth beat in the mth measure, then this position has *musical onset time*

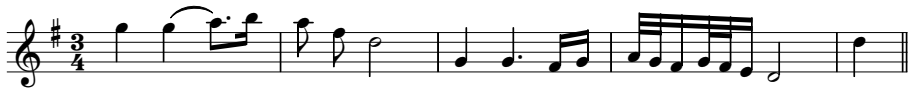
$$E = b \cdot (m-1) + k.$$

As an example, Fig. 9 shows the onset times of the first few note objects in the right hand of the Aria where one has $b = 3$ beats per measure.

If E_0 and E are musical onset times with $E \geq E_0$ then, presupposing a unifrom tempo T, the *physical onset time* $e(E)$ corresponding to E relative to the physical onset time corresponding to E_0 is given by

$$e(E) = e(E_0) + \frac{E - E_0}{T}. \tag{1}$$

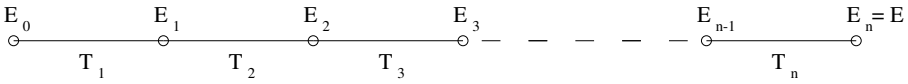
In reality one seldom has a uniform tempo. Slight local tempo deviations as well as tempo changes prescribed by the score have to be considered.



ith note	1	2	3	4	5	6	7	8	9	10	11	12
E_i	0	1	2	$2\frac{3}{4}$	3	$3\frac{1}{2}$	4	6	7	$8\frac{1}{2}$	$8\frac{3}{4}$	9

Figure 9: Onset times of the first 12 note objects in the right hand of the Aria.

Closer to reality is in this case a piecewise uniform tempo. Suppose $E_0 < E_1 < \ldots < E_n = E$ are the musical onset times, so that the tempo between E_{i-1} and E_i is uniformly given by T_i:



Then the physical onset times of E are given by the formula

$$e(E) = e(E_0) + \sum_{i=1}^{n} \frac{E_i - E_{i-1}}{T_i}.$$

Considering a finer and finer subdivision, the sum $\sum_{i=1}^{n}(E_i - E_{i-1})/T_i$ may be regarded as a Riemann-sum and T_i may be interpreted as instantaneous tempo. Under suitable assumption the *time flow* at musical onset time $E \geq E_0$ based on the tempo function T is given by

$$e(E) = e(E_0) + \int_{E_0}^{E} \frac{dx}{T(x)}.$$

Hence, given time flow e, the *tempo* in E is the reciprocal value

$$T(E) = \left(\frac{de}{dE}(E)\right)^{-1}.$$

We illustrate the introduced notions by means of two typical examples. In case $T(x) = T$ is constant, one has the case of a uniform tempo and $e(E)$ reduces to the above formula $e(E) = e(E_0) + (E - E_0)/T$. In case

$T(x) = \alpha \cdot (x - E_0) + T_0$ (for some $\alpha \neq 0$, a constant $T_0 > 0$ and all $x \in [E_0, E]$), one gets

$$e(E) = e(E_0) + \frac{1}{\alpha} \ln \frac{\alpha(E - E_0) + T_0}{T_0}.$$

In other words, for $\alpha > 0$ one has, starting with the musical onset time E_0, a uniform *accelerando*, i.e., a tempo acceleration, whereas for $\alpha < 0$ one has a uniform *ritardando*.

Now, suppose two given tempo functions T_1 und T_2 of two different interpretations of the same piece of music are known leading to the time flows

$$e_1(E) = e_1(E_0) + \int_{E_0}^{E} \frac{dx}{T_1(x)} \quad \text{und} \quad e_2(E) = e_2(E_0) + \int_{E_0}^{E} \frac{dx}{T_2(x)}.$$

Then the (ideal) *synchronization* of the two interpretations consists in linking the point of time $e_1(E)$ with the corresponding point $e_2(E)$ of the second interpretation for each musical onset time E of the score. This is illustrated by Fig. 10.

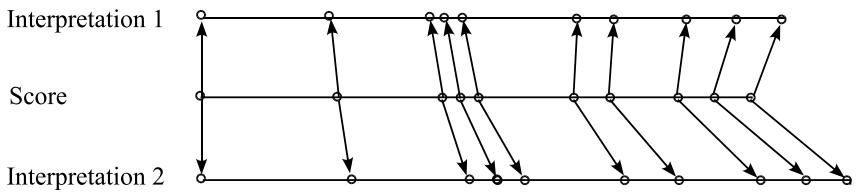


Figure 10: Ideal synchronization of two interpretations with time flows $e_1(E)$ and $e_2(E)$ via the score.

The reality, however, is far from this idealized scenario. First, in the score one often has only vague annotations for the tempo such as "Allegro" or "Largo". Second, (positive or negative) tempo accelerations are just adumbrated by notions such as "ritardando" or "accelerando". The exact course of the tempo is not specified and varies greatly from the respective interpretation. Hence, in general there is no canonical tempo function, which could be assigned to the score. Each tempo function is based on the concrete interpretation of the score.

We are now able to specify the various synchronization problems in more detail (see Fig. 4). The problems are listed by increasing degree of difficulty.

1. In the easiest scenario the score and some MIDI-data stream are given. Here the SM-synchronization problem boils down to computing the tempo function of the MIDI-recording. This task is comparatively easy since the MIDI-file contains the note parameters more or less explicitly. Nevertheless, there are still some problems caused by the vague annotations of the score such as trills, arpeggios or grace notes. To cope with this problem we will introduce the concept of *fuzzy-notes*.

2. The score and a CD-recording in PCM-data format are given. Then the SP-synchronization problem consists of computing the tempo function of the PCM-data stream. This task is much more difficult than the previous one since the extraction of the note parameters from the PCM-file, as shown in Section 3, causes great trouble leading to erroneous extraction results. We will cope with this problem using cost-optimal partial matches based on some suitably designed cost function which takes possible erroneous extraction parameters into account.

3. The score, a MIDI- and PCM-data stream are given. The MIDI- and PCM-interpretations are to be synchronized. In this case, the MP-synchronization can be done by combining the solutions of the two previous tasks.

4. Only the MIDI- and the PCM-interpretation are given. Score information does not exists. Then instead of using the missing score as "pure" reference one takes the MIDI-version instead. Here, the procedure is similar to the second case; the main difference is the lack of the musical onset times.

Furthermore, MM-synchronization and PP-synchronization, i.e., the synchronization of two MIDI-versions respectively two PCM-versions of the same piece of music may be realized by combing solutions of above problems via

$$\begin{array}{lcll} \mathrm{SM}_1 & \& & \mathrm{SM}_2 & \longrightarrow \quad \mathrm{M}_1\mathrm{M}_2 \\ \mathrm{SP}_1 & \& & \mathrm{SP}_2 & \longrightarrow \quad \mathrm{P}_1\mathrm{P}_2. \end{array}$$

Due to space limitation, we consider in the rest of this sectino only the case of a score- and a PCM-data stream (SP-synchronization). The other cases such as SM- or MP-synchronization are even easier or can be done in a similar fashion (see Arifi (2002)). Fig. 11 shows the steps performed in our algorithm. The extraction step has already been described in Section 3. Preprocessing of the score will be content of Subsection 4.1 whereas the other steps are the content of Subsection 4.2. A further goal of these subsections is to show how such concepts may be mathematically modeled using the concept of sets and partial maps. We refer the reader to Subsection 4.3 where the abstract, mathematical notions are illustrated by means of our running example.

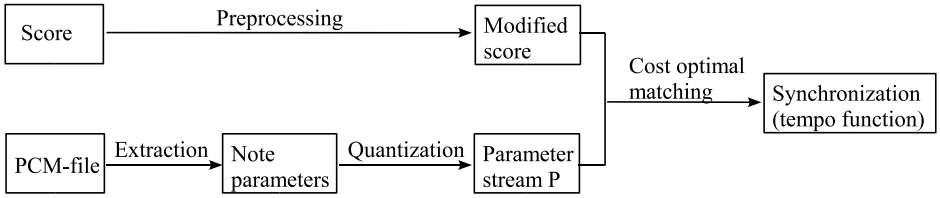


Figure 11: Synchronization algorithm for a score and PCM-data stream.

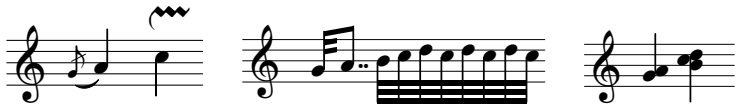
4.1 Preprocessing of Score Data

First, we discuss how to preprocess the score data which is assumed to exist in electronic form (e.g., as a file in the Capella format). In view of the synchronization algorithm we only use the musical onset time and pitch; other parameters such as duration or dynamical annotations are not used. As defined before the *musical onset time* of some the kth beat in the mth measure is given by $E = b \cdot (m-1) + k$, where b denotes the number of beats per measure. Hence $E \in \mathbb{Q}$, where $\mathbb{Q}$ denotes the set of rational numbers. Furthermore, we identify a pitch p with the corresponding MIDI pitch given by an integer between 0 and 127. Defining $[0:127] := \{0, 1, \ldots, 127\}$ one writes $p \in [0:127]$. For example, the standard pitch a of frequency 440 Hz is represented by the number $p = 69$.

We distinguish between two kinds of note objects: *explicit* and *implicit* ones. We first consider *explicit* objects where all note parameters are

given explicitly. Then *all* explicit note objects at a given onset time E will be represented by a pair (E, H_0) which consists of the onset time E and a set of pitches $H_0 \subseteq [0:127]$ containing precisely all the pitches of the corresponding notes. For example, the explicit notes at onset times $E = 0$ and $E = 1$ of the Aria in Fig. 1 are represented by $(0, \{55, 79\})$ and $(1, \{59, 79\})$, respectively.

By an *implicit* note object we understand notes or a group of notes with some additional specification such as a trill, an arpeggio or grace notes. Implicit objects allow different realizations, depending on the epoch and the actual interpretation. To get this ambiguity under control we introduce the concept of a fuzzy note. A *fuzzy note* is defined to be a tuple (E, H_1) consisting of some musical onset time $E \in \mathbb{Q}$ and some set of *alternative* pitches $H_1 \subseteq [0:127]$. Then an implicit note object, such as a trill, is represented by the musical onset time of a certain main note and the set of all pitches appearing in a possible realization of this object. Fig. 12 illustrates the definition. Here, the two fuzzy notes are given by $(0, \{67, 69\})$ and $(1, \{71, 72, 74\})$.



(a) (b) (c)

Figure 12: Appoggiatura and trill, (a) notation, (b) possible realization, (c) fuzzy note.

To simplify the further discussion we assume that there is at most one implicit note object at a given musical onset time E. It is a straightforward generalization to also admit several implicit note objects a time. Then after preprocessing we may assume that a score is given by some subset $S \subset \mathbb{Q} \times 2^{[0:127]} \times 2^{[0:127]}$, where $2^{[0:127]}$ denotes the set of all subsets of $[0:127]$. Here, in a triple $(E, H_0, H_1) \in S$ the subset $H_0 \subseteq 2^{[0:127]}$ consists of all pitches of explicit note objects having musical onset time E and similarly the subset $H_1 \subseteq 2^{[0:127]}$ consists of all pitches of implicit note objects having musical onset time E. We want to emphasize that H_0 is a set of pitches which certainly appear in some interpretation of the score at onset time E, whereas H_1 represents just a set of *alternatives* of possible or likely pitches. Table 1 shows the encoding of the first few

notes of the Aria from Fig. 1.

$(0, \{55, 79\}, \emptyset)$,	$(1, \{59, 79\}, \emptyset)$,	$(2, \{62\}, \{79, 81\})$,	$(2\frac{3}{4}, \{83\}, \emptyset)$,
$(3, \{54, 81\}, \emptyset)$,	$(3\frac{1}{2}, \emptyset, \{78, 79\})$,	$(4, \{57\}, \{74, 76\})$,	$(5, \{62\}, \emptyset)$.

Table 1: First two measures of the Aria in Fig. 1 after preprocessing.

4.2 Matching and Cost Function

In this subsection we describe the SP-synchronization algorithm based on dynamic programming. The decisive ingredient for our approach is a carefully designed cost function which will be explained in detail.

After having preprocessed the score, we now turn to the data stream given in PCM-format. As described in Section 3, we extract a set of possible candidates of note objects given by their physical onset times and pitches (including in general erroneous objects). In view of the synchronization it is useful to further process this extracted data by quantizing the onset times. Simply speaking, we pool all note objects by suitably adjusting those physical onset times which only differ by some small value — e.g., smaller than some suitably chosen $\Delta > 0$ — since these note objects are likely to have the same musical onset time in the corresponding score format. After quantization we also may assume that the extracted PCM-data is given by some subset $P_\Delta \subset \mathbb{Q} \times 2^{[0:127]}$. Note that in the PCM-case there are only explicit note objects.

Altogether, we may assume that the score and the Δ-quantized extracted PCM-data are given by the sets

$$S = [(s_1, S_{01}, S_{11}), \ldots, (s_s, S_{0s}, S_{1s})]$$

and

$$P_\Delta = [(p_1, P_{01}), \ldots, (p_p, P_{0p})].$$

Here, the s_i, $1 \leq i \leq s$, denote the musical onset times and the p_j, $1 \leq j \leq p$, denote quantized physical onset times. Furthermore, $S_{0i}, S_{1i}, P_{0j} \subseteq [0 : 127]$ are the respective sets of pitches for the explicit and implicit objects.

On the basis of S and P_Δ we now accomplish the SP-synchronization. Since the score and the PCM-data represent the same piece of music, it is reasonable to assume $s_1 = p_1 = 0$ by possibly shifting the onset times. Now, the goal is to partially link the onset times $s_1, \ldots, s_s$ to $p_1, \ldots, p_p$ by maximizing the matches of the corresponding pitch sets. In the following, we formalize this approach.

Definition 4.1. A *Score-PCM-match* (SP-match) of S and P_Δ is defined to be a partial map $\mu\colon [1:s] \to [1:p]$, which is strictly monotonously increasing on its domain satisfying $(S_{0i} \cup S_{1i}) \cap P_{0\mu(i)} \neq \emptyset$ for all $i \in \mathrm{Domain}(\mu)$.

This definition needs some explanations. The fact that objects in S or P_Δ may not have a counterpart in the other data stream is modeled by the requirement that μ is only a partial function and not a total one. The monotony of μ reflects the requirement of faithful timing: if a note in S precedes a second one this also should hold for the μ-images of these notes. Finally, the requirement $(S_{0i} \cup S_{1i}) \cap P_{0\mu(i)} \neq \emptyset$ prevents that onset times are linked which are completely unrelated with respect to their pitches.

Obviously, there are many possible SP-matches between S and P_Δ. By means of some suitable cost function we can compare different matches. The goal is then to compute an SP-match minimizing the cost function. To simplify the notation we identify the partial function μ with its graph $\mathrm{Graph}(\mu) := \{(i_1, j_1), \cdots, (i_\ell, j_\ell)\}$, where $\{i_1 < \cdots < i_\ell\} \subseteq [1:s]$ and $\{j_1 = \mu(i_1) < \cdots < j_\ell = \mu(i_\ell)\} \subseteq [1:p]$. In the following definition we assign costs to each SP-match μ. In doing so, we make use of a parameter vector $\pi := (\alpha, \beta, \gamma, \delta, \zeta, \Delta) \in \mathbb{R}^6_{\geq 0}$ consisting of six real parameters which will be specified later.

Definition 4.2. Let $\pi := (\alpha, \beta, \gamma, \delta, \zeta, \Delta) \in \mathbb{R}^6_{\geq 0}$ be a parameter vector. Then the *SP-cost* of an SP-match μ w.r.t. π between some score S and some Δ-quantized set P_Δ of the corresponding PCM-document is defined

as

$$\begin{aligned} C_{\pi}^{\mathrm{SP}}(\mu|S,P_{\Delta}) \;:=\; & \alpha\cdot\sum_{(i,j)\in\mu}\Big(|S_{0i}\setminus P_{0j}|+\lambda(i,j)\Big)\\ & +\beta\cdot\sum_{(i,j)\in\mu}\Big(|P_{0j}\setminus(S_{0i}\cup S_{1i})|+\rho(i,j)\Big)\\ & +\gamma\cdot\sum_{k\notin\mathrm{Domain}(\mu)}\Big(|S_{0k}|+\sigma(k)\Big)\\ & +\delta\cdot\sum_{t\notin\mathrm{Image}(\mu)}|P_{0t}|\\ & +\zeta\cdot\sum_{(i,j)\in\mu}\Big|s_i-p_j\cdot\ell(S)/\ell(P)\Big|. \end{aligned}$$

This definition also requires some explanations. The sum corresponding to the factor α represents the cost of the non-matched explicit and implicit note objects of the score S. To be more accurate, the cardinality $|S_{0i}\setminus P_{0j}|$ measures the cost arising from the difference of the set S_{0i} of explicit note objects at the ith onset time of S and the set P_{0j} of explicit quantized note objects at the jth onset time of P_{Δ}. Furthermore, $\lambda(i,j)$ is defined to be 1 if and only if the score S has an implicit note object at the ith onset time and P_{Δ} has no counterpart at the jth onset time, i.e., $S_{1i}\neq\emptyset$ and $S_{1i}\cap P_{0j}=\emptyset$. In all other cases $\lambda(i,j)$ is defined to be 0. Next, we consider the sum corresponding to the factor β. The first summand in the brackets measures the cost of (possibly erroneously) extracted notes at the jth physical onset time whose pitches do not lie in $S_{0i}\cup S_{1i}$. Furthermore, $\rho(i,j)$ is defined to be $|P_{0j}\cap S_{1i}|-1$ if $P_{0j}\cap S_{1i}\neq\emptyset$. Otherwise $\rho(i,j)$ is defined to be 0. In other words, for the implicit note objects only *one* match is free of cost whereas each additional match is penalized. (This is motivated by the idea that all notes belonging to some realization of a fuzzy note are expected to have pairwise distinct physical onset times.) The sum corresponding to γ accounts for all onset times of the score which do not belong to the match μ. The first term within the brackets counts the number of explicit note objects at the kth onset time, $k\notin\mathrm{Domain}(\mu)$. Furthermore, $\sigma(k)$ is defined to be 1 if there is a non-matched implicit note object and 0 if there is no implicit note object at the kth onset time. (The idea is that a non-matched fuzzy note

should only be penalized by 1 since it only represents a set of alternatives.) The sum corresponding to δ accounts for the cost of those notes in P_Δ which do not have a counterpart in S. Finally, the last sum corresponding to ζ measures some kind of adjusted ℓ^1-distance (also known as Manhattan-distance) of the vector pairs $(s_i, p_j)_{(i,j)\in\mu}$, where $\ell(S)$ and $\ell(P)$ denote the differences of the last and the first musical respectively physical onset times (a kind of musical and physical duration). By this sum one penalizes matches with large relative time deviations thus preventing large global deviations in the synchronization.

In the following we fix some parameter vector π, a preprocessed score S, and quantized extracted PCM-data P_Δ. Note that if μ is an SP-match then also $\mu' := \mu\backslash\{(i,j)\}$ for some $(i,j) \in \mu$. An easy computation shows

$$\begin{aligned} C_\pi^{\mathrm{SP}}(\mu|S,P_\Delta) - C_\pi^{\mathrm{SP}}(\mu'|S,P_\Delta) \;=\; & \alpha\cdot\Big(|S_{0i}\setminus P_{0j}| + \lambda(i,j)\Big) \\ & +\beta\cdot\Big(|P_{0j}\setminus(S_{0i}\cup S_{1i})| + \rho(i,j)\Big) \\ & -\gamma\cdot\Big(|S_{0i}| + \sigma(i)\Big) \\ & -\delta\cdot|P_{0j}| \\ & +\zeta\cdot\Big|s_i - p_j\cdot\ell(S)/\ell(P)\Big|. \end{aligned} \tag{2}$$

Now, one can determine a cost-minimizing SP-match by means of dynamic programming. We recursively define a matrix $C = (c_{ij})$ with $i \in [0:s]$ and $j \in [0:p]$. First, initialize $c_{0j} := c_{i0} := C_\pi^{\mathrm{SP}}(\emptyset|S,P_\Delta)$ for all $i \in [0:s], j \in [0:p]$. Note that this accounts for the costs that there is no match at all between S and P_Δ. At position $(i,j) \in [1:s]\times[1:p]$ the value c_{ij} expresses the cost for a cost-minimizing SP-match within the subset $[1:i]\times[1:j] \subset [1:s]\times[1:p]$. Hence, c_{sp} expresses the minimal cost of a global SP-match. For $(i,j) \in [1:s]\times[1:p]$, the value c_{ij} is defined as

$$c_{ij} := \min\{c_{i,j-1}, c_{i-1,j}, c_{i-1,j-1} + d_{ij}^{\mathrm{SP}}\},$$

where

$$d_{ij}^{\mathrm{SP}} := \begin{cases} \text{right hand side of Eq. (2)}, & \text{if } (S_{0i}\cup S_{1i})\cap P_{0j} \neq \emptyset, \\ 0, & \text{otherwise.} \end{cases}$$

Using the resulting matrix C, the procedure of Fig. 13 computes a cost-minimizing SP-match.

```
SCORE-PCM-SYNCHRONIZATION(C,s,p)
1   i := s, j := p, SP-Match := ∅
2   while (i > 0) and (j > 0)
3      do if c[i,j] = c[i,j-1]
4            then j := j - 1
5         else if c[i,j] = c[i-1,j]
6            then i := i - 1
7         else SP-Match := SP-Match ∪ {(i,j)},
                 i := i - 1, j := j - 1
8   return SP-Match
```

Figure 13: Procedure for computing the cost-minimizing SP-match.

In the next section we give an example to illustrate this procedure and report some of our experiments. As we mentioned before, SM- and MP-synchronization can be done similarly to the SP-case. Furthermore, other synchronization problems such as synchronization of two PCM-data streams P_1 and P_2 (P_1P_2-synchronization) may be achieved by using a score S as a reference and carrying out both an SP_1- and an SP_2-synchronization.

We conclude this section with some comments on the parameter vector $\pi := (\alpha, \beta, \gamma, \delta, \zeta, \Delta)$. In most of our experiments we set the quantization constant to $\Delta = 50$ ms. This threshold was chosen since it represents a good compromise between psychoacoustically distinguishable asynchronisms of chords and the shortest possible musical note durations. By the parameters α and β one can weight the cost for the symmetric difference of pitch sets corresponding to matched onset times, whereas by the parameters γ and δ one can weight the cost of those note objects which do not have a counterpart in the other data stream. One meaningful standard choice of the parameters is $\alpha = \beta = \gamma = \delta = 1$. However, if one wants to penalize non-matched onset times, for example, one may increase γ and δ. In the case $\zeta = 0$ the last sum of the cost function remains unconsidered. Increasing ζ will hamper matches (i, j) whose onset times s_i and p_j differ too much with respect to their relative positions in their respective data stream. In other words, excessive global time divergence

in the synchronization of the two data streams can be controlled.

4.3 An Example

We illustrate the SP-synchronization by means of the Aria shown in Fig. 1. After the preprocessing step the score is given by the data stream $S = [(s_1, S_{01}, S_{11}), \ldots, (s_s, S_{0s}, S_{1s})]$ with $s = 23$. For example, the set $S_{01} = \{55, 79\}$ contains the two pitches at musical onset time $s_1 = 0$ corresponding to the first note g with MIDI-pitch 55 in the left and the first note g^2 with MIDI-pitch 79 in the right hand. As another example, the trill in the right hand of the first measure is modeled by a fuzzy note (implicit note object) at musical onset time $s_3 = 2$ given by $S_{13} = \{79, 81\}$. Here the MIDI-pitches 79 and 81 correspond to the two possible notes a^2 and h^2 involved in the trill. Table 2 shows all the note objects of S corresponding to the first two measures of the Aria.

S				P_Δ		
i	s_i	S_{0i}	S_{1i}	j	p_j	P_{0j}
1	0	{55, 79}	$\emptyset$	1	0	{55, 67}
2	1	{59, 79}	$\emptyset$	2	1.23	{59, 79}
3	2	{62}	{79, 81}	3	2.44	{55, 62}
				4	2.56	{62, 79}
				5	2.68	{62, 81}
4	2.75	{83}	$\emptyset$	6	3.58	{83}
5	3	{54, 81}	$\emptyset$	7	3.86	{54, 66}
6	3.5	$\emptyset$	{78, 79}	8	4.47	{79}
				9	4.75	{78}
7	4	{57}	{74, 76}	10	5.06	{57, 76}
				11	5.71	{74}
8	5	{62}	$\emptyset$	12	6.39	{57, 62}

Table 2: Matching of the preprocessed score data stream S and the data stream P_Δ consisting of the quantized note parameters extracted from the PCM-data stream for the first two measures of the Aria.

The PCM-version P represents a recording of the Aria performed on a Steinway grand piano. The physical length is $\ell(P) = 13$ sec. After

extraction of the note parameters (shown in Fig. 8) and Δ-quantization with $\Delta = 50$ ms one obtains $P_\Delta = [(p_1, P_{01}), \ldots, (p_p, P_{0p})]$ with $p = 38$ note objects. Note that the number $p = 38$ of note objects in P_Δ is much greater than the number $s = 23$ in S. The reason is that P_Δ not only contains for each trill and appoggiatura several note objects (which correspond in S to a single implicit note object, respectively) but also erroneous extraction results. Table 2 shows all the note objects of P_Δ corresponding to the first two measures of the Aria. Here, p_j denotes the physical note onset time for the positions $j = 1, 2, \ldots, 38$ measured in seconds.

This example also illustrates several typical phenomena appearing in the extraction step. For example, at position $j = 1$ the set $P_{01} = \{55, 67\}$ contains the MIDI-pitch 67 instead of the expected pitch 79. This may be explained as follows: at onset time p_1 the notes g of MIDI-pitch 55 and g^2 of MIDI-pitch 79 are played. Note that the sound corresponding to the note g contains the harmonics g^1 and g^2 with high energy. Because of the complex interaction of all the harmonics of all played notes at a given time, it is nearly impossible to decide whether g^1 or g^2 are fundamental frequencies of independent notes or just harmonics of g. At position $j = 1$ the extraction algorithm has, instead of g^2, mistakenly interpreted g^1 of MIDI-pitch 67 as an independent note. Such "octave errors" are typical for the extraction step. (To tackle this problem one may restrict oneself to only considering pitches which are reduced modulo 12 when using the note parameters as input for the synchronization algorithm.)

Furthermore, the "erroneous" extraction of the note g of MIDI-pitch 55 at position $j = 3$ may be ascribed to the fact that this note has been played at onset time p_1 and still continues to sound at time p_3. The extraction algorithm mistakenly rediscovers this note as a "new" note object at time p_3.

As a final example, we want to mention that the trill over a^2 in the first measure, corresponding to the fuzzy note $S_{13} = \{79, 81\}$ in the score, has several counterparts on the PCM-side, namely P_{03}, P_{04}, and P_{05}. In passages with many short notes having similar spectral components the extraction algorithm is error-prone leading to many additional pitches.

In spite of such "erroneously" extracted note parameters the SP-synchronization works quite well. Using the cost function $C_\pi^{\mathrm{SP}}(\mu|S, P_\Delta)$ from Definition 4.2 with prototypical parameter vector $\pi = (1, 1, 1, 1, 0, 50)$

leads to the cost matrix $C = (c_{ij})$, which is shown in Fig. 3 for the note objects corresponding to the first two measures.

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	102	102	102	102	102	102	102	102	102	102	102	102	102
1	102	**98**	98	98	98	98	98	98	98	98	98	98	98
2	102	98	**94**	94	94	94	94	94	94	94	94	94	94
3	102	98	94	**90**	90	90	90	90	90	90	90	90	90
4	102	98	94	90	90	90	**88**	88	88	88	88	88	88
5	102	98	94	90	90	88	88	**85**	85	85	85	85	85
6	102	98	94	90	88	88	88	85	**83**	83	83	83	83
7	102	98	94	90	88	88	88	85	83	83	**79**	79	79
8	102	98	94	90	88	86	86	85	83	83	79	79	**77**

Table 3: Cost matrix $C = (c_{ij})$ for the note objects of S (row index $i = 0, \ldots, 8$) and P_Δ (column index $j = 0, \ldots, 12$) corresponding to the first two measures of the Aria.

We recall that the number c_{ij} expresses the costs for a cost-minimizing SP-match of the note objects of S up to position i with the note objects of P_Δ up to position j. (For c_{ij} all note objects above the positions i and j respectively are not considered and remain unmatched.) For example, the number $c_{00} = 102$ is the cost for the empty match, i.e., the cost where there is no match at all. Hence c_{00} is maximal among all numbers c_{ij}. When the first note object S_{01} at position $i = 1$ of S is matched with the first note object P_{01} at position $j = 1$ of P and all other note objects are left unconsidered, one gets the cost $c_{11} = 98$ which is lower than c_{00}, and so on. Now, using dynamic programming, one gets the cost minimizing global match μ. In Table 3 the matrix entries c_{ij} corresponding to a matched pair (i, j) (i.e., $\mu(i) = j$) are written in boldface, i.e., the matches of the first two measures are $\{(1,1),(2,2),(3,3),(4,6)\}, \{(5,7),(6,8),(7,10)$ and $(8,12)\}$. In this example, the fuzzy note S_{13} is matched with the "first note" P_{03} of the three note objects belonging to the trill realization of the interpreted PCM-version. Furthermore, the SP-algorithm has matched the appoggiatura of the score S modeled by the fuzzy notes S_{16} at position $i = 6$ with the corresponding first note object P_{08} of the appoggiaturas of P_Δ at position $j = 8$, and similarly S_{17} with $P_{0,10}$.

In Fig. 14 the computation of an optimal SP-match is illustrated

by means of a 3D-plot which visualizes the cost matrix C. The right coordinate represents the indices j for the onset times s_j, $1 \leq j \leq s$, $s = 38$. The middle coordinate represents the indices i for the onset times p_i, $1 \leq i \leq p$, $p = 23$. Finally, the vertical coordinate shows the values c_{ij} of the matrix C. The path indicates the cost minimizing match representing the solution of the synchronization problem.

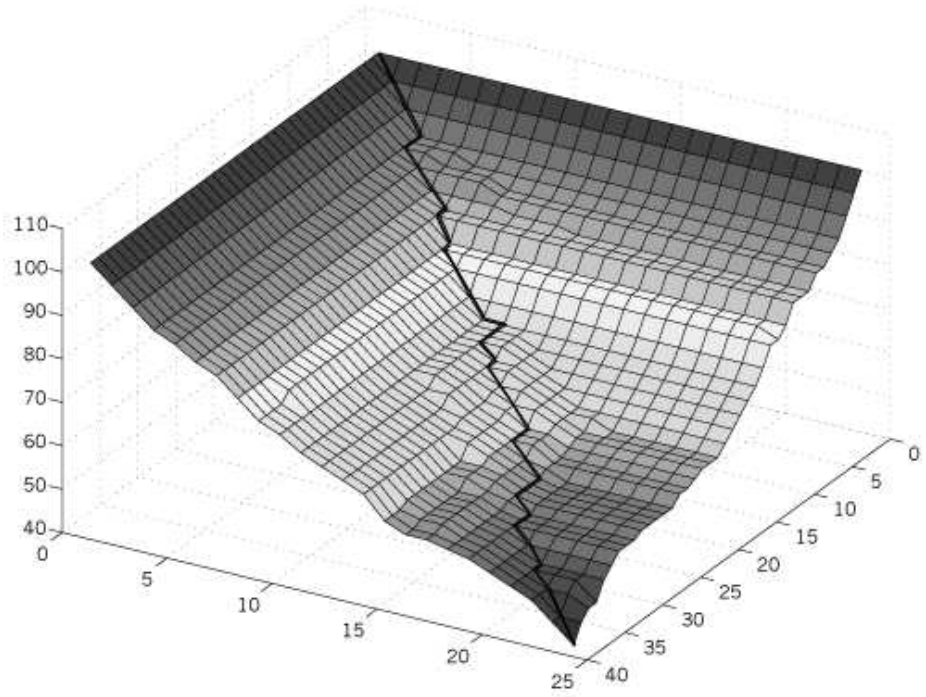


Figure 14: The cost matrix C and a cost-minimizing SP-match for the Aria.

5 Experiments

We have implemented a prototype of the extraction algorithms from Section 3 and the synchronization algorithms in the MATLAB programming language and tested our algorithms for SM-, SP-, and MP-synchronization on a variety of classical polyphonic piano pieces of different complexity and length (ranging from 10 to 60 seconds) played on various instruments. Furthermore, we have systematically generated a library of more

than one hundred test pieces both in MIDI- and PCM-format played on a MIDI-piano, a Steinway grand piano, and a Schimmel piano. In some of those pieces our player has deliberately built in excessive accelerandi, ritartandi, rhythmic distortions, and wrong notes. Even in these extreme situation, where one unsurprisingly has many "erroneously" extracted note objects which considerably differ from the score-data, our SP-synchronization algorithm resulted in good overall global matches which are sufficient for the applications mentioned in the introduction. Even more, in case of rather accurate extracted note parameters our synchronization algorithms could resolve subtle local time variations in some interpreted version of the piano piece. For further details and results of our experiments we refer to Arifi (2002).

We close this section by describing one of our experiments where we started with an uninterpreted score-like MIDI-version and an interpreted PCM-version of some piano piece. Using the results of our MP-synchronization, we automatically modified the onset times of the MIDI-stream to correspond to the PCM-stream in view of the global tempo and the local tempo variations. This resulted in some "expressive" MIDI-version which represented a sonification of our synchronization results. In case of good extraction parameters the so modified MIDI-version sounded rhythmically like a real interpretation of the underlying piano piece.

6 Conclusions

In this paper we have discussed algorithms for the automatic synchronization of different versions of some polyphonic piano piece given in different data formats (score, MIDI, PCM). Our implementation yields good synchronization results even for complex PCM-based polyphonic piano CD-recordings. One of the decisive features is a carefully designed cost function which not only penalizes non- or partially matched note objects but also large relative global time deviations (in case $\zeta > 0$). The parameter vector π allows to weight different aspects in the matching process and leaves room for experiments.

Our cost function may further be improved and extended in various ways. One improvement may result if one allows that a fuzzy note of the score data stream S may be assigned to several note objects of P_Δ.

For example, in Table 2 a simultaneous match of the trill S_{13} at position $i = 3$ with all three note objects P_{03}, P_{04}, P_{05} (leading to the matches $(3,3),(3,4)$ and $(3,5)$) would better reflect the correspondence of the score parameters and the extracted note parameters of the PCM-data. In this case the match μ would be a relation rather than a partial function. Another improvement may be achieved by exploiting "safe" note objects, i.e., note objects which can be detected correctly by the extraction algorithm with high probability. Such "safe" note objects may be low notes whose fundamental pitch cannot be mixed up with harmonics of other notes; or notes with some "rare" spectral components sticking out from the overall harmonic structure of the underlying piece. Here, the prior knowledge of the score may be used to determine such safe note objects which should then be matched with preference. This can be achieved by modifying the cost function in such a way that non-matching of safe notes leads to disproportionally higher costs. In the Aria, for example, the first notes of each measure in the left hand seem to be good candidates for such safe notes.

Finally, we note that our current system works off-line, where the bottle-neck lies in the preprocessing step needed to extract note parameters from the PCM-files (where we up to now did not use any score information). An ongoing research project is to exploit the score information already in the extraction step. (See also Scheirer (1995) for a similar approach.) This prior knowledge allows to use prediction methods (in particular Kalman-filtering) which in connection with time-varying comb filters may result in extraction algorithms running in real-time. Such fast algorithms may be at the expense of the quality of the extraction parameters. However, even low quality and coarse parameters may be sufficient for a successful synchronization when using a suitably designed cost function which is robust under erroneous parameters.

References

Arifi, Vlora. *Algorithmen zur Synchronisation von Musikdaten im Paritur-, MIDI- und PCM-Format.* PhD thesis, Universität Bonn, Institut für Informatik, 2002.

Blackham, E.D. "Klaviere" in *Die Physik der Musikinstrumente*, 2. Au-

flage, Spectrum, Akademischer Verlag, 1998.

Bobrek, Miljko, and Koch, Daniel. "Music Signal Segmentation Using Tree-Structured Filter Banks," *Journal of Audio Engineering Society*, Vol. 46, No. 5 (1998), 412–427.

Cano, P., Loscos, A., Bonda, J. "Score-Performance Matching using HMMs," *Proceedings of ICMC.* (1999), 441–444.

Cemgil, A. T., Desain, P., Kappen, B. "Rhythm Quantisation for Transcription," *Computer Music Journal*, Vol. 24 No. 2 (2000), 60–76.

Cemgil, A. T., Kappen, B., Desain, P., Honing, H. "On tempo tracking: Tempogram Representation and Kalman filtering," *Proceedings of ICMC* (2000), 352-355.

Dannenberg, R. B. "An On-Line Algorithm for Real-Time Accompaniment," *Proceedings of ICMC* (1984).

Dannenberg, et al. *Automated musical accompaniment with multiple input sensors.* US Patent #5521324 (1994).

Dannenberg, R. B., Mukaino, H. "New Techniques for Enhanced Quality of Computer Accompaniment," *Proceedings of ICMC* (1988).

Desain, P., Honing, H., Heijink, H. "Robust Score-Performance Matching: Taking Advantage of Structural Information," *Proceedings of ICMC* (1997), 377–340.

Foote, J. "ARTHUR: Retrieving Orchestral Music by Long-Term Structure," *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)* (2000).

Foote, J., Uchihashi, S. "The Beat Spectrum: A New Approach To Rhythm Analysis," *Proc. International Conference on Multimedia and Expo (ICME)* (2001).

Foster, S., Schloss, W.A., Rockmore, A.J. "Towards an Intelligent Editor of Digital Audio: Signal Processing Methods" *Computer Music Journal*, Vol. 6, No. 1 (1982).

Goto, M. "An Audio-based Real-time Beat Tracking System for Music With or Without Drum-sounds," *Journal of New Music Research*, Vol.30, No.2 (2001), 159–171.

Large, E. W. "Dynamic programming for the analysis of serial behaviours," in *Behaviour Research Methods, Instruments, & Computers* (1993).

Mazzola, Guerino. *The Topos of Music.* Birkäuser, 2002.

Ortiz-Berenguer, L. I., Casajús-Quirós, F. J. "Pattern recognition of piano chords based on physical model," *AES Convention Paper, Presented at the 112th Convention* (2002), 10–13.

Raphael, C. "Automatic Segmentation of Acoustic Musical Signals Using Hidden Markov Models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 21, No. 4, (1999).

Raphael, C. "A Probabilistic Expert System for Automatic Musical Accompaniment." *Jour. of Comp. and Graph. Stats.*, Vol. 10, No. 3 (2001), 487–512.

Scheirer, E. D. *Extracting Expressive Performance Information from Recorded Music.* M. S. thesis, MIT Media Laboratory, 1995.

Scheirer, E. D. "Tempo and Beat Analysis of Acoustic Musical Signals," *J. Acoust. Soc. Am.* Vol. 103, No. 1 (1998), 588–601.

Selfridge-Field, Eleanor (ed.). *Beyond MIDI, The Handbook of Musical Codes.* MIT Press, 1997.

Vercoe, B. "The Synthetic Performer in the Context of Live Performance," *Proceedings of ICMC* (1984).