Score-PCM Music Synchronization Based on Extracted Score Parameters

Vlora Arifi, Michael Clausen, Frank Kurth, and Meinard Müller

Universität Bonn, Institut für Informatik III
Römerstr. 164, D-53117 Bonn, Germany
{arifi, clausen, kurth, meinard}@cs.uni-bonn.de

Abstract. In this paper we present algorithms for the automatic time-synchronization of score-, MIDI- or PCM-data streams which represent the same polyphonic piano piece. In contrast to related approaches, we compute the actual alignment by using note parameters such as onset times and pitches. Working in a score-like domain has advantages in view of the efficiency and accuracy: due to the expressiveness of score-like parameters only a small number of such features is sufficient to solve the synchronization task. To obtain a score-like representation from the waveform-based PCM-data streams we use a preprocessing step to extract note parameters. In this we use the concept of novelty curves for onset detection and multirate filter banks in combination with note templates for pitch extraction. Also the data streams in MIDI- and score-format have to be suitably preprocessed. In particular, we suggest a data format which handles possible ambiguities such as trills or arpeggios by introducing the concept of fuzzy-notes. Further decisive ingredients of our approach are carefully designed cost functions in combination with an appropriate notion of alignment which is more flexible than the classical DTW concept. Our synchronization algorithms have been tested on a variety of classical polyphonic piano pieces recorded on MIDI- and standard acoustic pianos or taken from CD-recordings.

1 Introduction

Modern digital music libraries consist of large collections of documents containing music data of diverse characteristics and formats. For example, for one and the same piece of music, the library may contain the corresponding score in the Capella or Score format, as MIDI-files, and several interpretations by various musicians as 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 align data streams of different data formats representing a similar kind of information. In particular, in the framework of audio by *synchronization* we mean a 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).

U.K. Wiil (Ed.): CMMR 2004, LNCS 3310, pp. 193–210, 2004.

Such synchronization algorithms have applications in many different scenarios: following score-based music retrieval, linking structures can be used to access a suitable audio CD accurately to listen to the desired part of the interpretation. A further future application is the automatic annotation of a piece of music in different data formats to support content-based retrieval and browsing. 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 a reading aid of scores.

In the following, we concentrate on three widely used formats for representing music data: the symbolic *score format* contains information on the notes such as musical onset time, pitch, duration, and further hints concerning the agogic and dynamics. The purely physical *PCM-format* encodes the waveform of an audio signal as used for CD-recordings. The *MIDI-format* may be thought of as a hybrid of the last two data formats containing content-based information on the notes as well as agogic and dynamic niceties of some specific interpretation. These different data formats lead to various synchronization problems as illustrated by Figure 1.

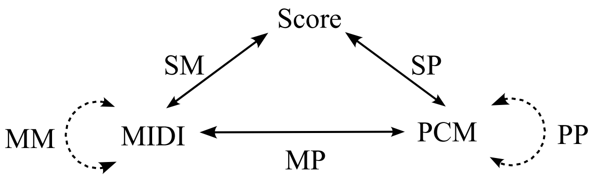


Fig. 1. Overview on various synchronization problems.

In this paper we concentrate on the case of Score-PCM (SP) synchronization where one data stream is given in a score-like format (in this case the note parameters are explicitly available) and the other data stream is given in the PCM-format. The other cases such as SM- or MP-synchronization are even easier or can be done in a similar fashion (see [1]).

As will be discussed in Section 2 there are various strategies to make the score-like data stream comparable to the PCM-data stream. In all approaches the data streams are divided up into sequences of frames which are aligned by techniques of dynamic time warping (DTW). Since the memory requirement as well as the running time of this approach is at least proportional to the product of the lengths of the two sequences to be aligned, efficiency becomes an important issue. Therefore, in contrast to the approaches described in [17,18], we suggest to use score-like features such as onset times and pitches for the alignment process. Due to the expressiveness of such note parameters only a small number of features is sufficient to solve the synchronization task. Furthermore, a very good temporal accuracy of the alignment can be achieved.

The major problem with this approach is that a PCM-data stream does not explicitly contain information on the notes. Therefore, in a preprocessing step, we first extract information such as note onsets and pitches from the PCM-recording in order to make it comparable in the score-like domain. However, 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 (see also Section 2). Especially, in the most general case of orchestral music the extraction problem (not to mention the transcription problem) is a largely open problem which seems 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., music data from a piano CD. This in general extremely complex data leads to many erroneous extraction results which would not be acceptable when treated as a transcription into a score-like format of the original piece of music. However, the extracted data — even from complex piano pieces — is good enough to ensure success in view of the synchronization problem.

One major problem in Score-PCM synchronization results from the fact that 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. To cope with ambiguities in the score, such as trills, arpeggios, or grace notes, we introduce the concept of fuzzy-notes which allows a set of alternative notes in the alignment process. Furthermore, concerning the PCM-based interpretation one often has to deal with wrong, additional, or missing notes. Using classical DTW-based approaches for the alignment, as e.g. described in [13,17,18], matches are forced by the local constraints imposed on the warping paths — even in regions where there are no equivalent events in the score- and PCM-data streams. From our experiments we conclude that it is preferable to avoid such kinds of matches rather than having "erroneous" matches. To this means, we introduce a more general notion of a so-called *Score-PCM match* which allows to skip notes and whole regions, in which the PCM-version differs considerably from the score.

The rest of this paper is organized as follows. In Section 2, we discuss recent approaches to Score-PCM synchronization and briefly review some literature related to the extraction and transcription problem. Then, in Section 3, we summarize our system for the extraction step of musically relevant parameters from PCM-data streams. The actual synchronization algorithms, based on a carefully designed cost function, are described in Section 4. Finally, Section 5 contains an examples, Section 6 summarizes some of our experiments, and Section 7 closes with concluding remarks and possible future research directions.

2 Background and Related Work

The synchronization problem in music and in particular the related problems of automatic score following and automatic music accompaniment has been a research field for many years. Dannenberg et. al. describe in [3] and [4] 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. Raphael [14] has developed a system for automatic musical accompaniment of an oboe based on Hidden Markov Models. Desain et. al. describe in [5] a general sequential and tree-based score-performance matching algorithms. A similar problem addresses Large in [9] 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 with no such explicit and clean parameters. Further links may be found in the overview article [12]. In the following, we discuss two recent contributions by Turetsky et al. [18] and by Soulez et al. [17] which also address the problem of Score-PCM synchronization.

To make the data sequences comparable, Turetsky et al. [18] first convert the score-like version (given as MIDI data stream) into a PCM-data stream using a suitable synthesizer. Then, the two PCM-data streams are analyzed by means of a Short-Time-Fourier Transform (STFT) which in turn yields a sequence of suitable feature vectors. A pairwise comparison of those feature vectors with respect to a suitable local distance measure is used to compute a cost matrix. Based on this matrix, the best alignment is derived by means of dynamic time warping. As reported in [18], good synchronization results are achieved in the case of commercial pop music. In this genre, one generally has a relatively constant tempo and also a clear rhythm resulting in regular patterns in the spectral domain. Actually, Turetsky et al. convert the Score-PCM synchronization problem into a PCM-PCM synchronization problem — in the alignment process the note parameters given by the score-data stream are not used. On the one hand this makes their algorithm applicable for synchronizing arbitrary types of music. On the other hand, when dealing with e.g. classical music, where one can have strong local time deviations and only small spectral variations, algorithms relying solely on the comparison of the spectral information of the underlying PCM-data streams will lead to significant synchronization error rates.

In the approach of Soulez et al. [17] the score-data stream is used to generate a sequence of attack-, sustain- and silence models corresponding to note onsets, pitches and rests. As in [18], the PCM-data stream is converted into a sequence of spectral vectors using a STFT. Based on suitably defined local distance measures, which allow a comparison between the note models and the spectral vectors, a cost matrix is computed. Again dynamic time warping is applied to derive the alignment. In contrast to [18], Soulez et al. explicitly use the note parameters such as onset times and pitches in the synchronization process which results in a more robust algorithm w.r.t. local time deviations and small spectral variations.

Both approaches [17] and [18] have the following drawbacks due to the STFT used for the analysis of the PCM-data stream. Firstly, the STFT computes spectral coefficients which are *linearly* spread over the spectrum resulting in a bad resolution of low frequencies. Therefore, in the case of low notes one has to rely on the harmonics. This is problematic in polyphonic music where one often has the situation that harmonics and fundamental frequencies of different notes coincide. Secondly, in order to obtain a satisfying time resolution one has to work with a relatively large number of feature vectors on the PCM-side. (For example, even with a rough time resolution of 46 ms as suggested in [18] these are more than 20 feature vectors per second.) This leads to huge memory requirements as well as long running times in the DTW computation, which are both proportional to the product of the lengths of the two sequences to be aligned.

As mentioned in the last section, the extraction problem of score-like note parameters from waveform-based PCM-data itself constitutes an active research area with many yet unsolved problems. We only give a small choice of relevant literature where the reader finds further links. As an example, we mention the approach of Raphael [15] who uses Hidden Markov Models to transcribe polyphonic piano music. Klapuri et. al. [10] use moving-average techniques to extract note pitches in polyphonic musical signals. As is also mentioned by the authors, the extraction of such parameters in polyphonic music still leaves a lot of work to do. Foote [6] uses the concept of the so-called novelty score to automatically segment audio recordings. In Section 3 we use a similar technique to extract candidates of onset times. Bobrek et. al. [2] use filter bank techniques in combination with note templates for the transcription of polyphonic piano music. We have modified their approach to extract the pitches of the previously determined onset candidates. Finally, we want to mention the comprehensive book [11] by Mazzola who gives, among many related topics, a detailed account on local tempo variations resulting from expressiveness in performances.

3 Feature Extraction

In this section we summarize our system for extracting note parameters from the PCM-data stream. Using several established tools from audio signal processing, our main contributions are a refined template matching algorithm for polyphonic pitch extraction and a two-step algorithm for note onset detection.

Figure 2 shows the overall feature extraction algorithm. An input PCM-signal is transformed to a subband representation using a multirate filterbank. Simultaneously, a two-stage peak-picking algorithm detects probable note onset positions. According to those onset positions, the subband representation is split into time intervals. For each interval, we calculate an energy vector with components corresponding to the subbands: each component contains the total energy within the interval of the respective subband. Then, for each energy vector a pitch extraction based on a template matching algorithm is performed. The pitch extraction yields a set of notes for the corresponding time interval. The feature extraction algorithm outputs a note object for each note in each time interval, where a note object consists of an onset time and a pitch information.

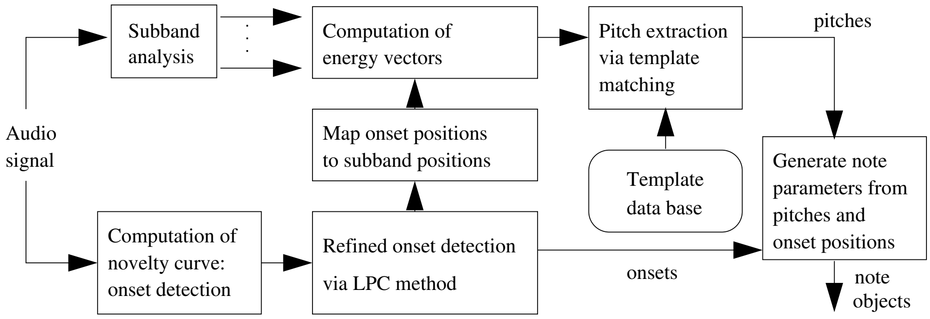

Audio signal
Subband analysis
Computation of energy vectors
Pitch extraction via template matching
pitches
Map onset positions to subband positions
Template data base
Generate note parameters from pitches and onset positions
Computation of novelty curve: onset detection
Refined onset detection via LPC method
onsets
note objects

Fig. 2. Diagram of the feature extraction algorithm.

We now briefly discuss the components of our algorithm. For onset detection, we first calculate a *novelty*[1] *curve* of the input signal. This step basically consists of a STFT with a step size of 23 ms, where for each step a novelty value is calculated by summing over the absolute changes of the last two steps' magnitude spectra. Peaks in the resulting novelty curve constitute candidate onset times. As the time-resolution of 23 ms is very rough, a postprocessing of all candidate onset-times is performed based on linear prediction. In linear predictive coding (LPC), the harmonic parts of a signal segment are summarized in a few prediction coefficients which may in turn be used to predict the short-time signal behavior. A method proposed in [7] compares ratios of prediction errors resulting from crossover predictions of neighboring signal segments to detect significant signal changes. We adopt this method to increase the time resolution of our candidate onset times to about 10 ms. For a detailed account we refer to [1].

The simultaneously applied subband filterbank transforms the input signal into $M = 224$ subband signals. The filterbank is realized by a tree structured cascade of orthogonal 2-band filterbanks. 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 subsequent template matching algorithm templates can be uniquely assigned to energy vectors. For further details on the filter bank tree structure we refer to [2].

For pitch detection, template matching is performed w.r.t. a template data base (TDB) consisting of one template for each musical note. A template is an M-point vector representing the energy distribution of a certain note over the above subbands. The templates were recorded using a Yamaha GranTouch E-Piano. We furthermore 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.

[1] We adopted the term *novelty* from [6] even though our definition of *novelty curve* does only conceptually correspond to the one used by Foote.

Starting with an initial energy vector, our template matching algorithm roughly tries to select a template from the TDB which optimally fits the energy vector w.r.t. a certain criterion. 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 a specified 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 a 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. In [2] the authors, 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.

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). A detailed discussion is beyond the scope of this paper. For further details, we refer to [1].

4 Synchronization Algorithms

In this section we describe the actual synchronization algorithms. Due to space limitation, we only consider 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 [1]).

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). We distinguish between two kinds of note objects: *explicit* and *implicit* ones. For *explicit* objects all note parameters such as measure, beat, duration, and pitch are given explicitly. In view of the synchronization algorithm we only use the musical onset time and pitch. We represent each explicit note object by a tuple $(e, p) \in \mathbb{Q} \times [0 : 127]$, where $[a : b] := \{a, a + 1, \ldots, b\}$ for integers a and b. Here, we identify a pitch with the corresponding MIDI pitch given by an integer between 0 and 127. Furthermore, the musical onset time $e \in \mathbb{Q}$ is computed by $e = r \cdot (m - 1) + b$ if the piece of music has r beats per measure, where $m \in \mathbb{N}$ and $b \in \mathbb{Q}$ denote the measure and beat respectively of the explicit note object. For example, the first and fourth explicit note object in the right hand of the Aria (Fig. 3) are given by $(0, 79)$ and $(2.75, 83)$ respectively.

By an *implicit* note object we understand notes or a group of notes with an 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) consisting of an onset time $e \in \mathbb{Q}$ and a set of alternative pitches $H \subset [0 : 127]$. Then an implicit note object, such as a trill, is represented by the musical onset time of a certain main

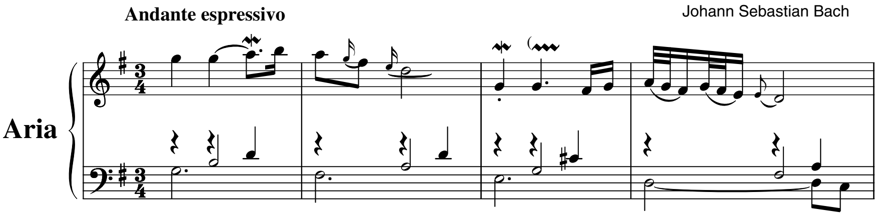


Fig. 3. First four measures of the *Aria con Variazioni* by J. S. Bach, BWV 988.

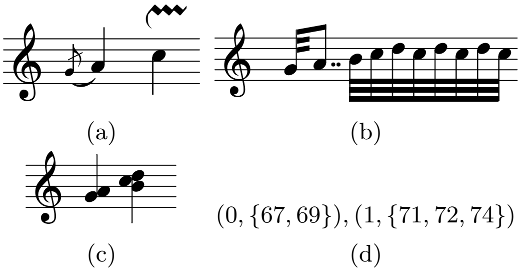


Fig. 4. Appoggiatura and trill, (a) notation, (b) possible realization, (c) fuzzy note, (d) encoding.

note and the set of all pitches appearing in a possible realization of this object (see Fig. 4 for an illustration). For example, the third note object of the first measure in the right hand of the Aria (Fig. 3) is given by $(3, \{79, 81\})$.

Using this encoding we may assume that a score is given by a 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 $H_0 \subset 2^{[0:127]}$ consists of all pitches of explicit note objects having musical onset time e and similarly the subset $H_1 \subset 2^{[0:127]}$ consists of all pitches of implicit note objects having musical onset time e.

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 a suitably chosen $\varDelta > 0$ — 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 a subset $P_\varDelta \subset \mathbb{Q} \times 2^{[0:127]}$. Note that in the PCM-case there are only explicit note objects.

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

$$S = [(s_1, S_{01}, S_{11}), \ldots, (s_s, S_{0s}, S_{1s})] \quad \text{and} \quad P_\varDelta = [(p_1, P_{01}), \ldots, (p_p, P_{0p})].$$

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, $S_{0i}, S_{1i}, P_{0j} \subset [0 : 127]$ are the respective sets of pitches for the explicit and implicit objects.

On the basis of S and $P_\varDelta$ 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 1. *A* score-PCM-match *(SP-match) of S and $P_\varDelta$ 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 $P_\varDelta$ 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 $P_\varDelta$. By means of a 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 this, we make use of a parameter vector $\pi := (\alpha, \beta, \gamma, \delta, \zeta, \varDelta) \in \mathbb{R}^6_{\geq 0}$ consisting of six real parameters which will be specified later.

Definition 2. *With respect to the parameter vector $\pi := (\alpha, \beta, \gamma, \delta, \zeta, \varDelta) \in \mathbb{R}^6_{\geq 0}$ the* SP-cost *of an SP-match μ between a score S and a $\varDelta$-quantized set $P_\varDelta$ of the corresponding PCM-document is defined as*

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

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 of explicit note objects at the ith onset time of S and the set 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 a 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 an 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 a parameter vector π, a preprocessed score S, and quantized extracted PCM-data P_Δ. Note that if μ is an SP-match then also $\mu' := \mu \setminus \{(i,j)\}$ for some $(i,j) \in \mu$. An easy computation shows

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

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

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

Using the resulting matrix $C = (c_{ij})$, the following algorithm computes a cost-minimizing SP-match:

```
SCORE-PCM-SYNCHRONIZATION($C$,$s$,$p$)
  $i := s$, $j := p$, SP-Match $:= \emptyset$
  while $(i > 0)$ and $(j > 0)$
     do if $c[i,j] = c[i,j-1]$
           then $j := j - 1$
        else if $c[i,j] = c[i-1,j]$
           then $i := i - 1$
        else SP-Match $:=$ SP-Match $\cup \{(i,j)\}$, $i := i - 1$, $j := j - 1$
  return 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 ($\mathrm{P}_1\mathrm{P}_2$-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.

5 An Example

We illustrate the SP-synchronization by means of the example from Fig. 3. The score S represents the first four measures of the Aria. The PCM-version P represents a recording or the same measure performed on a Steinway grand piano. The physical length is $\ell(P) = 13$ sec. The quantization constant is set to $\varDelta = 50$ ms. In the following, we restrict ourselves to the second measure, where two appoggiaturas appear in the right hand of the score. Those are modeled by fuzzy notes. Table 1 shows the note objects of the score S and the $\varDelta$-quantized extracted PCM-data $P_\varDelta$.

S				$P_\varDelta$		
i	s_i	S_{0i}	S_{1i}	j	p_j	P_{0j}
5	3	{54, 81}	$\emptyset$	7	3.86	{54, 81}
6	3.5	$\emptyset$	$\{78, 79\}$	8	4.47	{79}
				9	4.75	{66, 78}
7	4	{57}	{74, 76}	10	5.06	{57, 66, 76}
				11	5.71	{57, 74}
8	5	{62}	$\emptyset$	12	6.39	{57, 62}

Table 1. Note objects for the score S and the $\varDelta$-quantized extracted PCM-data $P_\varDelta$ for the second measure of the aria (cf. Fig. 3).

This example also illustrates two typical phenomena appearing in the extracted PCM-data. Firstly, in position $j = 10$ the extracted note of pitch 57 also appears in positions 11 and 12. This can be explained as follows: this note continues to sound and, at every new note attack (e.g., note of pitch 74 at position 11 or note of pitch 62 at position 12), the extraction algorithm again interprets the note of pitch 57 as a "new" note. Secondly, in position 9 appears an "erroneously" extracted note of pitch 66 which differs from the expected note of pitch 78 by an octave. This might be caused by the harmonics of the still sounding note of pitch 54 at position 7 and the new note of pitch 78 at position 9. Actually, these "octave errors" are typical for the extraction algorithm. To tackle this problem one can restrict oneself to only considering pitches which are reduced modulo 12 when using the note parameters as input for the synchronization algorithm. In spite of this reduction one is still left with sufficient information for a successful synchronization.

Table 2 shows the part of the cost matrix $C = (c_{ij})$ corresponding to the second movement using the parameter vector $\pi = (1, 1, 1, 1, 22, 50)$ and a modulo 12 reduction of the pitches.

From C one can compute the global SP-match μ. For the second measure this gives the matches $\{(5, 7), (6, 8), (7, 10), (8, 12)\}$ which are also printed in bold face in the above table. Note that the SP-algorithm has matched for both appoggiaturas of the score S the corresponding fuzzy notes at position $i = 6$ and

	6	7	8	9	10	11	12
4	114.8476	114.8476	114.8476	114.8476	114.8476	114.8476	114.8476
5	114.8476	**111.8700**	111.8700	111.8700	111.8700	111.8700	111.8700
6	114.8476	111.8700	**110.8132**	110.8132	110.8132	110.8132	110.8132
7	114.8476	111.8700	110.8132	110.8132	**107.4704**	107.4704	107.4704
8	114.8476	111.8700	110.8132	110.8132	107.4704	107.4704	**106.7956**

Table 2. Excerpt of the cost matrix $C = (c_{ij})$ corresponding to the second movement of the aria using the parameter vector $\pi = (1, 1, 1, 1, 22, 50)$ and a modulo 12 reduction of the pitches.

$i = 7$ respectively with the corresponding first notes of the appoggiaturas of P_Δ at position $j = 8$ and $j = 10$ respectively.

6 Experiments

We have implemented prototypes of the extraction algorithms described in Section 3 and the synchronization algorithms using MATLAB. Our algorithms for SM-, SP-, and MP-synchronization have been evaluated on a variety of classical polyphonic piano pieces of different complexity and length (ranging from 10 to 60 seconds) played on various instruments. For handling MIDI files within the MATLAB environment, we have implemented several tools for importing, exporting and visualizing MIDI data. Those tools have been made available for the research community, see [8]. 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 situations, 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 most applications mentioned in the introduction. Even more, in case of rather accurately extracted note parameters our synchronization algorithms could resolve subtle local time variations in an interpreted version of the piano piece.

As has already been observed in previous work, the evaluation of synchronization methods is not straightforward and requires special care. First, one has to specify the granularity of the alignment which very much depends on the application in mind. For example, if one is interested in a system which simultaneously highlights the current measure of the score while playing a corresponding interpretation (as a reading aid for the listener), a deviation of the alignment of a note or even several notes might be tolerable. However, for musical studies or when used as training data for statistical methods a synchronization at note-level or even ADSR-level (Attack-Decay-Sustain-Release) might be required.

Intuitive objective measures of synchronization quality may be the percentage of note events which are correctly matched, the percentage of mismatched notes, or the deviation between the computed and optimal tempo curve. (The output of a synchronization algorithm may be regarded as a tempo deviation or *tempo curve* between the two input data streams.) However, such a measure will fail if the note events which should be aligned do not exactly correspond (such as for trills, arpeggios, or wrong notes). In such a case the measure might give a low grade (bad score) which is not due to the performance of the algorithm but due to the quality of the input streams. Here, one would rate a synchronization as "good" if the musically most important note events are aligned correctly. Unfortunately, to determine such musically important note events a manual interaction is required, which makes the procedure unfeasible for large-scale examinations. Similarly, the measurement of tempo curves requires some ground truth about the desired outcome of the synchronization procedure. Moreover, if a synchronization algorithm is intended to handle even changes in the global structure of the musical piece such as an additional chorus or a missing bridge [18], modified notions of tempo curves have to be considered.

For those reasons, we decided to evaluate our synchronization results mainly via sonification, which allows an acoustic assessment on the ADSR-level. The design of suitable objective measures, which allow a systematic and automatic assessment of the synchronization results, is still an open research problem.

In the following we describe one of our experiments where we started with an uninterpreted score-like MIDI-version and an interpreted PCM-version of a given piano piece. Using the results of our MP-synchronization, we automatically modified the onset times of the MIDI-data stream to correspond to the PCM-data stream with regard to the global tempo and the local tempo variations. This results in an "expressive" MIDI-version which represents a sonification of our synchronization results. In case of good extraction parameters the modified MIDI-version rhythmically sounds like a real interpretation of the underlying piano piece.

To demonstrate our synchronization results we made some of our material available at `http://www-mmdb.iai.uni-bonn.de/download/syncDemo/`, where we provide the underlying test material as well as synchronized versions for several excerpts of classical piano pieces. For each example we supply the following data:

(1) An uninterpreted MIDI-version (representing the score of the underlying piece of music).
(2) An interpretation the score (1) performed on an acoustic grand piano (WAV format).
(3) A MIDI-file containing the score parameters (onset times and pitches) extracted from (2) using our extraction algorithm.
(4) An expressive MIDI-version created from (1) by modifying the onset times according to our synchronization result. (The note lengths are adopted from (1). Furthermore, the onset times of non-matched note events of (1) where modified via a simple interpolation.)

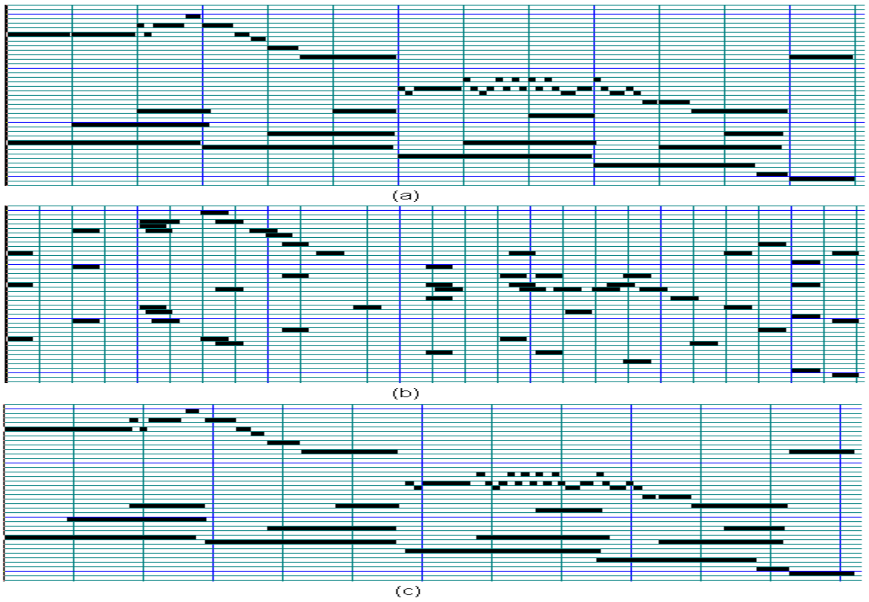
(5) An audio-file (WAV format) containing a mix of the the original interpretation (2) in the right audio channel and a synthesized version of the expressive MIDI (4) in the left audio channel.

In some of the cases, two different synchronized versions are given, which have been derived using slightly different choices of the parameter vector. As can be observed by listening to the audio-files (5), the synchronization results for the Mozart and the two Bach variations are very accurate. In the Burgmüller example, our player deliberately built in excessive local tempo deviations. Even in this case our algorithm computed a good global alignment with occasional local inaccuracies. The Bach Aria is synchronized very well except for the trill. This can be explained as follows: The number of notes in the trill of MIDI-version (1) is larger than the one of the corresponding trill in the interpreted PCM-version (2). In other words, there is no exact correspondence of note events in the data streams to be aligned (just as in the case where one has wrong notes in the interpretation). In this case, all what one can expect is not an alignment on the note-level but a rough alignment of the whole regions corresponding to the trills. This is actually what our algorithm does: the notes before and after the trill are matched correctly and the additional notes within the trill of the MIDI-version are not matched at all — only for the sonification we determined the onset times of the additional notes by interpolation which does not reflect the actual synchronization result. This also shows that the method of sonification has to be treated with care when evaluating synchronization results.

The last example is also illustrated by Fig. 5 corresponding to the first four measures of the of the Bach Aria BWV 988. The top part of the figure shows the piano roll representation of the score-like MIDI-version, the medium part shows the piano roll representation of the extracted note parameters from our interpreted PCM-version (note lengths are not considered), and the bottom part shows the piano roll representation of the "expressive" MIDI-version representing the synchronization result (note lengths of the score-like MIDI-version are used). As one can see in the medium part, the extraction algorithm also generates erroneous score-parameters such as typical octave errors coming from the harmonics. The quality of the extracted note parameters may also be low in parts consisting of many short notes. Furthermore, the uninterpreted MIDI-data stream and the extracted note parameters do note match well around the trill passage. Nevertheless, using the concept of partial matches we do not attempt to force alignment of non matching note objects. The bottom part of Fig. 5 shows that in spite of the minor extraction quality, this strategy allows us to find a proper MP-synchronization.

7 Conclusions

In this paper we have proposed algorithms for the automatic synchronization of different versions of a polyphonic piano piece given in different data formats (score, MIDI, PCM). Our implementation yields good synchronization results



(a)

(b)

(c)

Fig. 5. (a) Uninterpreted MIDI of the first four measures of the Bach Aria BWV 988, (b) MIDI representation of the extracted data from an interpreted version, and (c) expressive MIDI version representing the 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. The parameter vector π allows to weight different aspects in the matching process and leaves room for experiments. In contrast to related approaches [17,18], our method fully works on the symbolic, i.e., note-level domain. Working with such score-like features not only accounts for high accuracy — even when having strong local time deviations — but also makes DWT-like computation feasible due the relatively small data sets used in the actual synchronization. A further qualitative advantage of our matching-strategy is the focus on *partial matches*. Here we only include promising score-based matches in the final synchronization result rather than forcing a match for each note object in each of the two versions to be aligned.

Our system works off-line, where the computational 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 so far). An ongoing research project is to exploit the score information already in the extraction step. (See also [16] 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 possible lower quality of the extraction parameters. However, even lower quality of the parameters may be sufficient for a successful synchronization when using a suitably designed cost function which is robust under erroneous parameters.

Future work will also be concerned with properly defining various different types of synchronization problems some of which have been sketched in the last section. Based on such clearly defined problems, suitable measures for comparing synchronization results and assessing their overall quality have to be devised.

Automatic music processing is extremely difficult due to the complexity and diversity of music data. One generally has to account for various aspects such as the data format (e.g., score, MIDI, PCM), the genre (e.g., pop music, classical music, jazz), the instrumentation (e.g., orchestra, piano, drums, voice), and many other parameters (e.g., dynamics, tempo, or timbre). Therefore for most problems in computational musicology there are no universal algorithms yielding optimal solutions for all kinds of music. The approach by Turetsky et al. [18] works for general instrumentations but has weaknesses concerning large local tempo variations, whereas our approach captures even large local time deviations of a specific interpretation but is limited concerning the instrumentation. The approach of Soulez et al. [17] may be classified in between the former two. For the future it seems to be promising to build up a system which incorporates different, competing strategies (instead of relying on one single strategy) in combination with statistical methods as well as explicit instrument models in order to cope with the richness and variety of music.

References

1. Arifi, V.: *Algorithmen zur Synchronisation von Musikdaten im Paritur-, MIDI- und PCM-Format.* PhD thesis, Universität Bonn, Institut für Informatik, 2002. http://hss.ulb.uni-bonn.de:90/ulb_bonn/diss_online/math_nat_fak/2002/arifi_vlora
2. Bobrek, M., Koch, D.: *Music Signal Segmentation Using Tree-Structured Filter Banks.* Journal of Audio Engineering Society, Vol. 46, No. 5, May 1998.
3. Dannenberg, R. B.: *An On-Line Algorithm for Real-Time Accompaniment.* Proceedings of ICMC, 1984.
4. Dannenberg, R. B., Mukaino, H.: *New Techniques for Enhanced Quality of Computer Accompaniment.* Proceedings of ICMC, 1988.
5. Desain, P., Honing, H., Heijink, H.: *Robust Score-Performance Matching: Taking Advantage of Structural Information.* ICMC Proceedings 1997, pp. 377-340.
6. Foote, J.: *Automatic Audio Segmentation Using a Measure of Audio Novelity.* Proceedings of IEEE International Conference on Multimedia and Expo, vol. I, 2000, pp. 452-455.
7. 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, Spring 1982.
8. Kurth, F.: *A Matlab Toolbox for Handling Standard MIDI files.* `http://www-mmdb.iai.uni-bonn.de/download/miditools/`, Department of Computer Science III, Bonn University, Germany, 2003.
9. Large, E. W.: *Dynamic programming for the analysis of serial behaviours.* Behaviour Research Methods, Instruments, & Computers. 1993.

10. Klapuri, A., Virtanen, T., Holm, J.: *Robust Multipitch Estimation for the Analysis and Manipulation of Polyphonic Musical Signals.* In Proc. COST-G6 Conference of Digital Audio Effects, DAFx-00, Verona, Italy, 2000.
11. Mazzola, G.: *The Topos of Music.* Birkhäuser Verlag, 2002.
12. Orio, N., Lemouton, S., Schwarz, D.: *Score Following: State of the Art and New Developments*, Proc. of the Conference of New Interfaces for Musical Expression NIME, Montreal, 2003.
13. Rabiner, L.R., Juang, B.-H.: *Fundamentals of Speech Recognition,* Englewood Cliffs, Prentice Hall, 1993.
14. Raphael, C.: *Automatic Segmentation of Acoustic Musical Signals Using Hidden Markov Models.* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 21, No. 4, April 1999.
15. Raphael, C.: *Automatic Transcription of Piano Music.* ISMIR 2002, IRCAM, Paris, Conference Proceedings, 2002.
16. Scheirer, E. D.: *Extracting Expressive Performance Information from Recorded Music.* Unpublished M. S. thesis, MIT Media Laboratory, 1995 http://web.media.mit.edu/ eds/thesis.pdf
17. Soulez, F., Rodet, X., Schwarz, D: *Improving polyphonic and poly-instrumental music to score alignment.* International Conference on Music Information Retrieval, Baltimore, 2003.
18. Turetsky, R. J., Ellis, D. P., *Force-Aligning MIDI Syntheses for Polyphonic Music Transcription Generation.* International Conference on Music Information Retrieval, Baltimore, USA, 2003.