Automatic Synchronization of Music Data in Score-, MIDI- and PCM-Format

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. Since the waveform-based PCM-data streams do not contain any information on the notes we extract in a preprocessing step note parameters such as onset times and pitches in order to make the PCM-data comparable to symbolic score-like representations. In the extraction step we use novelity curves for onset detection and filter bank tree techniques in combination with note templates for pitch extraction. To handle ambiguities such as trills or arpeggios in the score data-stream we introduce the concept of fuzzy-notes. Further suitable normalization and quantization of the involved data streams are necessary to generate the input data of our synchronization algorithms which are based on the technique of dynamic programming. The decisive ingredient for our approach are carefully designed cost functions which will be explained in detail. Our synchronization algorithms have been tested on a variety of classical polyphonic piano pieces recorded by MIDI- and standard acoustic pianos or taken from CD-recordings.

Keywords: score to PCM-audio synchronization, note-extraction, polyphonic piano music

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 Capella or Score format, 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 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 future application is the automatic annotation of a piece of music in different data formats as 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 a reading aid of scores.

In this paper we concentrate on three representative data formats used for 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 some 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. We have developed synchronization algorithms for two data streams in any of these three data formats which will be referred to as Score-to-MIDI (SM) synchronization, Score-to-PCM (SP) synchronization, and MIDI-to-PCM (MP) synchronization.

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 from the PCM-recording information such as note onsets and pitches in order to make it comparable to other symbolic score-like representations and hence algorithmically useable 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 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 some score-like format of the original piece of music. However, the extracted data — even from very complex piano pieces — is good enough to ensure success in view of the synchronization problem.

The rest of this paper is organized as follows. After some review of related problems and links to the relevant literature in Section 2, we summarize in Section 3 our system for the extraction of musically relevant parameters from PCM-data streams. The actual synchronization algorithms are described in Section 4. Crucial for the algorithms is further preprocessing of the Score-, MIDI-, and extracted data used as the input data of the matching algorithm, which is based on carefully designed cost functions and which uses the technique of dynamic programming. Finally, Section 5 contains some examples and a summary of our experiments and Section 6 closes with concluding remarks.

2 Background and Related Work

There are a number of synchronization problems in music which have been the focus of various research activities, but which differ to a greater or lesser extent to the one discussed in this paper. In this section, we give references to some papers which address related problems and which, in part, apply similar approaches for their solution. Due to space limitation we are only able to give a small choice of the relevant literature.

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 [12] has developed a system for automatic musical accompaniment of an oboe based on Hidden Markov Models. Desain et. al. describe in [5] some 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 where there are no such explicit and clean parameters.

The extraction itself of such score-like note parameters from waveform-based PCM-data constitutes an active research area with many unsolved problems. As an example, we mention the approach of Raphael [13] 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 [7] uses the concept of the so-called novelity 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 1 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.

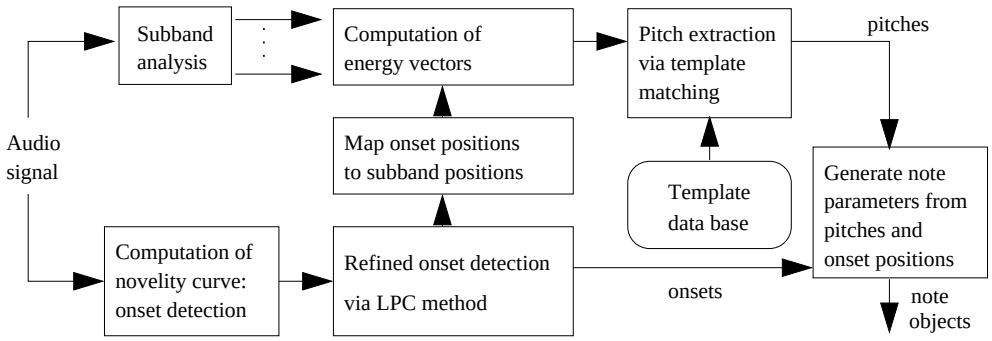

Figure 1: Diagram of the feature extraction algorithm.

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.

We now briefly discuss the components of our algorithm. For onset detection, we first calculate a novelity curve (similarly to [6]) of the input signal. This step basically consists of a short time Fourier transform with a step size of 23 ms, where for each step a novelity 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 [8] 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.

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.

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 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. 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. 2) are given by $(0, 79)$ and $(2.75, 83)$ respectively.

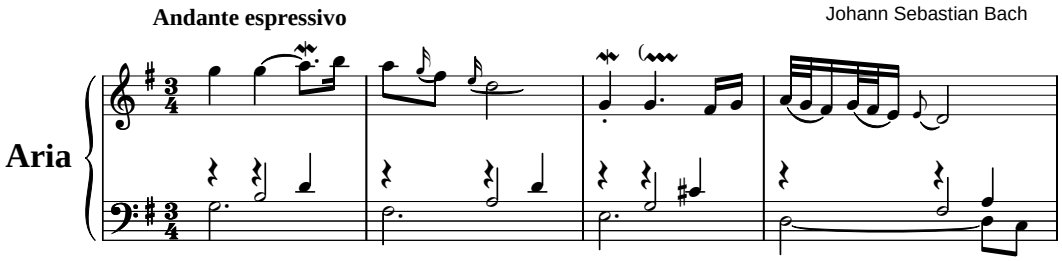

Figure 2: First four measures of the *Aria con Variazioni* by J. S. Bach, BWV 988.

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) consisting of some onset time $e \in \mathbb{Q}$ and some 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 note and the set of all pitches appearing in a possible realization of this object (see Fig. 3 for an illustration). For example, the third note object of the first measure in the right hand of the Aria (Fig. 2) is given by $(3, \{79, 81\})$.

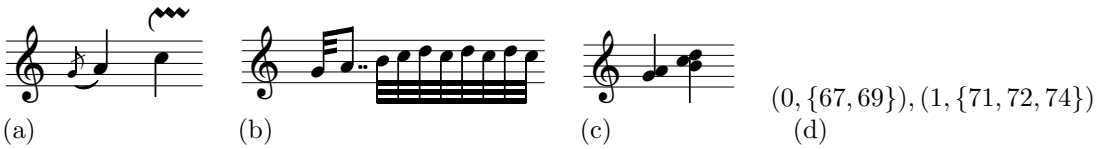

Figure 3: Appoggiatura and trill, (a) notation, (b) possible realization, (c) fuzzy note, (d) encoding.

Using this encoding 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 $H_0 \subset 2^{[0:127]}$ 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 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

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

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, $S_{0i}, S_{1i}, P_{0j} \subset [0:127]$ 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 this, 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. With respect to the parameter vector $\pi := (\alpha, \beta, \gamma, \delta, \zeta, \Delta) \in \mathbb{R}^6_{\geq 0}$ the *SP-cost* of an SP-match μ between some score S and some Δ-quantized set P_Δ of the corresponding PCM-document is defined as

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

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 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 $\mu' := \mu \setminus \{(i,j)\}$ 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, the following procedure 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
```

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 Examples and Experiments

We illustrate the SP-synchronization by means of the example from Fig. 2. The score S represents the first four measures of the Aria. The PCM-version P represents a recording of the same measures performed on a Steinway grand piano. The physical length is $\ell(P) = 13$ sec. The quantization constant is set to $\Delta = 50$ 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. The following table shows the note objects of the score S and the Δ-quantized extracted PCM-data P_Δ.

S				P_Δ		
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}

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. The following table 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:

	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**

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

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 [1].

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. The parameter vector π allows to weight different aspects in the matching process and leaves room for experiments.

Our 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 [14] 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

[1] self-citation

[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.: *ARTHUR: Retrieving Orchestral Music by Long-Term Structure.* Proceedings of the International Symposium on Music Information Retrieval, Plymouth, Massachusetts, October 2000.

[7] 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.

[8] 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.

[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] 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.

[13] Raphael, C.: *Automatic Transcription of Piano Music.* ISMIR 2002, IRCAM, Paris, Conference Proceedings, 2002.

[14] 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