AN AUDIO TO SCORE ALIGNMENT FRAMEWORK USING SPECTRAL FACTORIZATION AND DYNAMIC TIME WARPING

J.J. Carabias-Orti[1] **F.J. Rodriguez-Serrano**[2] **P. Vera-Candeas**[2]
N. Ruiz-Reyes[2] **F.J. Cañadas-Quesada**[2]
[1] Music Technology Group (MTG), Universitat Pompeu Fabra, Spain
[2] Polytechnical School of Linares, Universidad de Jaen, Spain
julio.carabias@upf.edu

ABSTRACT

In this paper, we present an audio to score alignment framework based on spectral factorization and online Dynamic Time Warping (DTW). The proposed framework has two separated stages: preprocessing and alignment. In the first stage, we use Non-negative Matrix Factorization (NMF) to learn spectral patterns (i.e. basis functions) associated to each combination of concurrent notes in the score. In the second stage, a low latency signal decomposition method with fixed spectral patterns per combination of notes is used over the magnitude spectrogram of the input signal resulting in a divergence matrix that can be interpreted as the cost of the matching for each combination of notes at each frame. Finally, a Dynamic Time Warping (DTW) approach has been used to find the path with the minimum cost and then determine the relation between the performance and the musical score times. Our framework have been evaluated using a dataset of baroque-era pieces and compared to other systems, yielding solid results and performance.

1. INTRODUCTION

In this work, we address the problem of audio-to-score alignment (or score matching), which is the task of synchronizing an audio recording of a musical piece with the corresponding symbolic score. There are two approaches to this problem, often called "offline" and "online" alignment. In offline alignment, the whole performance is accessible for the alignment process, i.e. it allows us to "look into the future" while establishing the matching. This is interesting for applications that do not require the real-time property such as Query-by-Humming, intelligent audio editors and as a front-end for many music information retrieval (MIR) systems. Online alignment, also known as score following, processes the data in realtime as the signal is acquired. This tracking is very useful for applications such as automatic page turning, automated computer accompaniment of a live soloist, synchronization of live sound processing algorithms for instrumental electroacoustic composition or the control of visual effects synchronized with the music (e.g. stage lights or opera supertitles).

Audio-to-score alignment is traditionally performed in two steps: feature extraction and alignment. On the one hand, the features extracted from the audio signal characterize some specific information about the musical content. Different representations of the audio frame have been used such as the output of a short-time Fourier transform (STFT) [1], auditory filter bank responses [2], chroma or "chroma-like" vectors [3, 4], multi-pitch analysis information [5–8]. On the other hand, the alignment is performed by finding the best match between the feature sequence and the score. In fact, most systems rely on cost measures between events in the score and in the performance. Two methods well known in speech recognition have been extensively used in the literature: statistical approaches (e.g. HMMs) [6–11], and dynamic time warping (DTW) [3, 12, 13].

In this paper we propose an audio to score framework based on two stages: preprocessing and alignment. On the first stage, we analyze the provided MIDI score to define the set of combinations of concurrent notes and the transitions between them (i.e. the different states of the provided MIDI). Then the score is converted into a reference audio signal using a synthesizer software and we use a method based on Non-Negative Matrix Factorization (NMF) with Beta-divergence to learn spectral patterns (i.e. basis functions) for each combination of notes. A similar approach was used by Fritsch and Plumbey in [14], but they use one component per instrument and note plus some extra-components to model the residual sounds. NMF was also used by Cont [8] as a multi-pitch estimator which defines the observation model. Joder et al. [10] also defined a set of template vectors for each combination of concurrent notes but directly from the score (i.e. without using audio synthesis). The combination templates are obtained as a linear mapping of individual notes trained patterns using several representations. On the second stage, alignment is performed in two steps. First, the matching measure between events in the score and in the performance is defined.

Concretely, a divergence (i.e. cost) matrix is estimated using a low latency signal decomposition method previously developed by the authors in [15] that uses the spectral patterns fixed from the previous stage. Finally a DTW strategy has been used to find the path with the minimum cost and then determine the relation between the performance and the musical score times. Both, offline and online DTW approaches are implemented as in [23] and [13], respectively.

The structure of the rest of the paper is as follows. In Section 2, we briefly review the DTW principles. In Sections 3 the proposed audio to score framework is explained. In Section 4, the evaluation set-up is presented and, in Section 5 the proposed method has been tested and compared with other reference systems. Finally, we summarize the work and discuss future perspectives in Section 6.

2. DTW BACKGROUND

DTW is a technique for aligning two time series or sequences. The series are represented by 2 vectors of features $U = u_1, ..., u_i, ...u_I$ and $V = v_1, ..., v_j, ..., v_J$ where i and j are the point indices in the time series. I and J represent the length of time series U and V, respectively. As a dynamic programming technique, it divides the problem into several sub-problems, each of which contribute in calculating the distance (or cost function) cumulatively.

The first stage in the DTW algorithm is to fill a local distance matrix (a.k.a cost matrix) $\mathbf{D}$ as follows:

$$D(i,j) = \psi(u_i, v_j) \tag{1}$$

where matrix $\mathbf{D}$ has IxJ elements which represent the match cost between every two points in the time series. The cost function ψ could be any cost function that returns cost 0 for a perfect match, and a positive value otherwise (e.g. euclidean distance).

In the second stage (forward step), a warping matrix $\mathbf{C}$ is filled recursively as:

$$C(i,j) = \min \left\{ \begin{array}{c} C(i, j-c_j) + D(i,j) \\ C(i-c_i, j) + D(i,j) \\ C(i-c_i, j-c_j) + \sigma D(i,j) \end{array} \right\} \tag{2}$$

where c_i and c_j are step size at each dimension and range from 1 to α_i and 1 to α_j, respectively. α_i and α_j are the maximum step size at each dimension. Parameter σ controls the bias toward diagonal steps. $C(i,j)$ is the cost of the minimum cost path from $(1,1)$ to (i,j), and $C(1,1) = D(1,1)$.

Finally, in the last stage (traceback step), the minimum cost path $\mathbf{w} = w_1, ..., w_k, ..., w_K$ is obtained by tracing the recursion backwards from $C(I,J)$. Each w_k is an ordered pair (i_k, j_k) such that $(i,j) \in \mathbf{w}$ means that the points u_i and v_j are aligned. Moreover, the path has to satisfy the following three conditions: i) $\mathbf{w}$ is bounded by the ends of both sequences, ii) $\mathbf{w}$ is monotonic and iii) $\mathbf{w}$ is continuous.

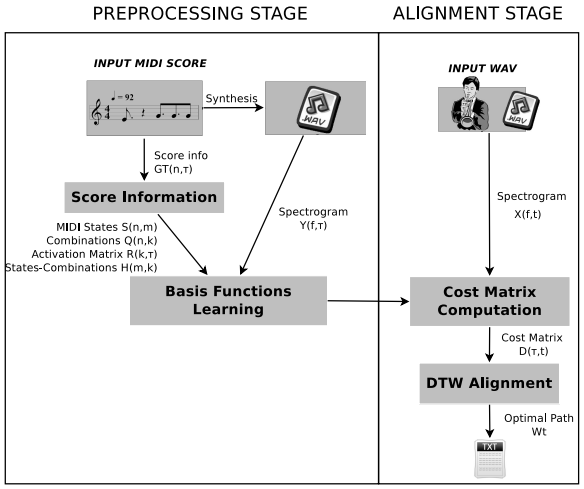


Figure 1: *Block diagram of the proposed system*

3. SYSTEM OVERVIEW

The proposed framework for audio-to-score alignment is presented in Figure 1. As can be seen, the framework has two stages. First, the preprocessing stage must be computed beforehand and only the MIDI score is required. Then, once the parameters are learned, alignment can be computed in realtime.

3.1 Preprocessing Stage

In this stage, the parameters for the alignment are learned from the score, which must be provided beforehand using MIDI representation. This stage is performed in two successive steps: states definition and spectral patterns learning, as detailed below.

3.1.1 States Definition

The aim of this step is to adequately organize the information given by the score to be used for alignment purposes.

First of all, the binary ground-truth transcription matrix $\mathbf{GT}(n,\tau)$ (see Figure 2(a)) is inferred from the MIDI score, where τ is the time in frames referenced to the score (MIDI time) and n are the notes in MIDI scale. In Figure 2(a) the MIDI score involves just one instrument (a piano) but more instruments can be defined in a score. For those cases the n index refers to each note of the different instruments. Consequently, the number of total notes for a composition, N, is obtained as the sum of the number of different notes per instrument. The score can be interpreted as a consecutive sequence of M states. Each state m is defined by its combination of concurrent notes in the score (for all instruments). Also, the score informs about the time changes from one state to the next state. In fact, a score follower must determine the time (referenced to the input signal) of all transitions between states. There are only K unique combination of notes in a score where $K \leq M$ because some states represent the same combination of notes.

From the ground-truth transcription matrix $\mathbf{GT}(n,\tau)$,

we obtain the following decomposition of binary matrixes

$$\mathbf{GT}(n,\tau) = \mathbf{Q}(n,k)\mathbf{R}(k,\tau) \tag{3}$$

where $\mathbf{Q}(n,k)$ is the notes-to-combination matrix, k the index of each unique combination of notes and $\mathbf{R}(k,\tau)$ represents the activation of each combination in MIDI time. In Figure 2(b), the note-to-combination matrix $\mathbf{Q}(n,k)$ is represented. This matrix contains the notes belonging to each combination but no information about MIDI time. Conversely, $\mathbf{R}(k,\tau)$ matrix retains the MIDI time activation per combination but no information about the notes active per combination, as can be seen in Figure 2(d).

In order to obtain the information for states required to perform the alignment, the notes-to-combination matrix $\mathbf{Q}(n,k)$ is further decomposed as

$$\mathbf{Q}(n,k) = \mathbf{S}(n,m)\mathbf{H}(m,k) \tag{4}$$

where $\mathbf{S}(n,m)$ is the notes-to-state matrix, m the index for the states, M the number of states and $\mathbf{H}(m,k)$ represents the unique combination k of notes active at each state m. In Figure 2(c), the notes-to-state matrix $\mathbf{S}(n,m)$ is represented, this matrix contains the notes belonging to each state, while $\mathbf{H}(m,k)$ matrix informs about the combinations active at each state, as can be seen in Figure 2(e).

The matrixes here defined will be used in the next stages to perform the alignment and are computed from the MIDI score.

3.1.2 Spectral Patterns Learning

When a signal frame is given to a score follower, the first step should be the computation of a similarity measure between the current frame and the different combinations of notes defined by the score. Our approach is to compute a distance (or divergence) between the frequency transform of the input and just one spectral pattern per combination of notes. A spectral pattern is here defined as a fixed spectrum which is learned from a signal with certain characteristics. The use of only one spectral pattern per combination allows us to compute the divergence with a low complexity signal decomposition method. This means that our method must learn in advance the spectral pattern associated to each unique combination of notes for the score. To this end, a state-of-the-art supervised method based on Non-Negative Matrix Factorizacion (NMF) with Beta-divergence and Multiplicative Update (MU) rules [15] is used, but in this work, we propose to apply it on synthetic signal generated from the MIDI score [1] instead of the real audio performance.

First of all, let us define the signal model as

$$\mathbf{Y}(f,\tau) \approx \hat{\mathbf{Y}}(f,\tau) = \mathbf{B}(f,k)\mathbf{G}(k,\tau) \tag{5}$$

where $\mathbf{Y}(f,\tau)$ is the magnitude spectrogram of the synthetic signal, $\hat{\mathbf{Y}}(f,\tau)$ is the estimated spectrogram, $\mathbf{G}(k,\tau)$ matrix represents the gain of the spectral pattern for combination k at frame τ, and $\mathbf{B}(f,k)$ matrix, for $k = 1,...,K$, represents the spectral patterns for all the combinations of notes defined in the score.

When the parameters are restricted to be non-negative, as it is the case of magnitude spectra, a common way to compute the factorization is to minimize the reconstruction error between the observed spectrogram and the modeled one.

The most popular cost functions are the Euclidean (EUC) distance, the generalized Kullback-Leibler (KL) and the Itakura-Saito (IS) divergences.

Besides, the Beta-divergence (see eq. 6) is another commonly used cost function that includes in its definition the three previously mentioned EUC ($\beta = 2$), KL ($\beta = 1$) and IS ($\beta = 0$) cost functions.

$$D_\beta(x|\hat{x}) = \begin{cases} x\log\frac{x}{\hat{x}} - x + \hat{x} & \beta = 1 \\ \frac{x}{\hat{x}} + \log\frac{x}{\hat{x}} - 1 & \beta = 0 \\ \frac{1}{\beta(\beta-1)}\left(x^\beta + (\beta-1)\hat{x}^\beta - \beta x\hat{x}^{\beta-1}\right) & otherwise, \end{cases} \tag{6}$$

In order to obtain the model parameters that minimize the cost function, Lee *et al.* [18] proposes an iterative algorithm based on MU rules. Under these rules, $D_\beta(\mathbf{Y}(f,\tau)|\hat{\mathbf{Y}}(f,\tau))$ is shown to be non-increasing at each iteration while ensuring non-negativity of the bases and the gains. Details are omitted to keep the presentation compact, for further information please read [18, 19]. For the model of eq. (5), multiplicative updates which minimize the Beta-divergence are defined as

$$\mathbf{B}(f,k) \leftarrow \mathbf{B}(f,k) \odot \frac{\left(\mathbf{Y}(f,\tau) \odot \hat{\mathbf{Y}}^{\beta-2}(f,\tau)\right)\mathbf{G}^T(\tau,k)}{\left(\hat{\mathbf{Y}}^{\beta-1}(f,\tau)\right)\mathbf{G}^T(\tau,k)} \tag{7}$$

$$\mathbf{G}(k,\tau) \leftarrow \mathbf{G}(k,\tau) \odot \frac{\mathbf{B}(f,k)\left(\mathbf{Y}(f,\tau) \odot \hat{\mathbf{Y}}^{\beta-2}(f,\tau)\right)}{\mathbf{B}(f,k)\left(\hat{\mathbf{Y}}^{\beta-1}(f,\tau)\right)} \tag{8}$$

where operator $\odot$ indicates Hadamard product (or element-wise multiplication), division and power are also element-wise operators and $(\cdot)^T$ denotes matrix transposition.

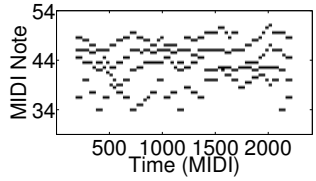
Finally, the method to learn the spectral patterns for each state is described in Algorithm 1.

Algorithm 1 Method for learning spectral patterns combinations

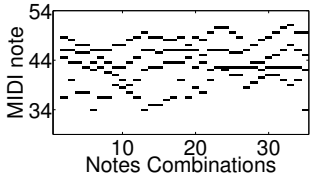
1. Initialize $\mathbf{G}(k,\tau)$ as the combinations activation matrix $\mathbf{R}(k,\tau)$ and $\mathbf{B}(f,k)$ with random positive values.
2. Update the bases using eq. (7).
3. Update the gains using eq. (8).
4. Normalize each spectral pattern of $\mathbf{B}(f,k)$ to the unit β-norm.
5. Repeat step 2 until the algorithm converges (or maximum number of iterations is reached).

As explained in Section 3.1.1, $\mathbf{R}(k,\tau)$ is a binary combination/time matrix that represents the activation of combination k at frame τ of the training data. Therefore, at each frame, the active combination k is set to one and the rest are zero. Gains initialized to zero will remain zero, and therefore the frame becomes represented with the correct combination.

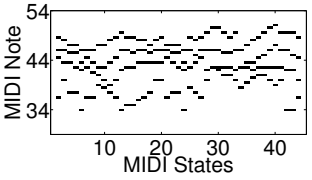
[1] MIDI synthetic signals are generated using Timidity++ with the FluidR3 GM soundfont on Mac OS



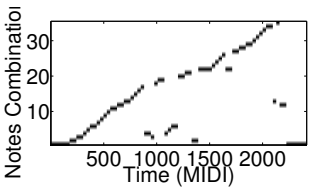
(a) MIDI Ground-Truth Transcription $\mathbf{GT}(n, \tau)$



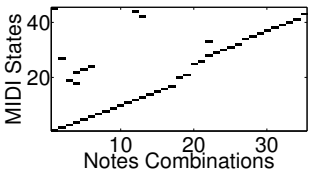
(b) Notes-to-combination matrix $\mathbf{Q}(n, k)$



(c) Notes-to-state matrix $\mathbf{S}(n, m)$



(d) Combinations activ. matrix $\mathbf{R}(k, \tau)$



(e) States-to-combination matrix $\mathbf{H}(m, k)$

Figure 2: *Music signal from the test database in Section 4 ("01-AchGottundHerr"). (a) MIDI Ground-Truth Transcription* $\mathbf{GT}(n, \tau)$. *(b) Notes-to-combination matrix* $\mathbf{Q}(n, k)$. *(c) Notes-to-state matrix* $\mathbf{S}(n, m)$. *(d) Combinations activation matrix* $\mathbf{R}(k, \tau)$. *(e) States-to-combination matrix* $\mathbf{H}(m, k)$.

3.2 Alignment Stage

In this stage, the alignment between the score and the audio performance is accomplished in realtime using the information from the preprocessing stage.

3.2.1 Observation Model

As explained in Section 3.1.2, the spectral patterns $\mathbf{B}(f, k)$ for the K different combinations of notes are learned in advance using a MIDI synthesizer and kept fixed. Each spectral pattern models the spectrum of a unique combination.

Now, the aim is to compute the gain matrix $\mathbf{G}(k, t)$ and the cost matrix $\mathbf{D}(\tau, t)$ that measures the suitability of each combination of notes belonging to each MIDI time τ to be active at each frame t (referenced to the signal input) by analyzing the similarity between the spectral patterns $\mathbf{B}(f, k)$ and the input signal spectrogram [2] . From the cost matrix $\mathbf{D}(\tau, t)$, a classical DTW approach can be applied to compute the alignment path.

In this work, we propose to perform the factorization using the realtime single-pitch constrained method proposed in [15]. Although this method was designed to address music transcription of monophonic signals, it can be adapted for audio to score alignment of polyphonic signals because only one combination will be active at a time. In this transcription method, the optimum combination k_{opt} is chosen to minimize the Beta-divergence function at frame t under the assumption that only one gain is non-zero at each frame. Taking the combinations as the index of gains $\mathbf{G}(k, t)$, this assumption is fair because only a unique combination k of notes is active at each time (at least when producing the audio signal).

Thus, the signal model with the single-combination constraint for the signal input vector at time t, $\mathbf{x}_t(f)$, is defined as follows.

$$\mathbf{x}_t(f) \approx \hat{\mathbf{x}}_{k_{opt},t}(f) = g_{k_{opt},t}\mathbf{b}_{k_{opt}}(f) \tag{9}$$

[2] Note that we are using $\mathbf{X}$ and t instead of $\mathbf{Y}$ and τ to represent the signal magnitude spectrogram and the time frames to distinguish between real world and synthetic signals.

where $\hat{\mathbf{x}}_{k_{opt},t}(f)$ is the modeled signal for the optimum combination k_{opt} at frame t.

$$k_{opt}(t) = \arg\min_{k=1,\ldots,K} D_\beta\left(\mathbf{x}_t(f)|g_{k,t}\mathbf{b}_k(f)\right) \tag{10}$$

The signal model in eq. (9) assumes that when combination k is active all other combinations are inactive and, therefore, the gain $g_{k,t}$ is just a scalar and represents the gain of the k combination. The model of eq. (10) allows the gains to be directly computed from the input data $\mathbf{X}(f, t)$ and the trained spectral patterns $\mathbf{B}(f, k)$ without the need of an iterative algorithm and thus, reducing the computational requirements. To obtain the optimum combination at each frame, we must first compute the divergence obtained by the projection of each combination at each frame and then select the combination that achieves the minimum divergence as the optimum combination at each frame.

In the case of Beta-divergence, the cost function for combination k and frame t can be formulated as

$$D_\beta(\mathbf{x}_t(f)|g_{k,t}\mathbf{b}_k(f)) = \sum_f \frac{1}{\beta(\beta-1)}(\mathbf{x}_t^\beta(f) + (\beta-1)(g_{k,t}\mathbf{b}_k(f))^\beta - \beta\mathbf{x}_t(f)(g_{k,t}\mathbf{b}_k(f))^{\beta-1}) \tag{11}$$

The value of the gain for combination k and frame t is then computed by minimizing eq. (11). Conveniently, this minimization has a unique non-zero solution due to the scalar nature of the gain for combination k and frame t (see more details in [15]).

$$g_{k,t} = \frac{\sum_f \mathbf{x}_t(f)\mathbf{b}_k(f)^{(\beta-1)}}{\sum_f \mathbf{b}_k(f)^\beta} \tag{12}$$

Finally, the divergence matrix for each combination at each frame is defined as:

$$\mathbf{\Phi}(k, t) = D_\beta(\mathbf{x}_t(f)|g_{k,t}\mathbf{b}_k(f)) \tag{13}$$

where β can take values in the range $\in [0, 2]$.

As can be inferred, the divergence matrix $\mathbf{\Phi}(k,t)$ provides us information about the similitude of each combination k spectral pattern with the real signal spectrum at each frame t. Using this information, we can directly compute the cost matrix between the MIDI time τ and the time of the input signal t as

$$\mathbf{D}(\tau,t) = \mathbf{R}^T(\tau,k)\mathbf{\Phi}(k,t) \tag{14}$$

where $\mathbf{R}(k,\tau)$ is the combinations activation matrix defined in Section 3.1.1 and superscript "T" stands for matrix transposition. The process is detailed in Algorithm 2.

Algorithm 2 Divergence matrix computation method

```
Initialize B(f,k) with the values learned in Section 3.1.2.
for t=1 to T do
    for k=1 to K do
        Compute the gains g_{k,t} using eq. (12).
        Compute the current value the divergence matrix
        Φ(k,t) using eq. (13).
    end for
end for
Compute the cost matrix D(τ,t) between MIDI time and input signal time using (14).
```

To resume, we propose the use the divergence matrix $\mathbf{D}(\tau,t)$ as the input of the DTW algorithm in order to perform the alignment.

3.2.2 Path Computation

We here propose to use a DTW based method to perform the alignment using the cost matrix $\mathbf{D}(\tau,t)$ obtained in eq. (14). This cost matrix is computed from the input signal $\mathbf{X}(f,t)$ and the "synthetic" spectral patterns per combination $\mathbf{B}(f,k)$ explained in Section 3.1.2. The term "synthetic" comes from the fact that the spectral patterns $\mathbf{B}(f,k)$ are computed from the score using a MIDI synthesizer.

a) Offline approach: This approach represents the classical offline alignment using DTW. To this end, we have used the code from [23]. The forward step is computed as in the classical DTW (see eq. (2)). In this experiment, c_i and c_j range from 1 to 4 in order to allow 4 times faster speed of interpretation. Finally the optimum path is obtained by tracing the recursion backwards from $\mathbf{C}(I,J)$ as in the original formulation of DTW (see Section 2).

b) Online approach: The online algorithm differs from an standard (i.e. offline) DTW algorithm in some points. Firstly, the signal is partially unknown (or the future of the signal is not known when making the alignment decisions), so the global path constraints cannot be directly implemented, in other words, the recursion backwards can not be traced from the last frame T of the signal. Secondly, if some latency (i.e. delay in the decision) is permitted, the recursion backwards can be traced in equally spaced frames of the input signal making the latency equal to the difference in time of the frame when the backtracking is done and the input signal frame. Finally, in order to run in realtime, the complete algorithm should not increase the complexity with the length of the signal.

In this work, we used the online scheme proposed by Dixon in [13]. In fact, Dixon's algorithm calculates an "adaptive diagonal" through the cost matrix by seeking the best path considering a searching band with a fixed width. Here, we propose an online algorithm with a fixed latency of just one frame. In order to obtain this low latency, no backtracking is allowed, taking the decision directly from the forward information at each frame t. As a consequence of the low latency of online algorithms (apart from the complexity reduction), the obtained results are degraded from their offline counterparts. In fact, for those situations in which a higher latency can be supported, delaying the decision in time using a limited traceback can improve the obtained results of the online algorithms.

4. EXPERIMENTAL SETUP

a) Time-Frequency representation: In this paper we use a low-level spectral representation of the audio data which is generated from a windowed FFT of the signal. A Hanning window with the size of 128 ms, and a hop size of 10 ms is used (for both synthetic and real-world signals). Here, we use the resolution of a single semitone as in [21]. In particular, we implement the time-frequency representation by integrating the STFT bins corresponding to the same semitone.

b) Evaluation metrics: We have used the same evaluation metrics as in the MIREX Score Following task. Detailed information can be found in [22]. For each piece, aligned rate (AR) or precision is defined as the proportion of correctly aligned notes in the score and ranges from 0 to 1. A note is said to be correctly aligned if its onset does not deviate more than a threshold (a.k.a tolerance window) from the reference alignment. Missed notes are events that are present in the reference but not reported. Recognized notes whose onsets are far from the given threshold are considered misaligned notes.

c) Dataset: The dataset used to evaluate our method is comprised of excerpts from 10 human played J.S. Bach four-part chorales. The audio files are sampled from real music performances recorded at 44.1 kHz that are 30 seconds in length per file. Each piece is performed by a quartet of instruments: violin, clarinet, tenor saxophone and bassoon. Each musician's part was recorded in isolation. Individual lines were then mixed to create 10 performances with four-part polyphony. Ground-truth alignment is provided for both, individual sources and mixture, the latter assuming constant tempo between annotated beats and a perfect synchronization between the musicians. More information about this dataset can be found in [6].

5. RESULTS

To analyze the performance of the proposed (offline and online) methods in Section 3. Evaluation has been per-

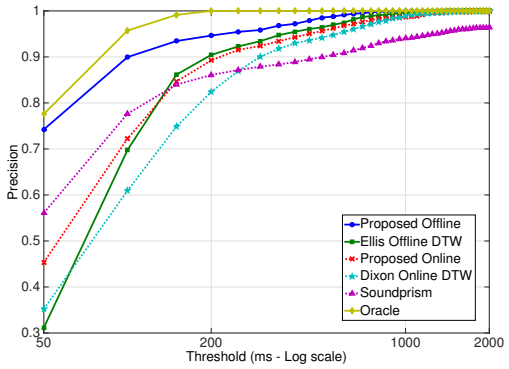


Figure 3: *Precision values in function of the tolerance window*

formed using the metrics detailed in Section 4. The proposed systems are compared with four reference methods that are detailed below: a) Ellis' Offline DTW [23] [24], b) Dixon's Online DTW [13], c) Soundprism [6] and d) Oracle. Note that the latter is not a score following system but the provided aligned MIDI score assuming constant tempo between annotated beats and perfect synchronization between musicians as explained in Section 4. The evaluation of this oracle information is very interesting to analyze the deviation of the different instrument performances between themselves and also to measure the best performance that can be obtained by score followers that are only capable of aligning on a global level (i.e., cannot detect the onset positions of individual notes). Note that there are several works that attempt to refine the alignment by synchronizing the onsets/offsets individually for each note in a post-processing stage [25–27].

In a first experiment, we have evaluated the precision of the analyzed methods as a function of the onset deviation threshold. To this end, the threshold value was varied from 50 to 2000 ms in 50 ms steps. Obtained results are plotted in Figure 3. As can be seen, the Oracle method, which can be considered as the upper bound for score followers performing global alignment, requires around 200 ms threshold to obtain a perfect alignment. This value comes from the difference between the ground-truth alignment for each instrument played in isolation and the global ground-truth of the whole mixture, obtained by interpolating the annotated beat times of each audio.

In general, our offline approach obtain the best results in terms of precision. In fact, our offline approach clearly outperforms Ellis' offline approach, mainly due to the factorization based feature extraction stage. Regarding the online methods, our online approach and Soundprism obtain similar results on average than the Ellis' offline approach and clearly outperform Dixon's online approach. Soundprism seems to perform better when using lower threshold values while our online approach allows convergence to the optimum alignment as the threshold is increased.

In a second experiment (see Table 1), we evaluate the performance of the proposed methods as a function of the polyphony. A fixed threshold (a.k.a tolerance window) of 200ms is used because, as illustrated in Figure 3, this value represents the difference between isolated instruments and

	Poly	Precision	Miss	Missalign	Av offset	Av |offset|	Std Offset
Prop. Offline	2	**94,59**	0,00	5,41	-11,27	**33,43**	44,38
	3	**94,75**	0,00	5,25	-11,78	**34,25**	44,89
	4	**94,50**	0,00	5,50	-11,09	**35,16**	46,51
Ellis Offline	2	90,57	0,00	9,42	66,90	71,28	47,30
	3	90,39	0,00	9,60	65,67	71,53	50,24
	4	89,80	0,00	10,19	66,97	72,62	49,93
Prop. Online	2	**88,44**	0,00	11,56	-41,18	57,78	56,40
	3	**90,13**	0,00	9,86	-42,96	60,56	57,65
	4	90,70	0,00	9,30	-44,15	62,97	58,58
Dixon Online	2	81,69	0,00	18,31	53,93	70,51	67,02
	3	83,17	0,00	16,83	53,88	70,65	67,02
	4	83,94	0,00	16,06	51,29	71,64	70,26
Soundprism	2	83,01	0,00	16,99	-25,90	**48,08**	55,36
	3	88,81	0,00	11,19	-21,23	**43,73**	51,54
	4	**93,50**	0,00	6,50	-22,05	**42,91**	49,13
Oracle	2	100,00	0,00	0,00	6,15	32,79	**42,47**
	3	100,00	0,00	0,00	6,09	32,67	**43,08**
	4	100,00	0,00	0,00	6,07	32,58	**43,38**

Table 1: Audio-to-score results as a function of polyphony in terms of piecewise precision (%). Offset values in ms. The bold percentage shows the best result for each measure.

mixture ground-truth alignment.

As explained in the previous section, offline methods perform better in general than the online ones. The proposed offline method obtains the best results among the compared methods and has demonstrated to be robust against polyphony in the analyzed dataset (polyphony 2 to 4). Regarding the online methods, our online approach and Soundprism obtain similar results on average and clearly outperforms Dixon's online approach, although the former seems to be more robust against the polyphony.

In relation to the offset, the oracle solution exhibits the minimum possible std offset due to the differences in starting times for the same states between musicians. Moreover, our offline approach and the online Soundprism have the lower average offset values which means that both methods are more responsive and thus provide better results when dealing with lower thresholds.

6. CONCLUSIONS

In this paper we present a score following framework based on spectral factorization and DTW. Spectral factorization is used to learn spectral patterns for each combination of concurrent notes in the MIDI score. Then, a cost matrix is computed using the divergence matrix obtained using a non-iterative signal decomposition method previously developed by the authors in [15] that has been tuned to perform the projection of each combination of notes. Finally, a DTW strategy is performed in an offline and online manner. The proposed offline and online approaches have been tested using a dataset with different polyphony levels (from 2 to 4) and compared them with other reference methods. On average, our approaches (offline and online) obtain the best results in terms of precision within the compared offline and online approaches, respectively, and has demonstrated to be robust agains the analyzed polyphony.

In the future we plan to track the tempo changes in order to enforce a certain degree of continuity in the online decisions. Besides, we will extend the evaluation of our method using a lager dataset of a varied range of instruments, dynamics and different styles.

7. REFERENCES

[1] A. Cont, "A coupled duration-focused architecture for real-time music-to-score alignment," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 6, pp. 974987, Jun. 2010.

[2] N. Montecchio and N. Orio, "A discrete filterbank approach to audio to score matching for score following," *in Proc. ISMIR*, 2009, pp. 495-500.

[3] N. Hu, R. B. Dannenberg, and G. Tzanetakis, "Polyphonic audio matching and alignment for music retrieval," *in Proc. IEEE WASPAA*, 2003, pp. 185188.

[4] O. Izmirli and R. Dannenberg. "Understanding features and distance functions for music sequence alignment". In Proceedings of ISMIR, 411- 416. 2010

[5] M. Puckette, "Score following using the sung voice," *in Proc. ICMC*, 1995, pp. 175178.

[6] Z. Duan and B. Pardo, "Soundprism: An Online System for Score-informed Source Separation of Music Audio," *IEEE Journal of Selected Topics in Signal Process.*, vol. 5, no. 6, pp. 1205-1215, 2011.

[7] P. Cano, A. Loscos, and J. Bonada, "Score-performance matching using HMMs," *in Proc. ICMC*, 1999, pp. 441-444.

[8] A. Cont, "Realtime audio to score alignment for polyphonic music instruments using sparse non-negative constraints and hierarchical hmms," *In Proceedings of IEEE ICASSP*, Toulouse. France, 2006.

[9] C. Raphael, "Automatic segmentation of acoustic musical signals using hidden Markov models," *IEEE Trans. Pattern Anal. Machine Intell.,* vol. 21, no. 4, pp. 360370, Apr. 1999.

[10] C. Joder, S. Essid, and G. Richard. "Learning optimal features for polyphonic audio-to-score alignment." *IEEE Transactions on Audio, Speech, and Language Processing*, 21, 10, 2118-2128, 2013

[11] P. Cuvillier and A. Cont. "Coherent time modeling of Semi-Markov models with application to realtime audio-to-score alignment". *Proceedings of the 2014 IEEE International Workshop on Machine Learning for Signal Processing*. 16, 2014.

[12] N. Orio and D. Schwarz, "Alignment of monophonic and polyphonic music to a score," *in Proc. International Computer Music Conference (ICMC)*, 2001.

[13] S. Dixon, "Live tracking of musical performances using on-line time warping," *in Proc. International Conference on Digital Audio Effects (DAFx)*, Madrid, Spain, 2005, pp. 92-97.

[14] J. Fritsch and M. Plumbley. "Score Informed Audio Source Separation using Constrained Nonnegative Matrix Factorization and Score Synthesis," *In Proc. ICASSP*, Vancouver, Canada, 2013.

[15] J.J. Carabias-Orti et al., "Constrained non-negative sparse coding using learnt instrument templates for realtime music transcription," *Engineering Applications of Artificial Intelligence*, April 2013

[16] F. Itakura, "Minimum prediction residual principle applied to speech recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing,* vol. 23, pp. 5272, 1975.

[17] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimisation for spoken word recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 26, pp. 4349, 1978.

[18] D. D. Lee and H. S. Seung, "Algorithms for Non-negative Matrix Factorization," *in Proc. of Neural Information Processing Systems*, Denver, USA, 2000.

[19] C. Févotte and J. Idier "Algorithms for Nonnegative Matrix Factorization with the Beta-Divergence," *Neural Computation*, vol. 23, no. 9, pp. 2421-2456, September 2011.

[20] J. F. Gemmeke et al., "Exemplar-based sparse representations for noise robust automatic speech recognition," *IEEE Trans. Audio, Speech and Language Processing*, Volume: 19, Issue: 7, 2011.

[21] J.J. Carabias-Orti et al., "Musical Instrument Sound Multi-Excitation Model for Non-Negative Spectrogram Factorization," *IEEE Journal of Selected Topics in Signal Processing*, Vol. 5, no. 6, pp. 1144 - 1158, October 2011

[22] A. Cont et al., "Evaluation of real- time audio-to-score alignment," *in Proc. International Conference on Music Information Retrieval (ISMIR)*, 2007.

[23] D. Ellis. Dynamic Time Warp (DTW) in Matlab, Web resource, available: http://www.ee.columbia.edu/∼dpwe/resources/matlab/dtw/.

[24] R. Turetsky and D. Ellis "Ground-Truth Transcriptions of Real Music from Force-Aligned MIDI Syntheses", *in Proc. International Conference on Music Information Retrieval (ISMIR)*, 2003.

[25] B. Niedermayer, and G. Widmer. "A multi-pass algorithm for accurate audio-to-score alignment" *in Proc. International Conference on Music Information Retrieval (ISMIR)*, 2010.

[26] J. Devaney, "Estimating Onset and Offset Asynchronies in Polyphonic Score-Audio Alignment" *Journal of New Music Research* 43 (3), 266-275, 2014

[27] M. Miron, J. Carabias-Orti and J. Janer. "Improving Score-Informed Source Separation Through Audio To Score Note-Level Refinement " *in Proc. International Conference on Music Information Retrieval (ISMIR)*, 2015.