A REAL-TIME NMF-BASED SCORE FOLLOWER FOR MIREX 2012

J.J.Carabias, F.J.Rodriguez, P.Vera, P.Cabañas, F.J. Cañadas and N.Ruiz
Telecommunication Engineering Department, University of Jaen
Polytechnic School, Linares, Jaen, Spain
{carabias,fjrodrig,pvera,pcabanas,fcanadas,nicolas}@ujaen.es

ABSTRACT

This abstract describes our score follower submitted to the MIREX 2012 Real-time Audio to Score Alignment (a.k.a. Score Following) task.

1. INTRODUCTION

A real-time score follower is a program that synchronizes a performance with its score in real time. It estimates a score position for each input time frame of the performance and the estimation is made in an online fashion, (i.e. only using past frames). In this MIREX task, the score and the audio signals are given in the formats of MIDI and WAV. Here we describe an overview of our score follower.

The presented system has two separated stages, preprocessing and alignment. On the first one, we convert MIDI data into a reference audio signal using a sequencer and we analyze the provided information in order to define the states sequence and the basis function associated to each state. Each state is defined as an unique combination of notes. These basis functions are learned from the synthetic MIDI signal using a method based on NMF with β-divergence where the gains are fixed as the ground-truth transcription inferred from the MIDI. On the second stage, NMF with fixed based is used over the WAV signal resulting in a distortion matrix that can be interpreted as the cost of each state at each frame. Finally the score alignment is obtained using an on-line Dynamic Time Warping (DTW) over the distortion matrix in order to find the path with the minimum cost and then determine the states real duration.

In the following sections, we will describe the method and implementation in detail.

2. SYSTEM DESCRIPTION

First of all an overview of the score following system and an example of the alignment can be seen in Figure 1.

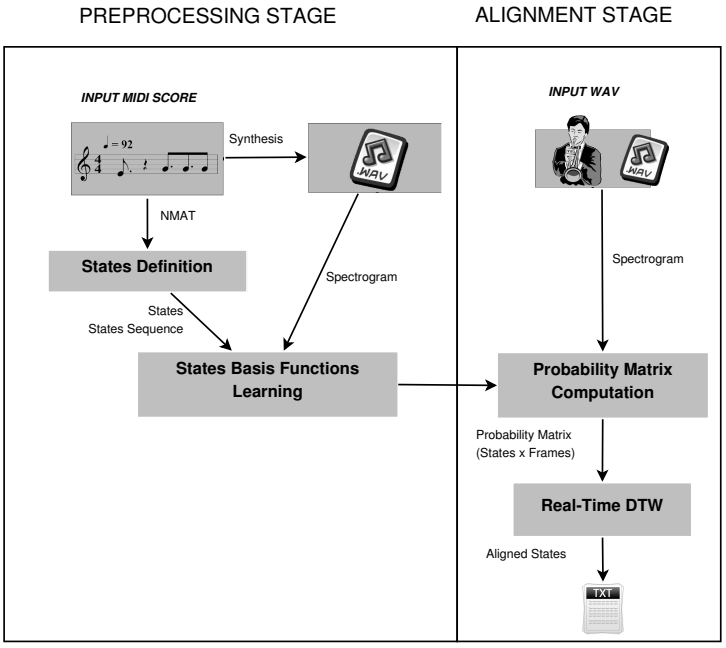

Figure 1. *Proposed Real-Time Score Follower Block Diagram*

2.1 Preprocessing Stage

2.1.1 States Definition

The aim of this stage is to compute the states and state sequence from the MIDI data. A state is defined as a combination of notes that occurs simultaneously in the ground-truth transcription inferred from the MIDI data and can be formulated as

$$S_k = \left\{ n_j^k, j = 1, ..., J, k = 1, ..., K \right\} \quad (1)$$

where n_j^k is the note played by instrument j, k is the state number, J is the total number of instruments and K the total number of states.

The states sequence is a $1xM$ vector that provides information about the states transitions in the MIDI data and is defined as

$$\Psi = \{ S_k^m, 1 \leqslant m \leqslant M \} \quad (2)$$

where S_k^m is the k-th state occurring at the m-th position in the state sequence vector and M is the total number of transitions between states.

2.1.2 Basis Functions Learning

Once the states and the states sequence have been defined, we will learn the basis functions associated to each state. To this end, we use a supervised method based on Non-Negative Matrix Factorizacion (NMF) with Multiplicative Update (MU) rules.

First of all, let us define the signal model as

$$x(f,t) \approx \hat{x}(f,t) = \sum_{k=1}^{K} g_k(t), b_k(f) \tag{3}$$

where $x(f,t)$ is the magnitude spectrogram of the synthetic signal generated from the MIDI data with a sequencer, $\hat{x}(f,t)$ is the estimated spectrogram, $g_k(t)$ is the gain of the basis function for state k at frame t, and $b_k(f)$, $k = 1, ..., K$ are the bases.

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-Leibner (KL) and the Itakura-Saito (IS) divergences. Besides, the β-divergence (see eq. 4) 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.

$$D_\beta(x|\hat{x}) = \begin{cases} \frac{1}{\beta(\beta-1)}\left(x^\beta + (\beta-1)\hat{x}^\beta - \beta x \hat{x}^{\beta-1}\right) & \beta \in (0,1) \cup (1,2] \\ x \log \frac{x}{\hat{x}} - x + \hat{x} & \beta = 1 \\ \frac{x}{\hat{x}} + \log \frac{x}{\hat{x}} - 1 & \beta = 0 \end{cases} \tag{4}$$

In order to obtain the model parameters that minimize the cost function, Lee *et al.* [1] proposes an iterative algorithm based on multiplicative update (MU) rules. Under these rules, $D\beta(x(f,t)|\hat{x}(f,t))$ 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 [1, 2]. For the model of eq. (3), multiplicative updates which minimize the β-divergence are defined as

$$b_k(f) \leftarrow b_k(f) \frac{\sum_{f,t} x(f,t)\hat{x}(f,t)^{\beta-2} g_k(t)}{\sum_{f,t} \hat{x}(f,t)^{\beta-1} g_k(t)} \tag{5}$$

$$g_k(t) \leftarrow g_k(t) \frac{\sum_{f,m} x(f,t)\hat{x}(f,t)^{\beta-2} b_k(f)}{\sum_{f,m} \hat{x}(f,t)^{\beta-1} b_k(f)} \tag{6}$$

Finally, the method to learn the basis functions for each state is described in Algorithm 2.

Note that $R_k(t)$ is a binary state/time matrix that represent the ground-truth transcription of the training data. Therefore, at each frame, the active state 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 state.

Algorithm 1 States Basis Functions Learning Method

1 Initialize $g_k(t)$ with the ground truth transcription $R_k(t)$ and $b_k(f)$ with random positive values.
2 Update the bases using eq. (5).
3 Repeat step 2 until the algorithm converges (or maximum number of iterations is reached).

2.2 Alignment Stage

2.2.1 Probability Matrix Computation

As explained in section 2.1.2, the basis functions $b_k(f)$ for each state are trained in advance using the MIDI data and kept fixed. Each basis function models the spectrum of an unique state.

Now, the aim is to compute the gain matrix $g_k(t)$ and the final cost matrix $D(t,k)$ that measures the likelihood between the estimated and the real spectrogram.

The process is detailed in Algorithm 2 and it is similar than the one in Algorithm 1.

Algorithm 2 States Basis Functions Learning Method

1 Initialize $b_k(f)$ with the values learned in section 2.1.2 and $g_k(t)$ with random positive values.
2 Update the gains using eq. (6).
3 Repeat step 2 until the algorithm converges (or maximum number of iterations is reached).
4 Compute the distortion matrix $D\beta(x|\hat{x})$ using eq. (4).

As can be inferred, the distortion matrix $D\beta(t,k)$ provides us information about the similitude of each state k basis function with the real signal spectrum at each frame t. Using this information, we can directly compute the probability matrix for the state sequence ψ as

$$D(t,m) = \{|D\beta_k^m(t)| \; 1 \leqslant m \leqslant M\} \tag{7}$$

Therefore, we should find the minimum cost path in order to determine the duration in the real performance of each state in the sequence. To this end, we have applied a real-time DTW as explained in the following section.

2.2.2 Real State Sequence Estimation by DTW

We used the following constrained DTW path:

$$D(t,m) = \min \left\{ \begin{array}{c} D(t-1,m) + d(t,m) \\ D(t-1,m-1) + 2d(t,m) \end{array} \right\} \tag{8}$$

where t is the index of the current performance frame to be searched, m is the index of current state in the state sequence Ψ, d(t, m) is value of the distortion computed with the β-divergence function for the t-th frame and the state in the m-th position in the sequence and $D(t,m)$ is the accumulated cost value at the t-th frame and the m-th state at the sequence. Note that this constrained path inhibits occurrence of vertical steps since only one state can be active at each frame.

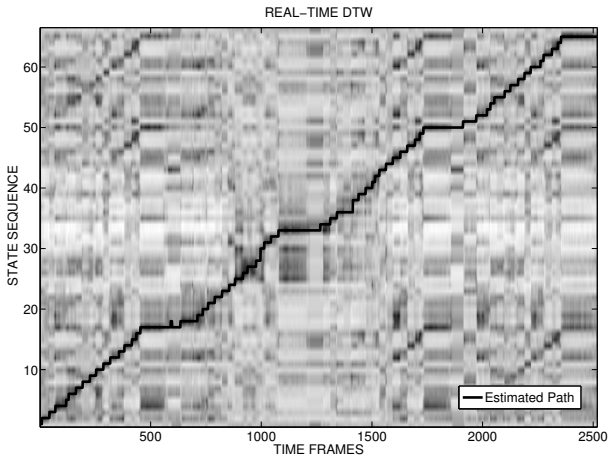

Figure 2. *Proposed Real-Time Score Follower Block Diagram*

2.2.3 Real-time DTW

Standard DTW assumes off-line search and the estimated path is obtained by backtracing of whole the signal. To extend DTW for the on-line search without backtracing, we simply select the reference state which has the smallest accumulated distance with the current performance frame t. An example of the on-line DTW performance is shown in Figure 3.

3. EVALUATION

4. REFERENCES

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

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