Soundprism: An Online System for Score-informed Source Separation of Music Audio

Zhiyao Duan, *Graduate Student Member, IEEE,* Bryan Pardo, *Member, IEEE*

***Abstract*—Soundprism, as proposed in this paper, is a computer system that separates single-channel polyphonic music audio played by harmonic sources into source signals in an online fashion. It uses a musical score to guide the separation process. To the best of our knowledge, this is the first online system that addresses score-informed music source separation that can be made into a real-time system.**

The proposed system consists of two parts: 1) a score follower that associates a score position to each time frame of the audio performance; 2) a source separator which reconstructs the source signals for each time frame, informed by the score. The score follower uses a hidden Markov approach, where each audio frame is associated with a 2-dimensional state vector (score position and tempo). The observation model is defined as the likelihood of observing the frame given the pitches at the score position. The score position and tempo are inferred using particle filtering. In building the source separator, we first refine the score-informed pitches of the current audio frame by maximizing the multi-pitch observation likelihood. Then, the harmonics of each source's fundamental frequency are extracted to reconstruct the source signal. Overlapping harmonics between sources are identified and their energy is distributed in inverse proportion to the square of their respective harmonic number. Experiments on both synthetic and human-performed music show both the score follower and the source separator perform well. Results also show that the proposed score follower works well for highly polyphonic music with some degree of tempo variations.

***Index Terms*—Source separation, score following, online algorithm, multi-pitch estimation.**

I. Introduction

FOR A ray of white light, a prism can separate it into multiple rays of light with different colors in real time. How about for sound? In this paper, we propose a computer algorithm called "Soundprism" to separate polyphonic music audio into source signals in an online fashion, given the musical score of the audio in advance. There are many situations where this algorithm could be used. Imagine a classical music concert where every audience member could select their favorite personal mix (e.g. switch between enjoying the full performance and concentrating on the cello part) even though the instruments are not given individual microphones. A soundprism could also allow remixing or upmixing of existing monophonic or stereo recordings of classical music, or live broadcasts of such music. Such a system would also be useful in an offline context, for making music-minus-one applications for performers to play along with existing music recordings.

Z. Duan and B. Pardo are with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208, USA. E-mail: zhiyaoduan00@gmail.com.

Unsupervised single-channel musical source separation is extremely difficult, even in the offline case [1]. Therefore, we consider the case where a musical score (in the form of MIDI) is available to guide the source separation process. Tens of thousands of MIDI scores for classical pieces are available on the web at sources such as http://www.classicalmidiconnection.com.

Source separation that is guided by a musical score is called score-informed source separation. In this scenario, the multi-source audio performance is faithful to the score with possible variations of tempo and instrumentation. Existing score-informed source separation systems either assumes the score and audio are well-aligned [2], [3], or use Dynamic Time Warping (DTW) to find the alignment [4], [5] before separating sources using the pitch information provided by the score. These are offline algorithms, since the DTW they use needs to see the whole audio performance from start to end, hence cannot be made in real time. There do exist some online DTW algorithms [6], [7], but to date, no work has explored using them to make an online source separation system.

In this paper, we address the score-informed source separation problem (for music with only harmonic sources) in an online fashion. An online algorithm is one that can process its input piece-by-piece in a serial fashion, in the order that the input (e.g. the audio stream) is fed to the algorithm, without having the entire input available from the start. Similarly to existing score-informed source separation methods, we decompose the whole problem into two stages: 1) audio-score alignment and 2) pitch-informed source separation. We differ from existing work in that audio-score alignment is done online and sources are separated in each 46-millisecond audio frame as soon as the frame's aligned score position is determined.

In the first stage, we use a hidden Markov process model, where each audio frame is associated with a 2-dimensional state (score position and tempo). After seeing an audio frame, our current observation, we want to infer its state. We use a multi-pitch observation model, which indicates how likely the current audio frame is to contain the pitches at a hypothesized score position. The inference of the score position and tempo of the current frame is achieved by particle filtering. In the second stage, score-informed pitches at the aligned score position are used to guide source separation. These pitches are first refined using our previous multi-pitch estimation algorithm [8], by maximizing the multi-pitch observation likelihood. Then, a harmonic mask in the frequency domain is built for each pitch to extract its source's magnitude spectrum. In building the mask, overlapping harmonics are identified and their energy is distributed in reverse proportion to the square of their harmonic numbers. Finally, the time domain signal of each source is reconstructed by inverse Fourier transform using the source's magnitude spectrum and the phase spectrum of the mixture. The whole process is outlined in Figure 1. To the best of our knowledge, this is the first paper addressing the online score informed source separation problem.

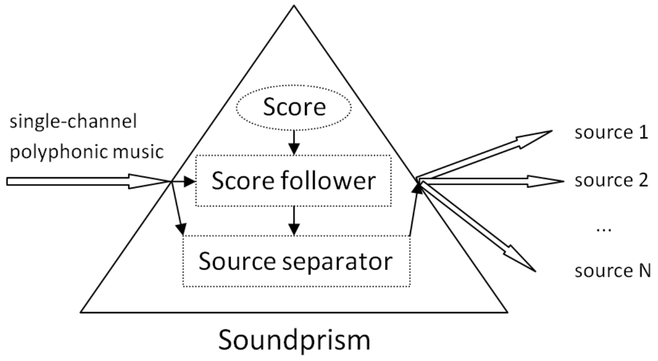


Fig. 1. System overview of Soundprism.

The remainder of this paper is arranged as follows: Section II presents the proposed online audio-score alignment algorithm; Section III describes the separation process given the score-informed pitches of each audio frame; Section IV describes computational complexity of the algorithms. Experimental results are presented in Section V and the paper is concluded in Section VI.

II. Real-time polyphonic audio-score alignment

The first stage of Soundprism is online polyphonic audio-score alignment, where polyphonic music audio is segmented into time frames and they are fed to the score follower in sequence. Soundprism outputs a score position for each frame, right after it is processed.

A. Prior Work

Most existing polyphonic audio-score alignment methods use Dynamic Time Warping (DTW) [9]–[11], a HMM [12], [13], a hybrid graphical model [14] or a conditional random field model [15]. Although these techniques achieve good results, they are offline algorithms, that is, they need the whole audio performance to do the alignment.

Dannenberg [16] and Vercoe [17] propose the first two real-time score followers, but both work for MIDI performance instead of audio. There are some real-time or online audio-score alignment methods [18]–[21]. However, these methods are for monophonic (one note at a time) audio performances. Two of these systems ([20], [21]) also require training of the system on prior performances of each specific piece before alignment can be performed for a new performance of the piece. This limits the applicability of these approaches to pieces with preexisting aligned audio performances.

For polyphonic audio, Grubb and Dannenberg [22] adopt string matching to follow a musical ensemble, where each instrument needs to be recorded by a close microphone and streamed into a monophonic pitch sequence. This method will not work in the situation where the instruments are not separately recorded, e.g. distance-miced acoustic ensembles. Dixon [6] proposes an online DTW algorithm to follow piano performances, where each audio frame is represented by an 84-d vector, corresponding to the half-wave rectified first-order difference of 84 spectral bands. This onset-informed low-level feature works well for piano performances, however, for instruments with smooth onsets like string and wind it may have difficulties.

Cont [23] proposes a hierarchical HMM approach to follow piano performances, where the observation likelihood is calculated by comparing the pitches at the hypothesized score position and pitches transcribed by Nonnegative Matrix Factorization (NMF) with fixed spectral bases. A spectral basis is learned for each pitch of the specific piano beforehand. This method might have difficulties in generalizing to multi-instrument polyphonic audio, as the timbre variation and tuning issues involved make it difficult to learn a general basis for each pitch. In [24], Cont proposes a probabilistic inference framework with two coupled audio and tempo agents to follow a polyphonic performance and estimate its tempo. This system works well on single-instrument polyphonic audio, but for multi-instrument polyphonic audio more statistical results are needed to evaluate the system's performance. Nevertheless, this is a state-of-the-art real-time score following system.

In this paper, we propose a novel on-line score follower that can follow a piece of multi-instrument polyphonic audio, without requiring training on prior performances of the exact piece to be followed.

B. Our Model Structure

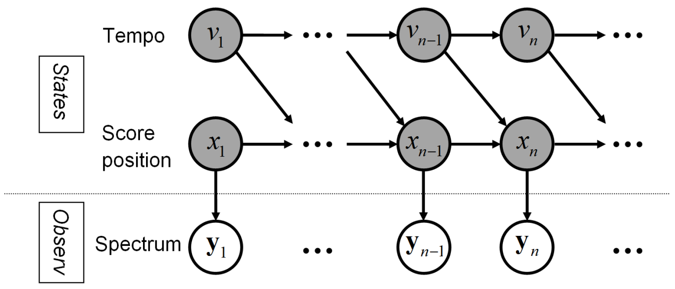


Fig. 2. Illustration of the state space model for online audio-score alignment.

We propose a hidden Markov process model, as illustrated in Figure 2. We decompose the audio performance into time frames and process the frames in sequence. The n-th frame is associated with a 2-dimensional hidden state vector $\mathbf{s}_n = (x_n, v_n)^T$, where x_n is its score position (in beats), v_n is its tempo (in Beat Per Minute (BPM)) and T denotes matrix transposition. x_n is drawn from the interval containing all score positions from the beginning to the end. v_n is drawn from the interval of all possible tempi $[v^l, v^h]$, where the lowest tempo v^l is set to half of the score tempo and the highest tempo v^h is set to twice the score tempo. These values were selected as broad limits on how far from the notated tempo a musician would be likely to deviate. Note that the the values of v^l and v^h can be chosen based on prior information about the score. In addition, for multi-tempi pieces, v^l and v^h can be changed correspondingly when a new tempo is encountered.

Each audio frame is also associated with an observation, which is a vector of PCM encoded audio, $\mathbf{y}_n$. Our aim is to infer the current score position x_n from current and previous observations $\mathbf{y}_1, \cdots, \mathbf{y}_n$. To do so, we need to define a process model to describe how the states transition, an observation model to evaluate hypothesized score positions for the current audio frame, and to find a way to do the inference in an online fashion.

C. Process Model

A process model defines the transition probability from the previous state to the current state, i.e. $p(\mathbf{s}_n|\mathbf{s}_{n-1})$. We use two dynamic equations to define this transition. To update the score position, we use

$$x_n = x_{n-1} + l \cdot v_{n-1} \tag{1}$$

where l is the audio frame hop size (10 milliseconds, in this work). Thus, score position of the current audio frame is determined by the score position of the previous frame and the current tempo. To update the tempo, we use

$$v_n = \begin{cases} v_{n-1} + n_v & \text{if } z_k \in [x_{n-1}, x_n] \text{ for some } k \\ v_{n-1} & \text{otherwise} \end{cases} \tag{2}$$

where $n_v \sim \mathcal{N}(0, \sigma_v^2)$ is a Gaussian noise variable; z_k is the k-th note onset/offset time in the score. This equation states that if the current score position has just passed a note onset or offset, then the tempo makes a random walk around the previous tempo according to a Gaussian distribution; otherwise the tempo remains the same.

The noise term n_v introduces randomness to our system, which is to account for possible tempo changes of the performance. Its standard deviation σ_v is set to a quarter of the notated score tempo through this paper. We introduce the randomness through tempo in Eq. (2), which will affect the score position as well. But we do not introduce randomness directly in score position in Eq. (1). In this way, we avoid disruptive changes of score position estimates.

In addition, randomness is only introduced when the score position has just passed a note onset or offset. This is because it is rather rare that the performer changes tempo within a note. Second, on the listener's side, it is impossible to detect the tempo change before hearing an onset or offset, even if the performer does make a change within a note. Therefore, changing the tempo state in the middle of a note is not in accordance with music performance, nor does it have evidence to estimate this change.

D. Observation Model

The observation model is to evaluate whether a hypothesized state can explain the observation, i.e. $p(\mathbf{y}_n|\mathbf{s}_n)$. Different representations of the audio frame can be used. For example, power spectra [25], auditory filterbank responses [26], chroma vectors [10], spectral and harmonic features [9], [20], multi-pitch analysis information [23], etc. Among these representations, multi-pitch analysis information is the most informative one to evaluate the hypothesized score position for most fully-scored musical works. This is because pitch information can be directly aligned to score information. Therefore, inspired by [23], we use multi-pitch observation likelihood as our preferred observation model.

In the proposed score follower, the multi-pitch observation model is adapted from our previous work on multi-pitch estimation [8]. It is a maximum likelihood-based method which finds the set of pitches that maximizes the likelihood of the power spectrum. This 'universal' likelihood model is trained on thousands of isolated musical chords generated by different combinations of notes from 16 kinds of instruments. These chords have different chord types (major, diminished, etc.), instrumentations, pitch ranges and dynamic ranges, hence the trained likelihood model performs well in multi-instrument polyphonic music pieces as shown in [8].

In [8], the frame of audio is first transformed to the frequency domain by a Fourier transform. Then, each significant peak of the power spectrum is detected and represented as a frequency-amplitude pair (f_i, a_i). Non-peak regions of the power spectrum are also extracted. The likelihood of the power spectrum given a set of hypothesized pitches $\boldsymbol{\theta} = \{F0_1, \cdots, F0_J\}$ is defined in the peak region and non-peak region respectively.

$$\mathcal{L}(\boldsymbol{\theta}) = \mathcal{L}_{\text{peak region}}(\boldsymbol{\theta}) \cdot \mathcal{L}_{\text{non-peak region}}(\boldsymbol{\theta}) \tag{3}$$

We assume spectral bins to be independent given the pitches, hence the peak region and non-peak region are conditionally independent. For the same reason, spectral peaks are also conditionally independent in the peak region likelihood:

$$\mathcal{L}_{\text{peak region}}(\boldsymbol{\theta}) = \prod_{k=1}^{K} p\left(f_k, a_k|\boldsymbol{\theta}\right) \tag{4}$$

$$\mathcal{L}_{\text{non-peak region}}(\boldsymbol{\theta}) \approx \prod_{F_0 \in \boldsymbol{\theta}} \prod_{\substack{h \in \{1 \cdots H\} \\ F_h \in \mathcal{F}_{\text{np}}}} \left(1 - P\left(e_h = 1|F_0\right)\right) \tag{5}$$

where F_h is the frequency of the predicted h-th harmonic of F_0; e_h is the binary variable that indicates whether this harmonic is present or not; $\mathcal{F}_{\text{np}}$ is the set of frequencies in the non-peak region; and H is the largest harmonic number we consider. Eq. (4) and (5) are further decomposed and their parameters are learned from training chords. $\mathcal{L}_{\text{peak region}}(\boldsymbol{\theta})$ is larger for pitch estimates whose harmonics better explain observed peaks; $\mathcal{L}_{\text{non-peak region}}(\boldsymbol{\theta})$ is larger for pitch estimates that better explain the observed non-peak region (i.e. have fewer harmonics in the non-peak region). The two likelihood models work as a complementary pair. Details of this approach are described in [8].

In calculating the observation likelihood $p(\mathbf{y}_n|\mathbf{s}_n)$ in our score follower, we extract the set of all pitches at score position x_n and use it as $\boldsymbol{\theta}$ in Eq. (3). Then $p(\mathbf{y}_n|\mathbf{s}_n)$ is defined as

$$p(\mathbf{y}_n|\mathbf{s}_n) = -\frac{C}{\ln \mathcal{L}(\boldsymbol{\theta})} \tag{6}$$

where C is the normalization factor to make it a probability.

Note that we do not define $p(\mathbf{y}_n|\mathbf{s}_n) = \mathcal{L}(\boldsymbol{\theta})$, because it turns out that $\mathcal{L}(\boldsymbol{\theta})$ can differ by orders of magnitude for different sets of candidate pitches drawn from the score. Since, we combine the process model and observation model to infer score position, this large variation in observation model outputs can cause the observation model to gain too much importance relative to the process model. For example, two very close score position hypotheses would get very different observation likelihood, if they indicate different sets of pitches. However, the probabilities calculated from the process model are not that different. Therefore, the posterior probability of score position would be overly influenced by the observation model, while the process model would be almost ignored. If the observation model does not function well in some frame (e.g. due to a pitch-estimation glitch), the estimated score position may jump to an unreasonable position, although the process model tends to proceed from the previous score position smoothly. Eq. (6) compresses $\mathcal{L}(\boldsymbol{\theta})$. This balances the process model and the observation model.

Note that in constructing this observation model, we do not need to estimate pitches from the audio frame. Instead, we use the set of pitches indicated by the score. This is different from [23], where pitches of the audio frame are first estimated, then the observation likelihood is defined based on the differences between the estimated pitches and score-informed pitches. By skipping the pitch estimation step, we can directly evaluate the score-informed pitches at a hypothesized score position using the audio observation. This reduces model risks caused by pitch estimation errors.

Our observation model only considers information from the current frame, and could be improved if considering information from multiple frames. Ewert et al. [11] incorporate inter-frame features to utilize note onset information and improve the alignment accuracy. Joder et al. [15] propose an observation model which uses observations from multiple frames for their conditional random field-based method. In the future we want to explore these directions to improve our score follower.

E. Inference

Given the process model and the observation model, we want to infer the state of the current frame from current and past observations. From a Bayesian point of view, this means we first estimate the posterior probability $p(\mathbf{s}_n|\mathbf{Y}_{1:n})$,then decide its value using some criterion like maximum a posterior (MAP) or minimum mean square error (MMSE). Here, $\mathbf{Y}_{1:n} = (\mathbf{y}_1, \cdots, \mathbf{y}_n)$ is a matrix whose each column denotes the observation in one frame. Recall $\mathbf{s}_n = (x_n, v_n)^T$, where x_n is score position (in beats), v_n is tempo (in Beat Per Minute (BPM)) and T denotes matrix transposition. By Bayes' rule, we have

$$\begin{aligned} & p(\mathbf{s}_n|\mathbf{Y}_{1:n}) \\ = \; & C_n p(\mathbf{y}_n|\mathbf{s}_n) \int p(\mathbf{s}_n|\mathbf{s}_{n-1}) p(\mathbf{s}_{n-1}|\mathbf{Y}_{1:n-1})\, d\mathbf{s}_{n-1} \end{aligned} \tag{7}$$

where $\mathbf{y}_n$, $\mathbf{Y}_{1:n}$, $\mathbf{s}_n$ and $\mathbf{s}_{n-1}$ are all random variables; $\mathbf{s}_{n-1}$ is integrated over the whole state space; C_n is the normalization factor. $p(\mathbf{y}_n|\mathbf{s}_n)$ is the observation model defined in Eq. (6) and $p(\mathbf{s}_n|\mathbf{s}_{n-1})$ is the process model defined by Eq. (1) and (2).

We can see that Eq. (7) is a recursive equation of the posterior probability $p(\mathbf{s}_n|\mathbf{Y}_{1:n})$. It is updated from the posterior probability in the previous frame $p(\mathbf{s}_{n-1}|\mathbf{Y}_{1:n-1})$, using the state transition probability $p(\mathbf{s}_n|\mathbf{s}_{n-1})$ and the observation probability $p(\mathbf{y}_n|\mathbf{s}_n)$. Therefore, if we can initialize $p(\mathbf{s}_1|\mathbf{y}_1)$ in the first frame and update it using Eq. (7) as each frame is processed, the inference can be done online.

This is the general formulation of online filtering (tracking). If all the probabilities in Eq. (7) are Gaussian, then we just need to update mean and variance of the posterior in each iteration. This is the Kalman filtering method. However, the observation probability $p(\mathbf{y}_n|\mathbf{s}_n)$ is very complicated. It may not be Gaussian and may not even be unimodal. Therefore, we need to update the whole probability distribution. This is not easy, since integration at each iteration is hard to calculate.

Particle filtering [27], [28] is a way to solve this problem, where the posterior distribution is represented and updated using a fixed number of particles together with their importance weights. In the score follower, we use the *bootstrap filter*, one variant of particle filters, which assigns equal weight to all particles in each iteration.

```
 1: Initialize $M$ particles $(x^{(1)}, v^{(1)}) \cdots, (x^{(M)}, v^{(M)})$,
    where $x^{(i)} = 1$ and $v^{(i)} \sim U[v^l, v^h]$.
 2: Initialize $x_{-1} = x_0 = 1$ and $v_0 =$ notated score tempo
 3: for $n = 1$ to the last frame $N$ do
 4:     $x^{(i)} \leftarrow x^{(i)} + l \cdot v^{(i)}$ by Eq. (1)
 5:     if $x_{n-2} \le z_k \le x_{n-1}$ then
 6:         $v^{(i)} \leftarrow v_{n-1} + n_v$ by Eq. (2)
 7:     else
 8:         $v^{(i)} \leftarrow v^{(i)}$
 9:     end if
10:     $w^{(i)} \leftarrow p(\mathbf{y}_n|(x^{(i)}, v^{(i)}))$ by Eq. (6)
11:     $w^{(i)} \leftarrow w^{(i)}/\Sigma w^{(i)}$
12:     Sample particles with replacement according to $w^{(i)}$ to
        get a new set of particles $(x^{(1)}, v^{(1)}) \cdots, (x^{(M)}, v^{(M)})$.
13:     Output $x_n = \frac{1}{M}\Sigma x^{(i)}$ and $v_n = \frac{1}{M}\Sigma v^{(i)}$.
14: end for
```

Fig. 3. The bootstrap filter algorithm for score following. Here $(x^{(i)}, v^{(i)})$ is the i-th particle and $w^{(i)}$ is its importance weight. All the other variables are the same as in the text.

Figure 3 presents the algorithm applied to score following. In Line 1, M particles are initialized to have score positions equal to the first beat and tempi assume a uniform distribution. Line 3 starts the iteration through the frames of audio. At this point, these particles represent the posterior distribution $p(\mathbf{s}_{n-1}|\mathbf{Y}_{1:n-1})$ of $\mathbf{s}_{n-1}$. From Line 4 to 9, particles are updated according to the process model in Eq. (1) and (2) and now they represent the conditional distribution $p(\mathbf{s}_n|\mathbf{Y}_{1:n-1})$ of $\mathbf{s}_n$. In Line 10 and 11, the importance weights of particles are calculated as their observation likelihood according to Eq. (6), and then normalized to a discrete distribution. Then in Line 12, these particles are resampled with replacement according to their weights to generate a new set of M particles.

This is the key step of a bootstrap filter, after which the new particles can be thought of having equal weights. These particles now represent the posterior distribution $p(\mathbf{s}_n|\mathbf{Y}_{1:n})$ of $\mathbf{s}_n$, and we output their mean as the score position and tempo estimate in the n-th frame in Line 13.

In updating the tempo of each particle in Line 6, instead of using its previous tempo $v^{(i)}$, we use the previously estimated tempo v_{n-1}, i.e. the average tempo of all particles in the previous frame. This practical choice avoids that the particles become too diverse after a number of iterations due to the accumulation of randomness of n_v.

The set of particles is not able to represent the distribution if there are too few, and is time-consuming to update if there are too many. In this paper, we tried to use 100, 1,000 and 10,000 particles. We find that with 100 particles, the score follower is often lost after a number of frames. But with 1000 particles, this rarely happens and the update is still fast enough. Therefore, 1000 particles are used in this paper.

Unlike some other particle filters, the bootstrap filter we use does not have the common problem of degeneracy, where most particles have negligible importance weights after a few iterations [27], [28]. This is because the resampling step (Line 12 in Figure 3) in each iteration eliminates those particles whose importance weights are too small, and the newly sampled particles have equal weights again. This prevents the skewness in importance weights from accumulating.

At each time step, the algorithm outputs the mean score position from the set of particles as the estimate of the current score position. Someone may suggest choosing MAP or median, since the mean value may lie in a low probability area if the distribution is not unimodal. However, we find that in practice there is not much difference in choosing mean, MAP or median. This is because the particles in each iteration generally only cover a small range of the score (usually less than 0.5 beat), and mean, MAP and median are close.

III. Source Separation in A Single Frame

If the estimate of the score position of the current audio frame is correct, the score can tell us what pitch (if any) is supposed to be played by each source in this frame. We will use this pitch information to guide source separation.

A. Prior Work

Existing score-informed source separation systems use different methods to separate sources. Woodruff et al. [4] work on stereo music, where spatial cues are utilized together with score-informed pitch information to separate sources. This approach does not apply to our problem, since we are working on single-channel source separation. Ganseman et al. [5] use Probabilistic Latent Component Analysis (PLCA) to learn a source model from the synthesized audio of the source's score, and then apply these source models to real audio mixtures. In order to obtain good separation results, the synthesized audio should have similar timbre to the source signal. However, this is not the case in our problem, since we do not know what instrument is going to play each source in the audio performance. Raphael [2] trains a model to classify time-frequency bins that belong to solo or accompaniment using a labeled training set, then applies this model to separate the solo from the mixture. This method, however, cannot separate multiple sources from the mixture.

In [29], Li et al. propose a single-channel music separation method when pitches of each source in the music is given. They use a least-square estimation framework to incorporate an important organizational cue in human auditory perception, common amplitude modulation, to resolve overlapping harmonics. This approach achieves good results in separating polyphonic musical sources, however, the least square estimation is performed for harmonic trajectories in the spectrogram hence it is not an online algorithm.

In the following, we develop a simple source separation method, which works in an online fashion on polyphonic music of unknown instruments.

B. Refine Pitches

The pitches provided by the score $\boldsymbol{\theta} = \{F0_1, \cdots, F0_J\}$ are integer MIDI pitch numbers. MIDI pitch numbers indicate keys on the piano keyboard. Typically, MIDI 69 indicates the A above Middle C. Assuming A440-based equal temperament allows translation from MIDI pitch to frequency in Hz. The resulting frequencies are rarely equal to the real pitches played in an audio performance. In order to extract harmonics of each source in the audio mixture, we need to refine them to get accurate estimates $\hat{\boldsymbol{\theta}} = \{\hat{F0}_1, \cdots, \hat{F0}_J\}$. We refine the pitches using the multi-pitch estimation algorithm as described in [8], but restricting the search space in the Cartesian product $\prod_i [F0_i - 50\text{cents}, F0_i + 50\text{cents}]$. The algorithm maximizes the multi-pitch likelihood $\mathcal{L}(\hat{\boldsymbol{\theta}})$ in Eq. (3) with a greedy strategy, i.e. refining (estimating) pitches one by one. The set of refined pitches $\hat{\boldsymbol{\theta}}$ starts from an empty set. In each iteration, the refined pitch that improves the likelihood most is added to $\hat{\boldsymbol{\theta}}$. Finally, we get the set of all refined pitches. In refining each pitch $F0_i$, we search $\hat{F0}_i$ in $[F0_i - 50\text{cents}, F0_i + 50\text{cents}]$ with a step of 1Hz.

C. Reconstruct Source Signals

For each source in the current frame, we build a soft frequency mask and multiply it with the magnitude spectrum of the mixture signal to obtain the magnitude spectrum of the source. Then we apply the original phase of the mixture to the magnitude spectrum to calculate the source's time-domain signal. Finally, the overlap-add technique is applied to concatenate the current frame to previously generated frames. The sum of the masks of all the sources equals one in each frequency bin, so that the sources sum up to the mixture.

In order to calculate masks for sources, we first identify their harmonics and overlapping situations from the estimated pitches. For each source, we only consider the lowest 20 harmonics, each of which covers 40Hz in the magnitude spectrum. This width is assumed to be where the main lobe of each harmonic decreases 6dB from the center, when we use a 46ms Hamming window. These harmonics are then classified into overlapping harmonics and non-overlapping harmonics, according to whether the harmonic's frequency range is overlapped with some other harmonic's frequency range of another source.

All frequency bins in the spectrum can then be classified into three kinds: a *nonharmonic bin* which does not lie in any harmonic's frequency range of any source, a *non-overlapping harmonic bin* which lies in a non-overlapping harmonic's frequency range and an *overlapping harmonic bin* which lies in an overlapping harmonic's frequency range. For different kinds of bins, masks are calculated differently.

For a nonharmonic bin, masks of all active sources are set to $1/J$, where J is the number of pitches (active sources) in the current frame. In this way the energy of the mixture is equally distributed to all active sources. Although energy in nonharmonic bins is much smaller than that in harmonic bins, experiments show that distributing the energy reduces artifacts in separated sources, compared to discarding it. For a non-overlapping harmonic bin, the mask of the source that the harmonic belongs to is set to 1 and the energy of the mixture is assigned entirely to it.

For an overlapping harmonic bin, the masks of the sources whose harmonics are involved in this overlapping situation, are set in inverse proportion to the square of their harmonic numbers (e.g. 3 is the harmonic number of the third harmonic). For example, suppose a bin is in a harmonic which is overlapped by $J-1$ harmonics from other sources. Then the mask of the i-th source in this bin is defined as

$$m_i = \frac{1/h_i^2}{\Sigma_{j=1}^{J} 1/h_j^2} \tag{8}$$

where h_k is the harmonic number of the k-th source.

This simple method to resolve overlapping harmonics corresponds to the assumption that 1) overlapping sources have roughly the same amplitude; 2) all notes have harmonics amplitudes decay at 12dB per octave from the fundamental, regardless of pitch and instrument that produced the note. These assumptions are very coarse and will never be fulfilled in the real world. One can improve upon these assumptions by designing a more delicate source filter [30], interpolating the overlapping harmonics from non-overlapping harmonics based on the spectral smoothness assumption in each frame [31], or the temporal envelope similarity assumption of different harmonics of one note [32] or both [33]. Nevertheless, Eq. (8) gives a simple and relatively effective way to resolving overlapping harmonics as shown in experiments.

IV. Computational Complexity

Soundprism is an online algorithm that makes a Markovian assumption. The score follower considers only the result of the previous time-frame and the current spectrum in calculating its output. Therefore the number of operations performed at each frame is bounded by a constant value in terms of the number of past frames. We analyze this constant in terms of the number of particles M (on the order of 1000 in our implementation), the number of spectral peaks in the mixture K (on the order of 100), the number of sources J (typically less than 10), the number of searching steps S in refining a score-informed pitch (on the order of 1000 in our implementation) and the number of harmonics H (20 in our implementation) in reconstructing the source signals.

In the score following stage, Line 4-9 and 11-13 in Figure 3 all involve $O(M)$ calculations. Line 10 involves M times observation likelihood calculations, each of which calculates a multi-pitch likelihood of the chord at the score position of the particle. However, these M particles usually only cover a short segment (less than 0.5 beats) of the score. Within the span of a beat there are typically few note changes (16 would be an extreme). Therefore there are usually a small number of potential pitch sets to estimate the likelihood of (less than 16). Therefore, we only need a few likelihood calculations, each of which is of $O(K+J)$ according to [8]. The number of sources J is much smaller than the number of spectral peaks K and can be ignored, so the score following stage requires in total $O(M+K)$ calculations.

In the separation stage, we first refine the J score-informed pitches. This takes $O(JS)$ times the number of multi-pitch likelihood calculations, hence is of $O(JSK)$. The reconstruction of source signals takes $O(H^2)$ to identify overlapping situations and $O(J)$ to calculate source signals. Therefore, in total the complexity of Soundprism is $O(JSK+H^2+M)$. In order to reduce the complexity, we can use a smaller S with a slight degradation of separation results (about 0.5dB in SDR from $S=1000$ to $S=100$).

In our experiments, Soundprism is implemented in Matlab and runs about 3 times slower than real time on a four-core 2.67GHz CPU under Windows 7.

V. Experiments

A. Datasets

A musical work that we can apply Soundprism to must have the following components: a MIDI score, and a single-channel audio mixture containing the performance. In order to measure the effectiveness of the system, we must also have the alignment between MIDI and audio (to measure the alignment system's performance) and a separate audio recording for each instrument's part in the music (to measure source separation performance).

In this work, we use two datasets, one synthetic and one real. The synthetic dataset is adapted from Ganseman [34]. It contains 20 single-line MIDI melodies made from random note sequences. Each melody is played by a different instrument (drawn from a set of sampled acoustic and electric instruments). Each melody is 10 seconds long and contains about 20 notes. Each MIDI melody has a single tempo but is rendered to 11 audio performances with different dynamic tempo curves, using Timidity++ with the FluidR3 GM soundfont on Linux. The statistics of the audio renditions of each melody are presented in Table I. "Max tempo deviation" measures the maximal tempo deviation of the rendition from the MIDI. "Max tempo fluctuation" measures the maximal relative tempo ratio within the dynamic tempo curve.

We use these monophonic MIDI melodies and their audio renditions to generate polyphonic MIDI scores and corresponding audio performances, with polyphony ranging from 2

TABLE I
STATISTICS OF 11 AUDIO PERFORMANCES RENDERED FROM EACH MONOPHONIC MIDI IN GANSEMAN'S DATASET [34].

Max tempo deviation	0%	10%	20%	30%	40%	50%
Max tempo fluctuation	1.00	1.20	1.43	1.71	2.06	2.50
Num. of performances	1	2	2	2	2	2

to 6[1]. For each polyphony, we generate 24 polyphonic MIDI pieces by randomly selecting and mixing the 20 monophonic MIDI melodies. We generate 24 corresponding audio pieces, 4 for each of the 6 classes of tempo variations. Therefore, there are in total 120 polyphonic MIDI pieces with corresponding audio renditions. Alignment between MIDI and audio can be obtained from the audio rendition process and are provided in [34]. Although this dataset is not musically meaningful, we use it to test Soundprism on audio mixtures with different polyphonies and tempi, which are two important factors in following polyphonic music. In addition, the large variety of instruments in this dataset lets us see the adaptivity of Soundprism to different timbres.

The second dataset consists of 10 J.S. Bach four-part chorales, each of which is about 30 seconds long. The scores were MIDI downloaded from the internet[2]. The audio files are recordings of real music performances. Each piece is performed by a quartet of instruments: violin, clarinet, tenor saxophone and bassoon. Each musician's part was recorded in isolation while the musician listened to the others through headphones. Individual lines were then mixed to create 10 performances with four-part polyphony. We also created audio files containing all duets and trios for each piece, totalling 60 duets and 40 trios. The ground-truth alignment between MIDI and audio was interpolated from annotated beat times of the audio. The annotated beats were verified by a musician through playing back the audio together with these beats. We note that, beside the general tempo difference between the audio and MIDI pieces, there is often a fermata after a musical phrase in the audio but not in the MIDI. Therefore, there are many natural tempo changes in the audio while the MIDI has a constant tempo. We use this dataset to test Soundprism's performance in a more realistic situation.

B. Error Measures

We use the BSS_EVAL toolbox [36] to evaluate the separation results of Soundprism. Basically, each separated source is decomposed into a true source part and error parts corresponding to interferences from other sources and algorithmic artifacts. By calculating the energy ratios between different parts, the toolbox gives three metrics: *Signal-to-Interference Ratio (SIR)*, *Signal-to-Artifacts Ratio (SAR)* and *Signal-to-Distortion Ratio (SDR)* which measures both interferences and artifacts.

We use *Align Rate (AR)* as proposed in [38] to measure the audio-score alignment results. For each piece, AR is defined as the proportion of correctly aligned notes in the score. This measure ranges from 0 to 1. A score note is said to be correctly aligned if its onset is aligned to an audio time which deviates less than 50ms from the true audio time. We note that MIREX[3] uses average AR (called *Piecewise Precision*) of pieces in a test dataset to compare different score following systems.

We also propose another metric called *Average Alignment Error (AAE)*, which is defined as the average absolute difference between the aligned score position and the true score position of each frame of the audio. The unit of AAE is musical beat and it ranges from 0 to the maximum number of beats in the score.

We argue that AR and AAE measure similar but different aspects of an alignment. Notice that AR is calculated over note onsets in the audio time domain, while AAE is calculated over all audio frames in the score time domain. Therefore, AR is more musically meaningful and more appropriate for applications like real-time accompaniment. For example, if an alignment error of of a note is 0.1 beats, then the corresponding alignment error in the audio time domain can be either 100ms if the tempo is 60BPM or 33.3ms if the tempo is 180BPM, which induce significantly different accompaniment perceptions. AAE, however, is more appropriate for applications like score-informed source separation, since not only note onsets but all audio frames need to be separated. In addition, AAE is well correlated with the accuracy of score-informed pitches given the typical lengths of notes in a piece of music, hence helps analyze the main factor of source separation errors. For example, suppose the shortest note is an eighth-note, then AAE of 0.2 beats will indicate a high accuracy of score-informed pitches, and the score following stage will not be the main factor causing source separation errors.

In [38] there is another important metric called "latency" to measure the time delay of an online score follower from detecting to reporting a score event. We do not need this metric since the score follower in Soundprism computes an alignment right after seeing the input audio frame and the computation time is negligible. Therefore, there is no inherent delay in the score follower. The only delay from the audio frame being performed to the aligned score position being output is the frame center hop size, which is 10ms in this work.

C. Reference Systems

We compare Soundprism with four source separation reference systems. *Ideally-aligned* is a separation system which uses the separation stage of Soundprism, working on the ground-truth audio-score alignment. This removes the influence of the score follower and evaluates the source separation stage only.

Ganseman10 is a score-informed source separation system proposed by Ganseman et al. [5], [34]. We use an implementation provided by Ganseman. This system first aligns audio and score in an offline fashion, then uses a Probabilistic Latent Component Analysis (PLCA)-based method to extract sources

[1]Note that sources in this paper are all monophonic, so polyphony equals the number of sources.

[2]http://www.jsbchorales.net/index.shtml

[3]The Music Information Retrieval Evaluation eXchange (MIREX) is an annual evaluation campaign for Music Information Retrieval (MIR) algorithms. Score Following is one of the evaluation tasks. http://www.music-ir.org/mirex

according to source models. Each source model is learned from the MIDI-synthesized audio from the source's score. For the synthetic dataset, these audio pieces are provided by Ganseman. For the real music dataset, these audio pieces are synthesized using the Cubase 4 DAW built-in synthesis library without effects. Instruments in the synthesizer are selected to be the same as the audio mixture, to make the timbre of each synthesized source audio as similar as possible to the real source audio. However, in real scenarios that the instruments of the sources are not recognizable, the timbre similarity between the synthesized audio and the real source cannot be guaranteed and the system may degrade.

MPET is a separation system based on our previous work on multi-pitch estimation [8] and tracking [35]. The system obtains pitch estimates at each frame for each source after multi-pitch estimation and tracking. Then the separation stage of Soundprism is applied on these pitch estimates to extract sources. Note that the score information is not utilized in this system.

Oracle separation results are calculated using the BSS_Oracle toolbox [37]. They are the theoretically, highest achievable results of the time-frequency masking-based methods and serve as an upper bound of source separation performance. It is noted that oracle separation can only be obtained when the reference sources are available.

We compare the score following stage of Soundprism with *Scorealign*, which is an open-source offline audio-score alignment system[4] based on the method described in [10].

D. Score Alignment Results

1) Synthetic Dataset: Table II shows the score alignment results of Soundprism and Scorealign for different polyphony on the synthetic dataset. It can be seen that Scorealign obtains higher than 50% average Align Rate (AR) and less than 0.2 beats Average Alignment Error (AAE) for all polyphony, while Soundprism's results are significantly worse, especially for polyphony 2. However, as polyphony increases, the gap between Soundprism and Scorealign is significantly reduced. This supports our claim that the score following stage of Soundprism works better for high polyphony pieces.

TABLE II
AUDIO-SCORE ALIGNMENT RESULTS (AVERAGE±STD) VERSUS POLYPHONY ON THE SYNTHETIC DATASET. EACH VALUE IS CALCULATED FROM 24 MUSICAL PIECES.

metric	AR (%)		AAE (beat)	
polyphony	Soundprism	Scorealign	Soundprism	Scorealign
2	27.6±17.3	50.1±27.4	0.60±0.64	0.15±0.08
3	36.3±16.5	51.6±24.2	0.25±0.20	0.13±0.07
4	41.4±13.7	53.9±23.3	0.21±0.09	0.15±0.09
5	47.0±18.7	60.8±20.1	0.24±0.10	0.16±0.09
6	49.8±19.6	55.5±23.8	0.30±0.23	0.18±0.09

Table III indicates that the score following stage of Soundprism slowly degrades as the tempo variation increases, but as quickly as Scorealign. For Soundprism on tempo variation from 0% to 30%, AR are around 45% and AAE are around 0.25 beats. Then they degrades to about 30% of AR and 0.4 beats of AAE. Results of Scorealign, however, obtains almost perfect alignment on pieces with no tempo variation. Then it degrades suddenly to about 50% of AR and 0.18 beats of AAE. Remember that in the case of 50% tempo variation, the tempo of the fastest part of the audio performance is 2.5 times of the slowest part (refer to Table I), while the score tempo is a constant. This is a very difficult case for online audio-score alignment.

[4]http://sourceforge.net/apps/trac/portmedia/wiki/scorealign

TABLE III
AUDIO-SCORE ALIGNMENT RESULTS VERSUS TEMPO VARIATION ON THE SYNTHETIC DATASET.

metric	PPR (%)		AAE (beat)	
tempo	Soundprism	Scorealign	Soundprism	Scorealign
0%	47.1±22.0	96.6±5.4	0.28±0.39	0.01±0.01
10%	51.6±18.9	38.7±16.9	0.31±0.49	0.18±0.05
20%	44.3±16.4	50.5±11.5	0.22±0.07	0.16±0.06
30%	41.5±14.2	51.6±12.5	0.25±0.12	0.16±0.04
40%	27.9±13.0	48.2±17.9	0.46±0.49	0.19±0.07
50%	30.0±15.7	40.7±14.9	0.39±0.25	0.22±0.05

TABLE IV
AUDIO-SCORE ALIGNMENT RESULTS VERSUS POLYPHONY ON THE BACH CHORALE DATASET.

metric	PPR (%)		AAE (beat)	
polyphony	Soundprism	Scorealign	Soundprism	Scorealign
2	53.8±13.9	45.1±9.2	0.17±0.16	0.19±0.04
3	60.6±12.7	45.7±8.5	0.13±0.03	0.19±0.04
4	69.3±9.3	46.6±8.7	0.12±0.03	0.15±0.05

2) Real Music Dataset: Table IV shows audio-score alignment results versus polyphony when measured on real human performances of Bach chorales. Here, Soundprism performs better than Scorealign on both PPR and AAE. This may indicate that the score following stage of Soundprism is more adapted for real music pieces than pieces composed of random notes. More interestingly, the average AAE of Soundprism decreases from 0.17 to 0.12 when polyphony increases. Again, this suggests the ability of dealing with high polyphony of our score follower. In addition, the average AAE of Soundprism is less than a quarter beat for all polyphony. Since the shortest notes in these Bach chorales are sixteenth notes, the score follower is able to find correct pitches for most frames. This explains why the separation results between Soundprism and Ideally-aligned are very similar in Figure 8.

E. Source Separation Results

1) Synthetic Dataset: Figure 4 shows boxplots of the overall separation results of the five separation systems on pieces of polyphony 2. Each box represents 48 data points, each of which corresponds to the audio from one instrumental melody in a piece. The lower and upper lines of each box show 25th and 75th percentiles of the sample. The line in the middle of each box is the sample median. The lines extending above and below each box show the extent of the rest of the samples, excluding outliers. Outliers are defined as points over 1.5 times the interquartile range from the sample median and are shown as crosses.

For pieces of polyphony 2, if the two sources are of the same loudness, then the SDR and SIR of each source in

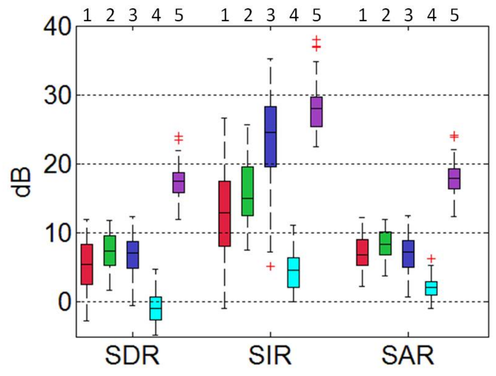


Fig. 4. Separation results on pieces of polyphony 2 from the synthetic dataset for Soundprism (1, red), Ideally-aligned (2, green), Ganseman10 (3, blue), MPET (4, cyan) and a perfect Oracle (5, purple). Each box represents 48 data points, each of which corresponds to an instrumental melodic line in a musical piece from the synthetic data set. Higher values are better.

the unseparated mixture should be 0dB. It can be seen that Soundprism improves the median SDR and SIR to about 5.5dB and 12.9dB respectively. Ideal alignment further improves SDR and SIR to about 7.4dB and 15.0dB respectively. This improvement is statistically significant in a nonparametric sign test with $p < 10^{-6}$. This suggests that the score following stage of Soundprism has space to improve. Comparing Soundprism with Ganseman10, we can see that they get similar SDR ($p = 0.19$) and SAR ($p = 1$) while Ganseman10 gets significant higher SIR ($p < 10^{-7}$). But remember that Ganseman10 uses an offline audio-score alignment and needs to learn a source model from MIDI-synthesized audio of each source. Without using score information, MPET obtains significantly worse results than all the three score-informed source separation systems. This supports the idea of using score information to guide separation. Finally, Oracle results are significantly better than all the other systems. Especially for Ideally-aligned, this gap of performance indicates that the separation stage of Soundprism has plenty of room to improve.

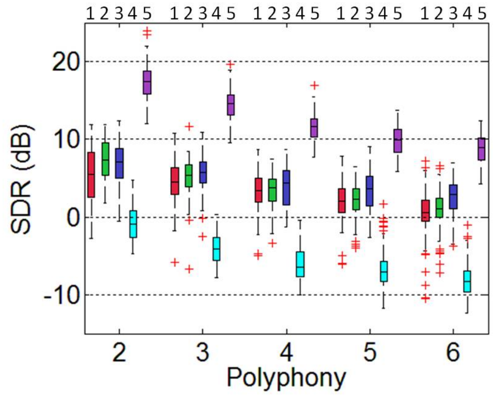


Fig. 5. SDR versus polyphony on the synthetic dataset for Soundprism (1, red), Ideally-aligned (2, green), Ganseman10 (3, blue), MPET (4, cyan) and Oracle (5, purple). Each box of polyphony n represents $24n$ data points, each of which corresponds to one instrumental melodic line in a musical piece.

Figure 5 shows SDR comparisons for different polyphony. SIR and SAR comparisons are omitted as they have the same trend as SDR. It can be seen that when polyphony increases, the performance difference between Soundprism and Ideally-aligned gets smaller. This is to be expected, given that Table II shows our score following stage performs better for higher polyphony. Conversely, the difference between Soundprism and Ganseman10 gets larger. This suggests that pre-trained source models are more beneficial for higher polyphony. Similarly, the performance gap from MPET to the three score-informed separation systems gets larger. This suggests that score information is more helpful for higher polyphony pieces.

The good results obtained by Scorealign helps the separation results of Ganseman10, as they use the same audio-score alignment algorithm. However, as the SDR obtained by Soundprism and Ganseman10 in Figure 4 and 5 are similar, the performance difference of their audio-score alignment stages is not vital to the whole separation systems.

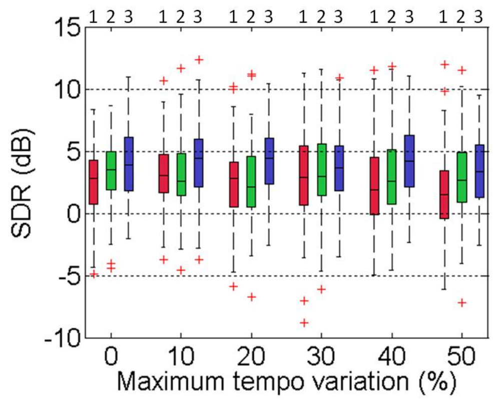


Fig. 6. SDR versus tempo variation on the synthetic dataset for Soundprism (1, red), ideally-aligned (2, green) and Ganseman10 (3, blue). Each box represents 80 data points, each of which corresponds to one instrumental melodic line in a musical piece.

It is also interesting to see how score-informed separation systems are influenced by the tempo variation of the audio performance. Figure 6 shows this result. It can be seen that the median SDR of Soundprism slowly degrades from 2.8dB to 1.9dB as the max tempo deviation increases from 0% to 50%. A two sample t-test with $\alpha = 0.05$ shows the mean SDR of the first 5 cases are not significantly different, while the last one is significantly worse. This supports the conclusion that the score following stage of Soundprism slowly degrades as the tempo variation increases, but not much.

2) *Real Music Dataset:* Next we compare these separation systems on a real music dataset, i.e. the Bach chorale dataset.

Figure 7 first shows the overall results on pieces of polyphony 2. There are four differences from the results of synthetic dataset in Figure 4. First, the results of Soundprism and Ideally-aligned are very similar on all measures. This suggests that the score following stage of Soundprism performs well on these pieces. Second, the difference between Soundprism/Ideally-aligned and Oracle is not that great. This indicates that the separation strategy used in Section III-C is suitable for the instruments in this dataset. Third, Soundprism obtains a significantly higher SDR and SAR than Ganseman10

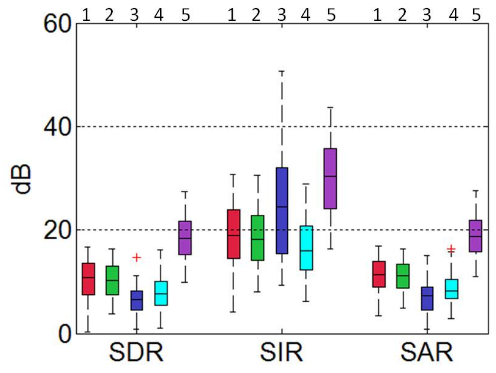


Fig. 7. Separation results on pieces of polyphony 2 from the Bach chorale dataset for Soundprism (1, red), Ideally-aligned (2, green), Ganseman10 (3, blue), MPET (4, cyan) and Oracle (5, purple). Each box represents 120 data points, each of which corresponds to one instrumental melodic line in a musical piece.

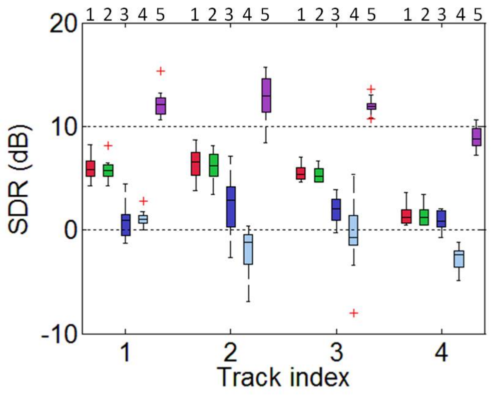


Fig. 9. SDR versus instrumental track indices on pieces of polyphony 4 in the Bach chorale dataset for Soundprism (1, red), Ideally-aligned (2, green), Ganseman10 (3, blue), MPET (4, cyan) and Oracle (5, purple). Tracks are ordered by frequency, i.e., in a quartet Track 1 is soprano and Track 4 is bass.

while a lower SIR. This indicates that Ganseman10 performs better in removing interference from other sources while Soundprism introduces less artifacts and leads to less overall distortion. Finally, the performance gap between MPET and the 3 score-informed source separation systems is significantly reduced. This means that the multi-pitch tracking results are more reliable on real music pieces than random note pieces. But still, utilizing score information improves source separation results.

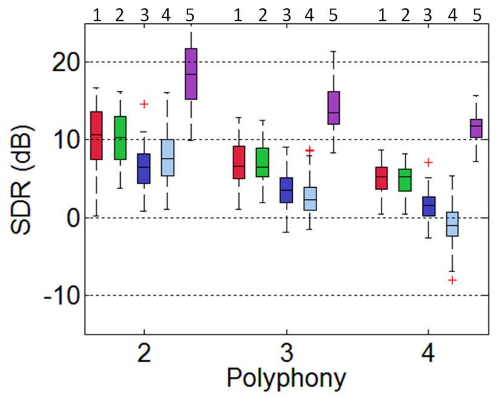


Fig. 8. SDR versus polyphony on the Bach chorale dataset for Soundprism (1, red), Ideally-aligned (2, green), Ganseman10 (3, blue), MPET (4, cyan) and Oracle (5, purple). Each box of polyphony 2, 3 and 4 represents $2 \times 60 = 120$, $3 \times 40 = 120$ and $4 \times 10 = 40$ data points respectively, each of which corresponds to one instrumental melodic line in a musical piece.

Figure 8 shows results for different polyphony. We can see that Soundprism and Ideally-aligned obtain very similar results for all polyphony. This suggests that the score following stage performs well enough for the separation task on this dataset. In addition, Soundprism obtains a significantly higher SDR than Ganseman10 for all polyphony ($p < 10^{-7}$). Furthermore, MPET degrades much faster than the three score-informed separation systems, which again indicates that score information is more helpful in the pieces with higher polyphony.

The SDR of polyphony 4 showed in Figure 8 are calculated from all tracks of all quartets. However, for the same piece of a quartet, different instrumental tracks have different SDRs. A reasonable hypothesis is that high frequency tracks have lower SDR since they have more harmonics overlapped by other sources. However, Figure 9 shows opposite results. It can be seen that Track 1, 2 and 3 have similar SDRs, but Track 4 has a much lower SDR. This may suggest that the energy distribution strategy used in Section III-C biases to the higher-pitched source.

F. Commercially recorded music examples

We test Soundprism and its comparison systems on two commercial recordings of music pieces from the RWC database [39]. These pieces were not mixed from individual tracks, but recorded directly as a whole from an acoustic environment. Therefore, we do not have the ground-truth sources and alignments, hence cannot calculate measures. The separated sources of these pieces can be downloaded from http://www.cs.northwestern.edu/~zdu459/jstsp2011/examples. This webpage also contains several examples from the Bach chorale dataset.

VI. Conclusion

In this paper we propose Soundprism, an online system for score-informed source separation of polyphonic music with harmonic sources. We decompose the system into two stages: score following and source separation. For the first stage, we use a hidden Markov process to model the audio performance. The state space is defined as a 2-d space of score position and tempo. The observation model is defined as the multi-pitch likelihood of each frame, i.e. the likelihood of seeing the audio frame given the pitches at the aligned score position. Particle filtering is employed to infer the score position and tempo of each audio frame in an online fashion. For the second stage, we first refine the score-informed pitches. Then sources are separated by time-frequency masking. Overlapping harmonics are resolved by assigning the mixture energy to each overlapping source in reverse proportion to the square of their harmonic numbers.

Experiments on both synthetic audio and real music performances show that Soundprism can deal with multi-instrument music with high polyphony and some degree of tempo variation. As a key component of Soundprism, the score follower performs better when the polyphony increases from 2 to 6. However, the score following results degrade significantly when the tempo variation of the performance increases.

For future work, we want to incorporate some onset-like features in the observation model of the score follower, to improve the alignment accuracy. In addition, a more advanced method to resolve overlapping harmonics should be used to improve the source separation results. For example, we can learn and update a harmonic structure for each source and use this harmonic structure to guide the separation of overlapping harmonics. Furthermore, we also want to improve the robustness of Soundprism, to deal with the situation that performers occasionally make mistakes and deviate from the score.

VII. Acknowledgement

We thank Joachim Ganseman for providing the code for his score-informed separation algorithm, Arshia Cont for providing code to evaluate the alignment rate measure, and Craig Sapp for providing the reference to the open source software Scorealign. We also thank the reviewers for their thorough comments. This work was supported by National Science Foundation awards 0643752 and 0812314.

References

[1] Z. Duan, Y. Zhang, C. Zhang and Z. Shi, "Unsupervised single-channel music source separation by average harmonic structure modeling," *IEEE Trans. Audio Speech and Lang. Process.*, vol. 16, no. 4, pp. 766-778, 2008.

[2] C. Raphael, "A classifier-based approach to score-guided source separation of musical audio," *Computer Music Journal*, vol. 32, no. 1, pp. 51-59, 2008.

[3] R. Hennequin, B. David and R. Badeau, "Score informed audio source separation using a parametric model of non-negative spectrogram", in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011.

[4] J. Woodruff, B. Pardo and R.B. Dannenberg, "Remixing stereo music with score-informed source separation," in *Proc. International Conference on Music Information Retrieval (ISMIR)*, 2006.

[5] J. Ganseman, G. Mysore, P. Scheunders and J. Abel, "Source separation by score synthesis," in *Proc. International Computer Music Conference (ICMC)*, New York, NY, June 2010.

[6] 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.

[7] R. Macrae and S. Dixon, "Accurate real-time windowed time warping," in *Proc. International Society for Music Information Retrieval Conference (ISMIR)*, 2010, pp. 423-428.

[8] Z. Duan, B. Pardo and C. Zhang, "Multiple fundamental frequency estimation by modeling spectral peaks and non-peak regions," *IEEE Trans. Audio Speech and Lang. Process.*, vol. 18, no. 8, pp. 2121-2133, 2010.

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

[10] N. Hu, R.B. Dannenberg and G. Tzanetakis, "Polyphonic audio matching and alignment for music retrieval," in *Proc. 2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, New York, USA, 2003, pp. 185-188.

[11] S. Ewert, M. Müller, and P. Grosche, "High resolution audio synchronization using chroma onset features," in *Proc. IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pp. 1869-1872, 2009.

[12] P. Cano, A. Loscos, and J. Bonada, "Score-performance matching using HMMs," in *Proc. International Computer Music Conference (ICMC)*, 1999.

[13] C. Raphael, "Automatic segmentation of acoustic musical signals using hidden markov models," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21, no. 4, pp. 360-370, 1999.

[14] C. Raphael, "Aligning music audio with symbolic scores using a hybrid graphical model," *Machine Learning*, vol. 65, pp. 389-409, 2006.

[15] C. Joder, S. Essid and G. Richard, "A conditional random field framework for robust and scalable audio-to-score matching," *IEEE Trans. Speech, Audio and Lang. Process.*, in press.

[16] R.B. Dannenberg, "An on-line algorithm for real-time accompaniment," in *Proc. International Computer Music Conference (ICMC)*, pp. 193-198, 1984.

[17] B. Vercoe, "The synthetic performer in the context of live performance," in *Proc. International Computer Music Conference (ICMC)*, pp. 199-200, 1984.

[18] M. Puckette, "Score following using the sung voice," in *International Computer Music Conference (ICMC)*, 1995.

[19] L. Grubb and R.B. Dannenberg, "A stochastic method of tracking a vocal performer," in *Proc. International Computer Music Conference (ICMC)*, 1997.

[20] N. Orio and F. Dechelle, "Score following using spectral analysis and hidden markov models," in *Proc. International Computer Music Conference (ICMC)*, 2001.

[21] C. Raphael, "A Bayesian network for real-time musical accompaniment," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2001.

[22] L. Grubb and R.B. Dannenberg, "Automated accompaniment of musical ensembles," in *Proc. the Twelfth National Conference on Artificial Intelligence (AAAI)*, 1994, pp. 94-99.

[23] A. Cont, "Realtime audio to score alignment for polyphonic music instruments using sparse non-negative constraints and hierarchical HMMs," in *Proc. IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, 2006.

[24] A. Cont, "A coupled duration-focused architecture for real-time music-to-score alignment," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 32, no. 6, pp. 974-987, June, 2010.

[25] G. E. Poliner and D. P. W. Ellis, "A discriminative model for polyphonic piano transcription," *EURASIP Journal on Advances in Signal Processing*, vol. 2007, Article ID 48317, 9 pages.

[26] A. Klapuri, "Multiple fundamental frequency estimation based on harmonicity and spectral smoothness," *IEEE Trans. Speech and Audio Processing*, vol. 11, no. 6, pp. 804-815, 2003.

[27] A. Doucet, N. de Freitas and N.J. Gordon, *Sequential Monte Carlo methods in practice*, Springer-Verlag, New York, 2001.

[28] M.S. Arulampalam, S. Maskell, N. Gordon and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 174-188, 2002.

[29] Y. Li, J. Woodruff and D.L. Wang. "Monaural musical sound separation based on pitch and common amplitude modulation," *IEEE Trans. Audio Speech and Lang. Process.*, vol. 17, no. 7, pp. 1361-1371, 2009.

[30] M. Every and J. Szymanski, "A spectral-filtering approach to music signal separation," in *Proc. International Conference on Digital Audio Effects (DAFx)*, 2004

[31] T. Virtanen, "Algorithm for the separation of harmonic sounds with time-frequency smoothness constraint," in *Proc. International Conference on Digital Audio Effects (DAFx)*, 2003.

[32] H. Viste and G. Evangelista, "A method for separation of overlapping partials based on similarity of temporal envelopes in multi-channel mixtures," *IEEE Trans. Audio Speech and Lang. Process.*, vol. 14, no. 3, pp. 1051-1061, 2006.

[33] C. Yeh, A. Roebel and X. Rodet, "Multiple fundamental frequency estimation and polyphony inference of polyphonic music signals," *IEEE Trans. Audio Speech and Lang. Process.*, vol. 18, no. 6, pp. 1116-1126, 2010.

[34] J. Ganseman, P. Scheunders, G.J. Mysore and J. S. Abel, "Evaluation of a score-informed source separation system," in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2010.

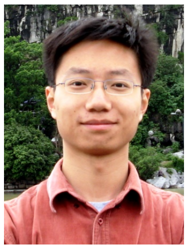
[35] Z. Duan, J. Han and B. Pardo, "Song-level multi-pitch tracking by heavily constrained clustering," in Proc. *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, 2010, pp. 57-60.

[36] E. Vincent, R. Gribonval and C. Févotte, "Performance measurement in blind audio source separation," *IEEE Trans. Audio Speech and Lang. Processing*, vol. 14, no. 4, pp. 1462-1469, 2006.

[37] E. Vincent, R. Gribonval and M.D. Plumbley, *BSS Oracle Toolbox Version 2.1*, http://bass-db.gforge.inria.fr/bssoracle/

[38] A. Cont, D. Schwarz, N. Schnell and C. Raphael, “Evaluation of real-time audio-to-score alignment,” in *Proc. International Conference on Music Information Retrieval (ISMIR)*, 2007.

[39] M. Goto, H. Hashiguchi, T. Nishimura and R. Oka, “RWC music database: popular, classical, and jazz music databases,” in *Proc. International Conference on Music Information Retrieval (ISMIR)*, 2002, pp. 287-288.



Zhiyao Duan (S’09) was born in Henan, China, in 1983. He received the B.E. and M.S. degrees from Tsinghua University, Beijing, China, in 2004 and 2008, respectively. He is currently pursuing the Ph.D. degree in the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL. His research interests lie primarily in the interdisciplinary area of signal processing and machine learning toward audio information retrieval applications, including source separation, multi-pitch estimation and tracking, audio-score alignment, etc.

Bryan Pardo (M’07) received the M.Mus. degree in jazz studies in and the Ph.D. degree in computer science in from the University of Michigan, Ann Arbor, in 2001 and 2005, respectively.

He is an Associate Professor in the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, with appointments in the Music Cognition Program and the Center for Technology and Social Behavior. He has developed speech software for the Speech and Hearing, The Ohio State University, statistical software for SPSS and worked as a machine learning Researcher for General Dynamics. While finishing his Ph.D. degree, he taught in the Music Department of Madonna University. When he’s not programming, writing, or teaching, he performs throughout the United States on saxophone and clarinet at venues such as Albion College, the Chicago Cultural Center, the Detroit Concert of Colors, Bloomington Indiana’s Lotus Festival, and Tucson’s Rialto Theatre.