Multimed Tools Appl (2019) 78:2017–2044
https://doi.org/10.1007/s11042-018-6349-y

An effective method for audio-to-score alignment using onsets and modified constant Q spectra

Chunta Chen[1] · Jyh-Shing Roger Jang[2]

Received: 28 August 2017 / Revised: 7 June 2018 / Accepted: 29 June 2018 /
Published online: 5 July 2018

Abstract This paper proposes an effective algorithm for polyphonic audio-to-score alignment that aligns a polyphonic music performance to its corresponding score. The proposed framework consists of three steps: onset detection, note matching, and dynamic programming. In the first step, onsets are detected and then onset features are extracted by applying the constant Q transform around each onset. A similarity matrix is computed using a note-matching function to evaluate the similarity between concurrent notes in the music score and onsets in the audio recording. Finally, dynamic programming is used to extract the optimal alignment path in the similarity matrix. We compared five onset detectors and three spectrum difference vectors at selected audio onsets. The experimental results revealed that our method achieved higher precision than did the other algorithms included for comparison. This paper also proposes an online approach based on onset detection that can detect most notes within only 10 ms. Based on our experimental results, this online approach outperforms all methods included for comparison when the tolerance window is 50 ms.

Keywords Music synchronization · Audio-to-score alignment · Audio onset detection · Score following

1 Introduction

Music synchronization or alignment in various forms has great importance for analyzing and processing music. According to music data format (audio recordings or symbolic representations), music synchronization tasks can be divided into three main categories: audio-to-audio,

✉ Chunta Chen
chun-ta.chen@mirlab.org

Jyh-Shing Roger Jang
jang@mirlab.org

[1] Computer Science Department, National Tsing Hua University, Hsinchu City, Taiwan

[2] Computer Science Department, National Taiwan University, Taipei City, Taiwan

symbolic-to-symbolic, and audio-to-symbolic [28]. Audio-to-audio alignment [42] matches a given position in a music performance to the corresponding position in a separate audio recording. Symbolic-to-symbolic alignment [9] seeks to indicate how each note in a piece of music is matched to its corresponding note in the score according to the correlation between their musical structures. Audio-to-score alignment, as its name implies, seeks to find a mapping between an audio performance and its corresponding symbolic musical score [27, 31]; in other words, the alignment system analyzes the content of the input audio at a given time point (i.e., an onset) and maps it to a corresponding time point on the score with a similar musical structure (i.e., note pitch and note duration).

This paper focuses on audio-to-score alignment in polyphonic music, including audio music with multiple instruments and audio music with unconstrained instruments (in contrast to piano-only music in some previous studies). Generally, audio-to-score alignment can be implemented online or offline. An online system, also known as a score follower, can be used to align a music score to a live performance and can participate in musical interactions with human musicians in real time for tasks such as automatic page turning [1] and computer-aided accompaniment [34]. In addition, because score followers can link note events in a score with time corresponding points in an audio performance, it plays an important role in score-informed source separation tasks [7, 18]. By contrast, offline systems have no real-time constraints, and thus the complete information of a given audio signal can be used during the alignment process. Therefore, offline alignment usually outperforms its online counterpart. An offline alignment system can be used to retrieve the closest matching MIDI document from a database for an audio query [22] and enables the indexing of timestamps of an audio recording according to the desired measure or passage in a music score [14]. Moreover, such an alignment system provides information on performance errors and temporal deviations of note events and thus can be used for music performance evaluation or music education.

Music alignment suffers from some well-known inherent difficulties [13, 27, 28] related to possible mismatches between human performances and their corresponding scores. Live music performance invariably entails certain deviations, intentional or not, from the score; for example, musicians may seek to enrich their performance by adding ornaments or chord variations that are not specified in the score. Another notable aspect is temporal deviation; for example, musicians may seek to express themselves through temporal inconsistencies such as variation in note onset note duration or tempo. Alignment is further complicated when dealing with polyphonic music, where multiple note events occurring simultaneously can lead to interference between harmonic series of notes, thereby increasing the difficulty of identifying the relation between an audio segment and its corresponding score.

The standard method of audio-to-score alignment consists of three steps, namely feature extraction, similarity calculation, and alignment. In step 1, informative features are extracted from an audio signal to characterize the musical content, such as onset and pitch or chord. Step 2 defines a similarity function to measure the difference between features of an audio recording and note events in the score. Step 3 uses an alignment algorithm to determine the closest match between feature sequences and note events. Most polyphonic audio-to-score alignment algorithms adopt a similar procedure to convert a symbolic score into an audio file, perform feature extraction on the input audio recording and the converted audio piece, and then align two sequences of features. In [30], dynamic time warping (DTW) was used to align a polyphonic audio file to another audio file synthesized from a MIDI file. The peak structure distance, which was derived from the spectra of the audio files, was used to measure the difference between an audio performance and a synthesized audio specimen. A similar scheme was proposed in [14, 22] for computation of a simple 12-dimensional pitch chroma as an input feature and alignment of the performance audio with the synthesized audio by using the DTW approach. In [32], an exhaustive search over DTW score normalization techniques was performed to identify the optimal method for reporting a reliable alignment confidence score. Such dynamic-programming-based algorithms can cope with tempo fluctuation and performance errors in musical signals. Therefore, the DTW algorithm and variants thereof are extensively used in audio-to-score alignment systems.

Other audio-to-score alignment systems use probabilistic models to enhance performance, such as hidden Markov models (HMMs) and conditional random field (CRF) models. These models account for matching uncertainty to enhance performance. In such systems, the hidden variable represents the current position in the score. Moreover, flexible transition probabilities permit structural changes. The Viterbi algorithm is usually used to identify the optimal alignment path in these models. For example, in [11], the hierarchical HMM was used in a polyphonic-score-following system, with previously learned pitch templates employed for multiple fundamental frequency matching. The CRF model has also been applied to the audio-to-score alignment problem. In [23], the authors incorporated frame-level and segment-level features, including chroma, onset and tempo information. Three CRF models have been proposed for various degrees of trade-off between accuracy and complexity. These probabilistic models generated efficient and accurate alignment techniques that can compete favorably with DTW-based methods. More recently, a multimodal convolutional neural network for audio-to-sheet matching [17] was proposed for offline applications; the network matches short music audio snippets to their corresponding pixel locations in images of piano score.

Some drawbacks of the methods proposed in the literature are summarized as follows.

a. Most audio-to-score studies [8, 11, 14, 22, 23, 30, 32, 33] have used synthesized audio (from MIDI files or music scores) for their experiments. The major disadvantage of using synthesized audio for audio-to-score experiments is that the musical properties of synthesized audio are likely to be different from those of real-world musical audio signals. For example, input audio clips for alignment do not always have the same timbre and harmonic properties as synthesized ones.
b. Some studies [11, 23, 24] have adopted probabilistic models to implement the alignment process, and these models usually require time-consuming training to fine tune their parameters.

This paper proposes an audio-to-score alignment framework that is effective for audio music with unconstrained instruments or multiple instruments. Such audio music poses a greater challenge for audio-to-score alignment. Recent studies on music with unconstrained or multiple instruments are represented by [8, 10, 33]. Carabias-Orti et al. [8] used nonnegative matrix factorization (NMF) to decompose the spectral patterns of an input performance and produce a divergence matrix; DTW was then used to find the minimum-cost path within the matrix. The online version [33] of this method was subsequently submitted for Music Information Retrieval Evaluation eXchange (MIREX)[1] task; the results demonstrated that this method outperformed all the algorithms submitted in previous years. Chen et al. [10] used

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

NMF to decompose harmonics around onsets of the constant Q spectrum and computed the degree of similarity with notes in the corresponding score, thereby generating robust results for aligning audio music with various instruments. This study compares our proposed method with these methods in an experiment.

This paper presents an audio-to-score alignment framework with onset detection and note matching. As shown in Fig. 1, the proposed alignment method involves three steps (onset segmentation, note matching, and dynamic programming). A piece of audio music is input to the onset detection system. In the note-matching block, a feature vector around each estimated onset is compared with the notes in the corresponding music score. Finally, dynamic programming is used to extract the best alignment path in the similarity matrix. By performing onset detection first, we were able to simplify the alignment process considerably. Moreover, consideration of the spectral variation around an onset can reduce the complexity of similarity computation and interference from vibrato. In addition, this paper proposes a method for converting a constant Q spectrogram into another format similar to a piano roll. This conversion does not require synthesis from MIDI to audio and is less likely to be influenced by instrumental timbre. This paper discusses the effects of various onset detectors and types of spectrum difference vectors in the experiments. The proposed algorithm outperforms the aforementioned methods. Furthermore, we propose an online audio-to-score alignment method based on onset detection, the feasibility and performance of which are experimentally validated. Moreover, we propose a method for onset detector fusion. We extract features from various onset detection functions (ODFs) and then select more reliable onsets by using a prelearned classifier. The experimental results verify that fusion is effective and outperforms the constituted ODFs.

The remainder of this paper is organized as follows. Section 2 introduces the onset detection methods used in our fusion method, which is described in Section 3. Section 4 presents a note-

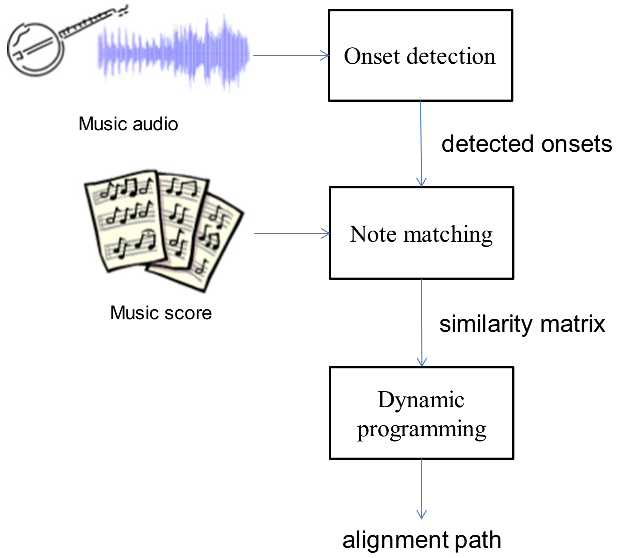


Fig. 1 Flowchart of the proposed audio-to-score alignment algorithm. A piece of audio music audio is inputted into onset detection system. In the note matching block, a feature vector around each estimated onset is compared with the notes in the music score. Finally, dynamic programming is used to extract the best alignment path from the similarity matrix

matching algorithm used to measure the similarity between a spectrum difference vector derived from the audio onset and a score pitch vector derived from the concurrent note-on event in the score. Section 5 explains our dynamic programming algorithm. In Section 6, we introduce an online alignment method based on onset detection. Section 7 describes our experimental setup and offers a comparison with other methods, alongside detailed analysis and discussion. Finally, Section 8 summarizes our findings and provides directions for future research.

2 Choice of onset detection methods

Onset detection [2, 16] is a crucial preprocessing step toward high-level music processing that involves beat tracking, song segmentation, and music transcription. An onset is usually defined as the starting point of a sound event. Onset detection typically applies an onset detection function (ODF) to the input audio music to generate an ODF curve that indicates onset strength. The local maxima can then be selected from the curve as the onsets because the ODF should have a larger value at the vicinity of an onset event. Different low-level features of audio music can be considered for computing the ODF, including the magnitude spectrum, the phase spectrum, and pitch. For example, spectral flux computes the temporal differences of magnitude spectra, and phase deviation estimates the change in the instantaneous frequency in a short-time Fourier transform frequency bin by computing the second-order temporal differences of phase angles, whereas the complex domain simultaneously considers differences in magnitude and phase spectra.

To construct general detection functions that can detect onsets in a wider range of audio music, machine-learning-based methods have gained popularity in recent years. In [25], onset detection algorithms based on neural networks were first used to classify a spectrogram of a frame as an onset or not. All winning algorithms of the MIREX audio onset detection task in the preceding few years are based on a probabilistic model consisting of a recurrent neural network (RNN) [20] and convolutional neural network (CNN) [36, 37]. Their resultant onset detectors exhibit good performance for almost all types of music.

Another promising direction for onset detection is the fusion of multiple detection methods [15]. Duxbury et al. [19] proposed an approach combining an amplitude-based detection method with a phase-based one. Holzapfel et al. [21] considered pitch, energy, and phase information in parallel for the detection of pitched onsets. Tian et al. [40] investigated multiple fusion strategies to combine existing onset detectors and provided a systematic evaluation of all tested combinations.

This paper proposes a new fusion strategy for ODF curves. For a given onset, the corresponding peak positions of different ODF curves may have some temporal deviation. In addition, some ODF curves may have no peak, and the onset strengths of different ODF curves may not be the same. Consequently, the fusion strategy in [19, 21, 40] is not desirable because the weightings of ODF curves are difficult to estimate. Our approach divides the peaks of ODF curves into groups that are likely to coincide with an onset. We then extract features from each group and train a classifier to determine whether a peak group corresponds to an onset.

In our implementation, we combine the ODFs of three onset detectors. The properties of these detection methods are described as follows.

2.1 SuperFlux

SuperFlux is an onset detection algorithm proposed in [3] that improves onset detection accuracy in the presence of vibrato. SuperFlux is an enhanced version of common spectral flux. It calculates the difference between two near short-time spectra and can suppress false positive detections of music with vibrato by tracking spectral trajectories using a maximum filter.

2.2 ComplexFlux

SuperFlux considers only the magnitude spectrogram without incorporating any phase information. ComplexFlux [4] is derived from SuperFlux and considers local group delay to deal with vibrato and tremolo. By incorporating phase information, ComplexFlux can determine steady tones and accordingly suppress spurious loudness variation. SuperFlux and ComplexFlux are the current state-of-the-art nonprobabilistic onset detection methods according to the results of the MIREX onset detection task from the preceding few years.

2.3 RNN

The recurrent neural network (RNN)-based method [20] incorporates an RNN alongside the spectral magnitude and its first time derivative as input features. Because of its bidirectional architecture, this method can model the context of an onset to detect barely discernible onsets in complex mixes and suppress events that are erroneously considered onsets by other algorithms.

3 Fusion method for onset detection

The fusion of multiple ODFs is based on the concept that different ODFs have different characteristics that, if used properly, can be synergized to capture onsets of audio music in various styles and genres. In particular, spectral flux is known to be sensitive to percussive signals, whereas SuperFlux and ComplexFlux can robustly suppress vibrato while detecting onsets. Moreover, ODFs of a shorter frame size are better at detecting onsets in weak and transient signals, whereas those of a longer frame size better capture soft onsets. The aforementioned observations indicate that a variety of ODF curves should be employed to capture different types of onsets.

In practice, different ODFs have different characteristics, and thus determining a universal criterion for selecting peaks as onset candidates is difficult. To automate this tuning process, we form peak groups and use a support vector machine (SVM) to determine whether a given peak group is a true onset. The use of multiple ODF curves alongside simple training of the SVM is the core concept of the proposed method.

Figure 2 shows a block diagram of a simple example (used in our experiment) of the proposed method. In this specific implementation, three onset detectors are used: the RNN method, SuperFlux, and ComplexFlux. Peak values of ODFs are cascaded with music harmonic/percussive (H/P) features from harmonic/percussive source separation of the input audio. These cascaded feature vectors are then classified by the SVM to predict onsets.

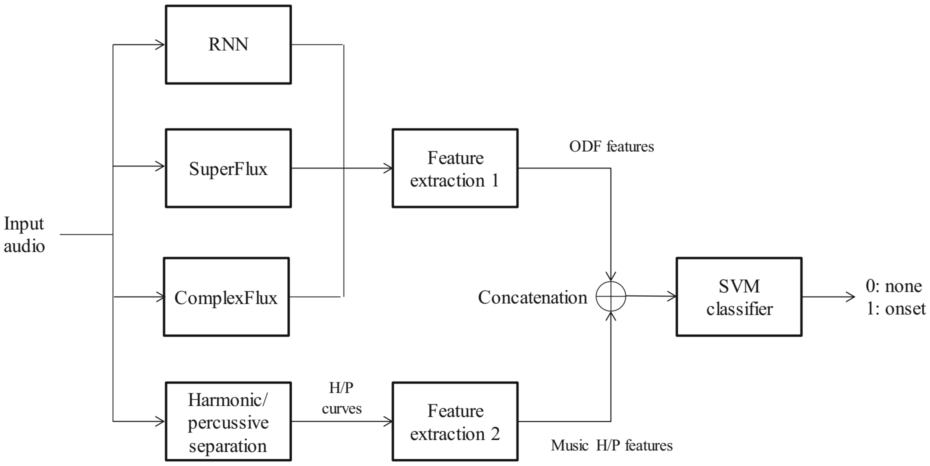


Fig. 2 Block diagram of a specific implementation of the proposed method: Peak values of ODFs of RNN, SuperFlux and ComplexFlux are cascaded with music features from harmonic/percussive source separation of the input audio. These feature vectors are then classified by SVM to predict onsets

3.1 Onset detection functions and peak selection

To select the ODF curve peaks, the peak-picking method described in [3] with nearly identical parameter settings is employed. Specifically, an ODF curve at index n, denoted $ODF[n]$, is selected as a peak if it fulfills these three requirements:

1. $$ODF[n] = max(ODF[n - w_1 : n + w_2])$$
2. $$ODF[n] \geq mean(ODF[n - w_3 : n + w_4]) + \theta$$
3. $$n - n_{previous\ onset} > w_5$$

The first requirement is to identify the local maximum, where w_1 and w_2 determine the size of the window within which the local maximum resides. The position of the local maximum within the window represents a possible onset. The second requirement is to filter out lower-intensity peaks, which are considered noise. Our approach to fulfilling this requirement is to select the local maximum that exceeds the local mean by a threshold θ. The parameters w_2 and w_4 determine the window size for determining the local mean. The third requirement is to set the minimum interval between adjacent onsets, which is determined by w_5. The parameter values in these three conditions are equal to the default parameter values of the program provided in [3, 4], except for the threshold parameter θ, which can be used to control the number of identified peaks. A lower value of θ results in more peaks being identified and thus more false positives toward final onset detection. By contrast, a higher value of θ may lead to more false negatives because some true onsets are not identified. In our experiment, we selected a median low value of θ to enable selection of more peaks and prevent rejection of any true onsets. An SVM classifier is used to make a final decision regarding whether selected peaks are onsets.

3.2 Peak grouping

After local peaks are identified from each ODF curve, peaks from all ODF curves must be grouped based on their temporal vicinity such that a final classifier can be used to determine whether a peak group is an onset. Let *MS* and *NS* be the time vectors of the peak locations given by two ODF curves, and let δ be the tolerance window (50 ms in our experiments). The peak grouping procedure described in Algorithm 1 is used to generate peak groups from the two ODF curves.

Algorithm 1: Peak grouping

```
1: procedure PEAKGROUP($MS$ , $NS$, $\delta$)
2:     $i_1, i_2 \leftarrow 0, 0$
3:     $TS \leftarrow$ empty list
4:     while $i_1 < len(MS)$ and $i_2 < len(NS)$ do
5:         if abs($MS[i_1] - NS[i_2]) \leq \delta$ then
6:             insert: $TS \leftarrow\ i_1, i_2)$
7:             $i_1, i_2 \leftarrow i_1+1, i_2+1$
8:         elseif $MS[i_1] < NS[i_2]$ then
9:             $i_1 \leftarrow i_1+1$
10:        elseif $MS[i_1] > NS[i_2]$ then
11:            $i_2 \leftarrow i_2+1$
12:    return $TS$
```

Notably, in our implementation, we used the average time of the grouped peaks as a reference vector of onset time.

3.3 Feature extraction

After the peak groups have been obtained, feature extraction must be performed on the groups; the features are then sent for SVM classification to determine whether a given peak group represents an onset.

3.3.1 ODF features

Peaks in a group are somewhat related and usually describe the same onset event in audio music. For each peak in a SuperFlux or ComplexFlux ODF, two features are extracted based on the ODF's value and local mean. These features are specified as follows (*ODF*[*n*] is a peak at frame *n* of its corresponding ODF):

- $$ODF[n]$$
- $$ODF[n] - mean(ODF[n - w_3 : n + w_4])$$

Because a peak in an ODF represents a high probability of there being an onset at the instant represented by the peak, the peak value *ODF*[*n*] is adopted as feature 1. Feature 2 represents the contrast between the peak and its local average; a higher contrast indicates a higher probability of the peak being an onset. In addition, two ODFs with different frame sizes are generated. The ODF with the shorter frame size is expected to capture more peaks as onset candidates, including some false positives. The ODF with the longer frame size is a smoother curve that can better capture soft onsets. These two types of frames can leverage the strengths of both situations and contribute to final onset detection in a complementary manner. Two frame sizes, 46 and 92 ms, were adopted in our implementation.

For each peak in the RNN ODF, only the peak value is used as the feature because the RNN model in [20] considers multiple frame size resolutions. Therefore, each peak group has nine features. If some ODFs in a peak group have no peaks, the default values of the corresponding features are set to zero. The estimated onset time is the average of all peaks in the group if the peak group is judged to be an onset.

3.3.2 Music H/P features

Musical signals typically consist of harmonic and percussive spectral components. Harmonic sound pertains to pitched instruments, which have a spectral structure that is horizontally smooth in time, whereas percussive sound pertains to drum-type instruments, which have a spectral structure that is vertically smooth in frequency. The goal of harmonic/percussive source separation [29, 39] is to decompose a given audio signal into two harmonic sounds and percussive sounds. Separating these two components is useful for some music information retrieval tasks, such as melody extraction [35] and chord recognition [41].

Onset detection is particularly vulnerable to vibrato interference [3], which tends to produce false positives. Separating musical signals into harmonic and percussive components can reduce interference from vibrato. The spectrogram of an input audio specimen is separated into a harmonic matrix *H* and percussive matrix *P* to obtain two curves, namely *Hv* and *Pv*, respectively, by summing *H* and *P* along the frequency axis. These two curves are normalized to have maxima of 1. Figure 3 presents examples of harmonic and percussive curves. Figure 3a shows the spectrogram of an audio excerpt of a violin, and Fig. 3b shows the harmonic curve (green line) and percussive curve (blue line) of said excerpt. The black circles on the x-axis represent the ground truth of the onsets. The estimated onsets identified using the SuperFlux method are with red asterisks. The figure shows that *Pv* has a relatively strong rising trend at some onsets, such as 1.6, 2.2, 2.4, and 2.8 s. At the onsets at 3.3 and 4.8 s, *Pv* has a relatively low degree of rising strength, but the relative amplitude between *Pv* and *Hv* is considerably lower than the cases of the insertions. By contrast, the *Pv* increase at the insertions at 0.5, 0.6, 4.0, and 4.1 s is very minor, and the gaps between *Pv* and *Hv* are relatively large. The relative amplitude between these two curves helps to determine whether an onset is present; for example, *Pv* is usually close to or greater than *Hv* around an onset. In addition, *Hv* is larger than *Pv* in a vibrato segment.

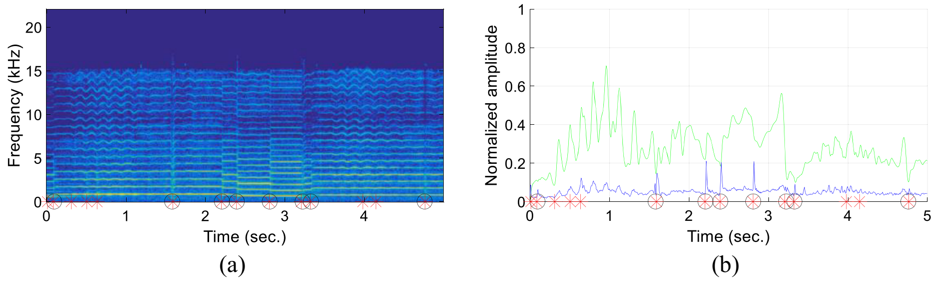


Fig. 3 **a** Spectrogram of an audio excerpt of a violin. **b**Harmonic curve (green line) and percussive curve (blue line) of the audio excerpt. The black circles on the x-axis represent the groundtruth of onsets. The estimated onsets of SuperFlux method are labeled as red asterisks

Assume n_a is the frame index of the average time of a peak group. We can then define 9 features extracted from Hv and Pv for each peak group as follows:

1. $Hv[n_a]$
2. $Pv[n_a]$
3. $\frac{Hv[n_h]-Hv[n_a]}{n_h-n_a}$, where $n_h = \underset{n_a \leq n \leq n_a+\Delta n_1}{\arg\max} \; Hv[n]$
4. $\frac{Pv[n_p]-Pv[n_a]}{n_p-n_a}$, where $n_p = \underset{n_a \leq n \leq n_a+\Delta n_1}{\arg\max} \; Pv[n]$
5. $Hv[n_a]-mean(Hv[n_a-\Delta n_0 : n_a])$
6. $Pv[n_a]-mean(Pv[n_a-\Delta n_0 : n_a])$
7. $Hv[n_h]-mean(Hv[n_a-\Delta n_0 : n_a])$
8. $Pv[n_p]-mean(Pv[n_a-\Delta n_0 : n_a])$
9. $Hv[n_h]-Pv[n_a]$

The aforementioned features represent changes in Hv and Pv around n_a. The first two features are the values of Hv and Pv for onset candidate n_a. Features 3 and 4 are slopes of Hv and Pv over the intervals $[n_a, n_p]$ and $[n_a, n_h]$, respectively, where n_h and n_p are the indices of local maxima in the range $[n_a, n_a+\Delta n_1]$ of Hv and Pv, respectively. Features 5 to 8 are the differences between Hv and Pv at various locations and their averages in the neighborhood. The final feature is the amplitude difference between Hv and Pv. These harmonic and percussive features are cascaded with their ODF features to form 18 features for each peak group; these features are then divided into two categories of "onset" and "not onset" by an SVM classifier, which is described as follows.

3.4 Final classifier

An SVM is used as the final classifier to determine whether a given peak group corresponds to an onset. SVMs have been proven effective classifiers because they attempt to maximize the margin or distance of closest examples from opposite classes from the decision hyperplane [38]. Regarding the dimensionality of the SVM input, we define nine features extracted from ODFs and nine features extracted from Hv and Pv in each peak group. Therefore, the overall dimension of the input to the SVM is 18. In our experiments, we used a Gaussian radial basis function kernel, and the parameters were optimized through a grid search of cross validation within the training set.

4 Similarity measure

After the onsets have been obtained, the note events in the score that trigger an onset according to the spectrum of constant Q transform must be determined. We have developed three methods for evaluating the likelihood of notes in the score inducing spectral variation near an onset.

4.1 Modified constant Q spectrum

We define a modified constant Q spectrum (MCQS) for note matching in audio-to-score alignment. The constant Q transform [6] is used extensively in music analysis; "Q" refers to the "quality factor," namely the ratio of the center frequency to its bandwidth. Thus, a constant Q indicates that the ratio is constant for all frequency bins in the spectrum. The transform reflects the human auditory system, where the spectral resolution is higher at lower frequencies and the temporal resolution is higher at higher frequencies. The frequency scale of the constant Q transform is logarithmic, and thus this method is particularly useful in music processing because musical transposition corresponds only to the translation of frequency bins. The Hamming window is used in our implementation. We select the Q factor in such a manner that 12 frequency bins exist over an octave, with each bin directly corresponding to a musical note.

The constant Q spectrum is further modified by retaining only the local maximum in the pitch axis of the spectrogram. In addition, we limit the pitch range to that of the score. The advantage of this modification is that it is relatively unaffected by instrumental timbre and can reduce overtone interference.

Figure 4 shows an example of the MCQS of an audio excerpt and its corresponding score. This excerpt is from the Bach10 dataset and contains four parts played by different instruments. Figure 4a shows the MCQS of the excerpt. Each dashed vertical line represents an onset time; these lines are labeled alphabetically at the top. Each horizontal bar is equivalent to a fundamental frequency or an overtone of a tone in the audio excerpt. Figure 4b is the piano-roll plot of the corresponding music score. Each horizontal red bar represents a note, with its height denoting the note pitch and its length denoting the note duration. Each dashed vertical line is the time point of note-on events; these lines are labeled alphabetically at the top. The same letters in Fig. 4a and b represent a time mapping in audio-to-score alignment; that is, timings A–U in Fig. 4a correspond to the same letters in Fig. 4b. In this example, the note duration and note onset of the performance deviates considerably from the score. The durations are 11 and 10 s in Fig. 4a and b, respectively, indicating that the speed of the performance is relatively slow. In addition, because of temporal deviation in the human performance, the time interval of the label is not as stable as that in the score. Despite these minor inconsistences, however, the musical structures are very similar at all corresponding points labeled alphabetically in the figure. Our proposed method for establishing their note correspondences is described in the following section.

4.2 Note-matching method

When computing similarity, the influence of overtones must be considered, as must differences in energy distribution in the MCQS at different pitches. In our method, for each onset, the

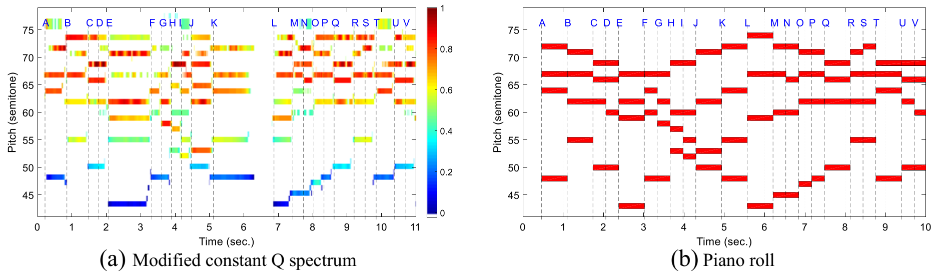


Fig. 4 **a** Modified constant Q spectrum of an audio excerpt. **b** Piano-roll plot of its corresponding score. Each dashed vertical line means an onset time and is labeled alphabetically at the top

frames immediately before and after the onset in the MCQS are taken. The number of frames used cannot exceed the onset interval or the given threshold (e.g., 200 ms). Two vectors are obtained by taking the averages of the magnitude spectrum immediately before and after an onset i. Assuming that dA_i is the difference between these two vectors, three types of spectrum difference vectors (SDVs) ψ_i can be defined, as explained subsequently.

a. For the first type, we only consider the difference if it is higher than a noise threshold:

$$\psi_i^1(k) = \begin{cases} dA_i(k), & \textit{if } dA_i(k) > \epsilon_i \\ 0, & \textit{otherwise} \end{cases} \tag{1}$$

where ϵ_i is a noise margin, which is set to 1/20 of the maximum of dA_i.

b. The second type converts a vector dA_i into a binary vector, in which the values greater than noise threshold are set to 1.

$$\psi_i^2(k) = \begin{cases} 1, & \textit{if } dA_i(k) > \epsilon_i \\ 0, & \textit{otherwise} \end{cases} \tag{2}$$

c. The third type takes the logarithm to narrow the numerical range.

$$\psi_i^3(k) = \begin{cases} \log(dA_i(k)), & \textit{if } dA_i(k) > \epsilon_i \\ 0, & \textit{otherwise} \end{cases} \tag{3}$$

For comparison with the SDVs from the MCQS, we define score pitch vectors (SPVs) with overtones for each note concurrence (i.e., notes with the same onset time) in the score. Assume that $\mathscr{P}_j^0$ is a set of note pitches in a concurrence in the score, where j is the index of the concurrence. Here, the harmonic series of a note is accounted for by the introduction of an overtone vector Ω equal to [0, 12, 19, 24], which are the index differences between the note and its overtones on the frequency axis. The elements of vector Ω correspond to the first to fourth harmonic partials of the note because an octave is divided into 12 frequency bins. For example, if a note has its first harmonic partial (i.e., fundamental frequency) at the first bin of the SPV, the 13th bin is its second harmonic partial, the 20th bin is its third harmonic partial, and so on.

We define overtone set $\mathscr{P}_j^n$ as follows:

$$\mathscr{P}_j^\ell = \mathscr{P}_j^{\ell-1} \cup \left(\mathscr{P}_j^0 + \Omega(\ell)\right) \tag{4}$$

where $\cup$ is the union operator. $\mathscr{P}_j^0 + \Omega(\ell)$ means that each element in the group $\mathscr{P}_j^0$ must be added $\Omega(n)$. In other words, $\mathscr{P}_j^\ell$ is an overtone variation of $\mathscr{P}_j^0$ and n stands for the number of overtones. Because we consider 4 harmonic partials, there are 4 sets for each concurrence. We define a score pitch vector with overtone (SPV) ϕ_j^ℓ as a binary vector derived from the set $\mathscr{P}_j^\ell$. The definition of a SPV ϕ_j^ℓ is as follows:

$$\phi_j^\ell(k) = \begin{cases} 1, & \textit{if } k \in \mathscr{P}_j^\ell \\ 0, & \textit{otherwise} \end{cases} \tag{5}$$

The frequency range of an SPV covers the lowest and highest note pitches in the score. The vector length of the SPV is the same as that of the SDV in the MCQS, and the frequency bins have one-to-one correspondence. If the set $\mathscr{g}_j^\ell$ contains a pitch, the corresponding index of the SPV is set at 1, or 0 otherwise. Here, four types of overtone distributions are considered to capture characteristics of different musical timbres from various instruments. Figure 5 shows a simple example of SPVs. Assume that there are two notes with corresponding pitches at indices 1 and 3 at the j-th concurrence, namely $\mathscr{g}_j^0 = \{1, 3\}$; ϕ_j^0 is a vector with a value 1 at indices 1 and 3. The following operations consider the first one, two, and three overtones to obtain three sets:

$$\begin{aligned}
\mathscr{g}_j^1 &= \mathscr{g}_j^0 \cup \left(\mathscr{g}_j^0 + \Omega(1)\right) = \{1, 3, 13, 15\} \\
\mathscr{g}_j^2 &= \mathscr{g}_j^1 \cup \left(\mathscr{g}_j^0 + \Omega(2)\right) = \{1, 3, 13, 15, 20, 22\} \\
\mathscr{g}_j^3 &= \mathscr{g}_j^2 \cup \left(\mathscr{g}_j^0 + \Omega(3)\right) = \{1, 3, 13, 15, 20, 22, 25, 27\}
\end{aligned}$$

ϕ_j^1, ϕ_j^2 and ϕ_j^3 then could be respectively built according to $\mathscr{g}_j^1$, $\mathscr{g}_j^2$ and $\mathscr{g}_j^3$, as shown in the figure. Each vector represents a possible variation of overtones for a concurrence. Our implementation only considers 4 variants for each concurrence.

We use a correlation coefficient to measure the similarity of two vectors. For each SDV ψ_i and each SPV $\left\{\phi_j^\ell,\ \ n = 0\sim3\right\}$ of a concurrence, their similarity is as follows:

$$S(i,j) = \max_{\ell=0\sim3} corr\left(\psi_i, \phi_j^\ell\right) \tag{6}$$

where the function $corr(\psi_1, \phi_2)$ returns the Pearson's linear correlation coefficient of vectors ψ_1 and ϕ_2.

Assume that M is the number of detected onsets in an audio performance and N is the number of concurrences in a score. The complexity of constructing a similarity matrix is O($M \times N$). For a long audio file, M and N are very large, and thus they considerably increase computation time. However, because the alignment path of audio-to-score alignment is usually concentrated diagonally, the similarity matrix does not need to be calculated in its entirety. To

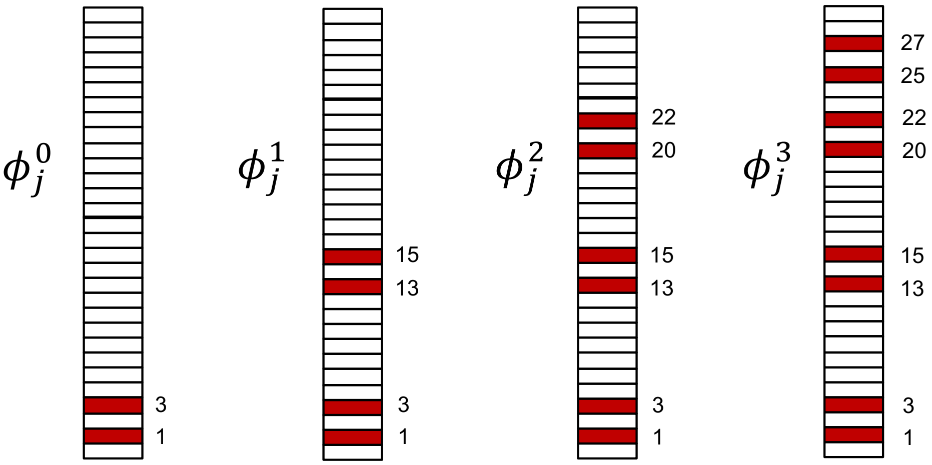


Fig. 5 An example of score pitch vectors (SPVs). ϕ_j^0 is a vector containing the pitch information at the concurrence j in the score. ϕ_j^1, ϕ_j^2 and ϕ_j^3 are variants which respectively consider the first one, two, three overtones

reduce the amount of computation, the number of SPVs is limited for comparison with each SDV. The pruning procedure is described in Algorithm 2.

Algorithm 2: Approach to pruning

```
1: procedure PRUNING:
2:     $w \leftarrow \max([\ |M - N|, w_0\ ])$
3:     for $i$ from 0 to $M$-1
4:         $j_0 \leftarrow \mathrm{round}(\ N / M * i)$
5:         $j_1 \leftarrow \min([\ j_0 - w, 0\ ])$
6:         $j_2 \leftarrow \max([\ j_0 + w, N\text{-}1\ ])$
7:         for $j$ from $j_1$ to $j_2$
8:             $S(\ i,\ j\ ) \leftarrow \max_{n=0\sim3} corr(\psi_i, \phi_j^n)$
```

The function round(x) rounds a floating-point number x to the nearest integer. The variable w_0 is a constant and is set to 5 in our implementation. After such treatment, the complexity is reduced from O($M \times N$) to O($M \times |M - N|$).

5 Dynamic programming

According to the aforementioned note-matching method, a similarity matrix $S(i,j)$ can be constructed, where i is an onset index of the input audio and j is an index of the concurrence in the music score. Each cell in the matrix represents the similarity between an SDV and SPV. Dynamic programming (DP) is employed to determine the path with maximum overall similarity.

$$D(i,j) = max\left\{ \begin{matrix} D(i\text{–}1,j) \\ D(i,j\text{–}1) \\ D(i\text{–}1,j\text{–}1) + S(i,j) + \eta(i,j) \end{matrix} \right\} \tag{7}$$

where $\eta(i,j)$ is a local tempo coefficient. Figure 6 shows an example of audio-to-score alignment. The matching curve was obtained for the audio used in Fig. 4. The upward-pointing triangles on the x-axis are the detected onsets of the audio. The right-pointing triangles on the y-axis are the positions of the concurrences in the score. The circles indicate the ground truth. The red asterisk is a matched pair in the DP process. The black asterisks are matched pairs back-traced from the red asterisk. The two dashed red lines are the first two segments and m_1 and m_2 are their slopes, which are used to evaluate the influence of temporal deviation. The local tempo coefficient is defined as follows:

$$\eta(i,j) = \alpha\left(1\text{–}\left|\log_2\frac{m_1}{m_2}\right|\right) \tag{8}$$

The ratio between m_1 and m_2 denotes the degree of tempo variation in a performance. If the tempo of the audio is the same as that of the score, the ratio is 1 and $\eta(i,j)$ is equal to α. If the tempo varies considerably, the ratio is considerably higher than 1 and $\eta(i,j)$ is decreased such that an SDV i is less likely to match the note vector j. The parameter α is a scalar that controls the influence of temporal deviation in the alignment process. In our experiment, we recognized that pitch matching is more crucial than temporal matching and thus set α to 0.1.

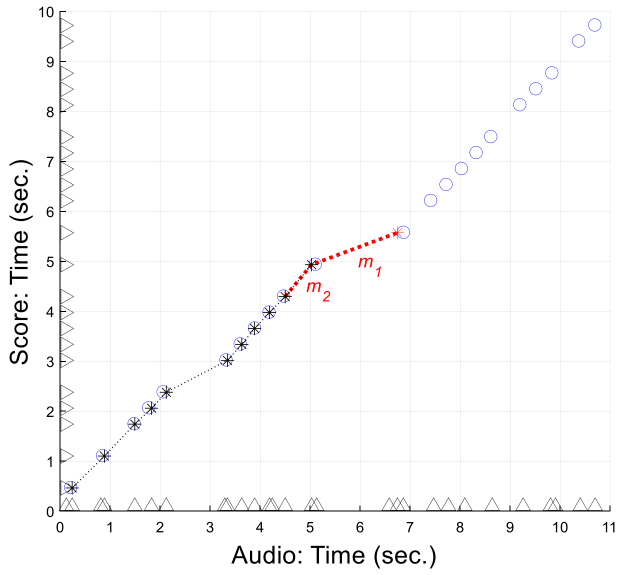


Fig. 6 An example of audio-to-score alignment. This matching curve is the result of the audio in Fig. 4. The circles are the groundtruth. The red asterisk is a matching pair in the dynamic programming process. The 2 dashed red lines are the first 2 segments back-traced from the red asterisk and m_1 and m_2 are their slopes which are used to evaluate the influence of temporal deviation

The alignment path is a sequence of adjacent cells, where each cell indicates a correspondence between an onset in the audio performance and a concurrence in the music score. After the accumulated similarity has been computed, the best alignment path by can be derived by back-tracking the path with the highest accumulated value in matrix D.

6 Real-time approach

Online and offline algorithms differ in terms of the time that is required to make an alignment decision. In real-time applications, a system must listen to music and immediately match the music to its corresponding notes in the score. The difference between the time at which an alignment decision is made and the estimated note onset time should be minimized. In this paper, we propose an online algorithm with a latency of only one frame (i.e., 10 ms) in most cases.

We created an online version of our offline method. The online method is based on onset detection and our note-matching method to enable construction of a similarity matrix. First, the onset detector is replaced with the online version. A state-of-the-art online onset detector [5] based on the RNN method is used. In addition, to obtain a positive result with low latency, the alignment process is designed to adhere to the following three conditions:

(1) Performance stability: A musician usually plays a piece of music according to tempo markings on the corresponding score. The performance tempo does not suddenly change, and thus predicting the next onset from previously matched onsets is possible.
(2) Performance continuity: We assume that the music is played according to the order of the notes in the score without any significant deviation. In other words, most matched

concurrences are continuous with the previous matched concurrence. The alignment system should not arbitrarily skip sequential concurrences or jump to other positions in the score.

(3) System response time: The real-time system must consider the response time. Therefore, the computation time for the alignment process must be reduced. Our algorithm balances precision and computation time; for each onset, only concurrences around the previous matched concurrence are considered for predicting score position.

To meet the above conditions, Eq. (8) should be slightly modified as follows:

$$\eta(i,j)=\begin{cases}\alpha\left(1-\left|\log_2\frac{m_1}{m_2}\right|-\Delta j\right), & if\ \frac{m_1}{m_2}<4\\ -1, & otherwise\end{cases} \tag{9}$$

We add an extra term, $\Delta j = j - j_0 - 1$, where j_0 is the index of the previous matched concurrence. This term is a penalty to avoid injurious score jumps. In addition, we limit the ratio between two adjoining tempo slopes to less than 4. Because of these simple modifications, the new version of the accumulation matrix is less affected by score jumps and onset insertions.

Figure 7 displays a flowchart of our proposed method, which is an onset-driven system. When an onset i is detected, the block "local tempo estimation" predicts the corresponding concurrence based on the assumption that the input audio has stable performance. The first step is to identify the most likely concurrence j_0 that is matched with the previous onset

$$j_0 = \underset{j_a-\Delta j\le j\le j_a+\Delta j}{\arg\max}\ D(i-1,j) \tag{10}$$

where j_a is the index of the previous matched concurrence and Δj is a tolerance window. Under normal circumstances, the current onset i should be matched to the concurrence $j_a + 1$ because of performance continuity. However, because some errors always exist in onset detection and the note-matching process, the real matched concurrence may deviate from $j_a + 1$, and thus the searching range $\pm\Delta j$ must be extended.

In the second step, a short-term back tracing is taken from the position $(i-1, j_0)$ in the accumulation matrix and the corresponding concurrence for the current onset i is estimated. The formula is as follows:

$$j_1 = \underset{j_0\le j\le j_0+\Delta j}{\arg\min}\ \left|f_{tempo}(j)-\tau_i\right| \tag{11}$$

where the function $f_{tempo}(j)$ predicts the corresponding onset time in the performance for the note concurrence j. The procedure of this function contains computation of the following three variables:

(1) Local alignment path: First, we track back N steps from $(i-1, j_0)$ in the accumulation matrix and obtain a local alignment path. The choice of N is relative to the local tempo variation of the audio. We set N as 5 in our experiment.

(2) Average tempo slope: We calculate the average intervals of the aligned positions of the audio and score according to the local alignment path. The average tempo slope is the ratio between these two average intervals.

(3) Probable onset time: Assuming that the tempo ratio at the current onset is the same as the average tempo ratio of the local alignment path, we can estimate the corresponding onset time of a note concurrence through linear extrapolation from the local alignment path.

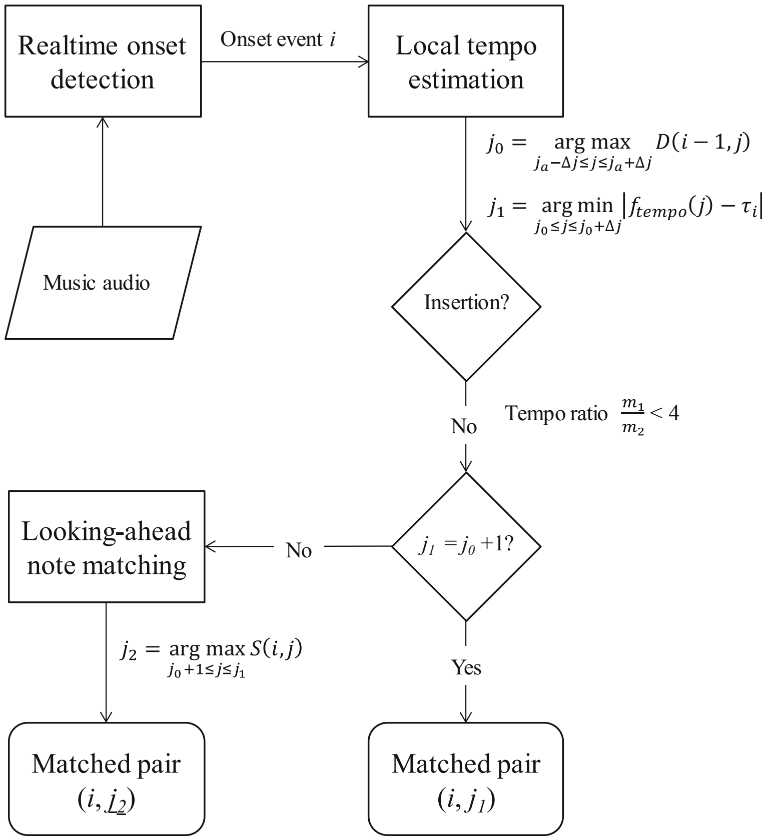

Realtime onset detection
Onset event i
Local tempo estimation
Music audio
$j_0 = \arg\max_{j_a-\Delta j\le j\le j_a+\Delta j} D(i-1,j)$
$j_1 = \arg\min_{j_0\le j\le j_0+\Delta j} |f_{tempo}(j) - \tau_i|$
Insertion?
No
Tempo ratio $\frac{m_1}{m_2} < 4$
Looking-ahead note matching
No
$j_1 = j_0 + 1$?
$j_2 = \arg\max_{j_0+1\le j\le j_1} S(i,j)$
Yes
Matched pair (i, j_2)
Matched pair (i, j_1)

Fig. 7 Flowchart of the proposed online audio-to-score alignment method. It is an onset-driven system. When an onset i is detected, the block "local tempo estimation" will predict the corresponding note concurrence. The block "looking-ahead note matching" will determine the optimal concurrence based on the note-matching method

We consider the concurrences that range from j_0 to $j_0 + \Delta j$ and select that with the smallest gap to the current onset. If the current detected onset is an insertion, the system discards it and waits for the next onset. A simple strategy is employed to filter out the insertion; specifically, the ratio of the tempo slope of the current onset to the average tempo slope of the local estimation is limited to less than 4.

The second comparison checks performance continuity. If the predicted concurrence j_1 is equal to $j_a + 1$, the current onset i is matched to the concurrence j_1; otherwise, there may be a score jump, which would require further checking of the note pitch by using our note-matching method. The formula is as follows:

$$j_2 = \underset{j_0+1\le j\le j_1}{\arg\max}\ S(i,j) \tag{12}$$

where $S(i,j)$ is the similarity matrix in Eq. (6). Improving note-matching accuracy requires reading a few frames further. In our experiments, we read an additional 40 ms of audio data after the current onset time. The range of computation is limited in $j_0 + 1$ and j_1, and the optimal concurrence j_2 is matched to the current onset i.

To clarify our method, we provide two examples here. Figure 8a contains no score jump; the piece of music is the same as that in Fig. 6 and is similarly labeled. The dashed vertical

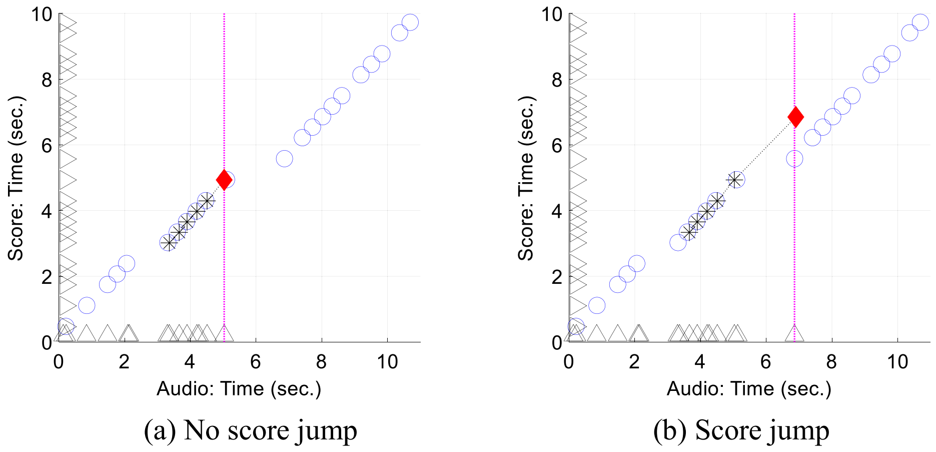


Fig. 8 Two examples of real-time audio-to-score alignment. The vertical magenta dashed line represents current detected onset timing. The black asterisks are the local alignment path. The red diamond is the matched score position according to the local tempo estimation. **a** The estimated score position is continuous with the local alignment path. **b** There is a score jump between the estimated score position and the local alignment path

magenta line represents the current detected onset timing. We estimated the score position using Eq. (9) and traced back five steps to obtain a local alignment path, which is denoted by black asterisks. We were then able to compute the average slope between these points and predict the score position that corresponded to the current onset of the audio. The most likely score position is denoted by the red diamond in Fig. 8a. We found that the estimated concurrence index was next to the previous one, denoting that no score jump occurred in this example. By contrast, Fig. 8b shows an example of a score jump; the estimated concurrence index (red diamond) is far from the alignment path (black asterisks), denoting a large jump of notes between them. If this estimated concurrence index is directly matched with the current onset, the subsequent alignment work may be lost, resulting in incorrect output. Thus, this case requires note matching of the nearby notes and selection of the most likely concurrence.

The decision time of our method is approximately equal to the onset time plus computation time of the local tempo estimation, except in the case of score jump, which requires additional audio data for note matching. Therefore, our method has a short response time. The reliability of this method was verified in an experiment. This paper details the experiment and compares our method with other methods.

7 Experiments

To demonstrate the performance of our proposed method and the influence of certain parameters, we conducted a series of experiments. The main task of this study was to verify the feasibility of the proposed alignment framework. Because our alignment method is based on onset detection, we intended to observe how the method influences alignment performance. In addition, we have proposed a fusion method of onset detection functions (ODFs). Therefore, experiment 1 was designed to investigate the performance of several onset detectors. Experiment 2 obtained results for audio-to-score alignment based on various onset detectors; in this paper, these results are discussed alongside the effects on final performance of various onset detectors and spectrum difference vectors. In experiment 3, we verified the feasibility of the proposed online approach and compared its performance with that of other online methods.

7.1 Datasets

The evaluation of audio-to-score alignment was performed using two datasets. Each recording had a corresponding MIDI representation of the score. The audio files were recordings of real music performances in 44.1-kHz 16-bit WAV format. The first dataset, Bach10 [18], consists of 10 human-played J.S. Bach four-part chorales, with 30-s audio files sampled from real music performances by an instrumental quartet consisting of violin, clarinet, tenor saxophone, and bassoon. Each instrument was recorded separately. Individual recordings of the same chorale were then mixed to create 60 duets, 40 trios, and 10 performances with four-part polyphony. The total number of notes is approximately 13,700 and the total duration is approximately 60 min. The second dataset, which was collected by Cont [12], has 46 recordings extracted from four distinct pieces of classical music performed in a monophonic or slightly polyphonic manner. The total number of notes is slightly more than 4600 and the total duration is approximately 32 min. Notably, the original human annotations in the second dataset have larger deviation from the real onset positions, and thus the ground truth alignment between the audio and MIDI has been relabeled. The revised labels have been released as a public resource.[2] These datasets contain a variety of musical types and tempo variation; they are used in MIREX score following contest. Recent studies [11, 23, 24] on multi-instrumental audio-to-score alignment have used these datasets. Therefore, these datasets were adopted to evaluate the proposed method in the experiment of the present study.

7.2 Experiment 1: onset detection

In experiment 1, we compared the performance of the proposed method with that of four other methods from the literature. The first was a convolutional neural network (CNN)-based method that is the start-of-the-art detector proposed in [37], and the other three were SuperFlux, ComplexFlux, and the RNN method, which were the constituted ODFs in our implementation of the fusion method. We have included their results to illustrate the improvement offered by the proposed fusion algorithm.

We trained our onset detector on several previously released datasets [2, 20, 21, 26] consisting of music of a range of genres, styles, and instrumentation. We used 203 audio excerpts with an overall duration of approximately 36 min and 10,053 onsets. For SVM training, the full datasets were initially split into five disjunctive folds to optimize the parameters of the proposed method based on five-fold cross validation within the training set.

For evaluation, the standard measures of precision, recall, and the F-measure were used. An onset was considered correctly detected if a ground truth annotation was observed within a tolerance window around the predicted position. If two or more decisions were made corresponding to a ground truth annotation within a tolerance window, only one decision was counted as a true positive, and all other were rendered false positives. In our experiments, we selected a tolerance window of ±50 ms, which is identical to that used in the MIREX audio onset evaluation task.

Tables 1 and 2 respectively list the experimental results for the Bach10 and Cont datasets. The bold type indicates the best result for each measure. As shown in Table 1, the CNN

[2] The revised labels of the dataset can be downloaded in https://github.com/audioscoredata/audio-to-score-label

Table 1 Onset detection results in BACH10 dataset

Method	P (%)	R (%)	F (%)
SuperFlux	57.08	82.28	67.20
ComplexFlux	56.73	82.72	67.12
RNN	54.96	82.14	65.62
CNN	59.33	83.38	**69.10**
Fusion (proposed)	59.18	82.13	68.58

The bold types indicate the best result for each measure

method had the highest precision, recall, and F-measure among all the methods. As shown in Table 2, the fusion method had the highest precision and F-measure, whereas the CNN method had the highest recall. In both tables, the fusion method had higher precision than did its constituted ODFs—SuperFlux, ComplexFlux, and the RNN method. Although its recall was slightly lower, overall, the F-measure was considerably improved. Notably, the improvement of F-measure was more evident in the second dataset because that dataset contained more monophonic music, for which features extracted from H/P curves are more helpful. Taken together, Tables 1 and 2 indicate the effectiveness of the proposed method. Moreover, our method is more transparent than the CNN-based method because we combine features from three ODFs and the final classifier is an SVM. Moreover, the proposed framework is relatively flexible, and thus the introduction of new features and new fusion methods is fairly straightforward; for example, we added H/P features to reduce interference from vibrato. By contrast, the CNN employs fully connected layers for classification; this approach is difficult to understand and generally viewed as a hard-to-interpret black box.

The onsets detected by the different methods were used in the audio-to-score alignment experiment described as follows, where we observed the impact of the onset detector on our alignment method.

7.3 Experiment 2: offline audio-to-score alignment

The second experiment compared the proposed algorithm with other audio-to-score alignment methods. The evaluation metrics are based on the precision of the matched notes in the score, which have been frequently adopted in related studies, such as [8, 10, 18, 23, 24, 33]. A note is said to be correctly matched if its estimated onset does not deviate from the real one by more than the specified tolerance window (e.g., 50 ms). The figure of merit used in the experiment was piecewise precision, which is the average of the percentage of detected notes for each recording.

The performance evaluation results for the Bach10 dataset are listed in Table 3, which details 18 systems for evaluation. The first 3 are based on published papers, whereas the remaining 15 are combinations from five onset detectors and three SDVs. RB is derived from Dannenberg's algorithm [22], which uses chroma features and DTW alignment; this algorithm

Table 2 Onset detection results in Cont dataset

Method	P (%)	R (%)	F (%)
SuperFlux	72.39	77.06	71.62
ComplexFlux	76.48	83.66	78.54
RNN	76.36	90.30	81.97
CNN	68.15	94.38	77.78
Fusion (proposed)	85.34	87.31	**85.74**

The bold types indicate the best result for each measure

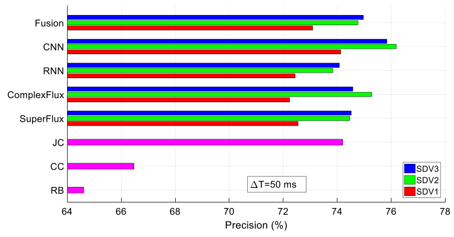
Table 3 Audio-to-score alignment results in BACH10 dataset

Method		25 ms (%)	50 ms (%)	75 ms (%)	100 ms (%)
RB		37.73	64.61	79.37	85.82
CC		42.99	66.47	80.75	86.98
JC		NA	~74.20	NA	~90.00
SuperFlux	SDV1	48.55	72.55	86.87	92.33
	SDV2	50.21	74.46	88.83	94.21
	SDV3	**50.28**	**74.52**	**88.96**	**94.34**
ComplexFlux	SDV1	48.37	72.24	86.34	92.24
	SDV2	**50.91**	**75.28**	**88.74**	**94.21**
	SDV3	50.51	74.58	88.43	93.90
RNN	SDV1	47.67	72.44	86.23	91.50
	SDV2	48.47	73.84	87.67	92.99
	SDV3	**48.70**	**74.08**	**87.82**	**93.21**
CNN	SDV1	53.04	74.13	86.22	91.63
	SDV2	**54.29**	**76.19**	**88.17**	**93.41**
	SDV3	54.18	75.84	87.93	93.39
Fusion	SDV1	49.55	73.09	86.07	91.93
	SDV2	51.01	74.77	87.78	93.50
	SDV3	**51.03**	**74.97**	**88.11**	**93.75**

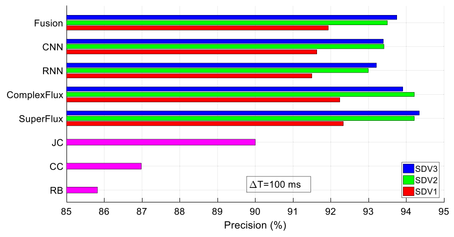
The bold types indicate the best result for each measure

was chosen as a baseline in our experiment. CC is an approach that we proposed in an earlier study [10]; in CC, an onset detector employs an RNN-based method and the similarity measure is based on NMF to decompose note overtones. JC is one of the most recently proposed methods [8]; it decomposes spectral patterns using learned basis functions and uses DTW to identify the best alignment path. Notably, the results of JC were estimated from Figure 3 in [8], and thus only two data items were included. The remaining 15 systems were based on the proposed methods, with five onset detectors (SuperFlux, ComplexFlux, an RNN, a CNN, and the fusion method) and three SDVs, namely SDV1, SDV2, and SDV3, which are the three representations of SDV ψ_i^m in Eqs. (1), (2), and (3), respectively, in Section 4.

As shown in Table 3, four tolerance windows were used to evaluate performance. For each onset detector with a specified tolerance window, the percentage in bold shows the most favorable result in the three SDVs. For the SuperFlux, RNN, and fusion methods, SDV3 consistently outperformed SDV1 and SDV2; for ComplexFlux and CNN, SDV2 consistently outperformed SDV1 and SDV3. In addition, our 15 alignment systems evidently outperformed RB and CC. For comparison with JC, Fig. 9 shows bar charts of the precision for the tolerance windows of 50 and 100 ms. As shown in Fig. 9a, where the tolerance window is 50 ms, except for SDV1, our systems all outperformed JC in



(a) Tolerance window=50 ms



(b) Tolerance window=100 ms

Fig. 9 Audio-to-score alignment results in Bach10 dataset at the two tolerance windows of (**a**) 50 ms and (**b**) 100 ms.

all onset detectors and RNN + SDV2. As shown in Fig. 9b, when the tolerance window was 100 ms, our systems consistently outperformed JC. This indicates that the proposed method is fairly robust with respect to different onset detectors.

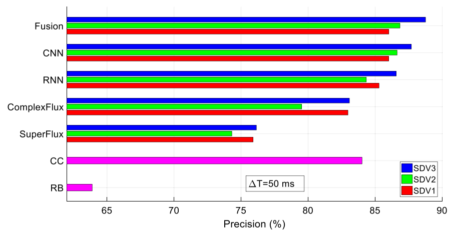
Notably, the F-measure of the fusion method was higher than those of its constituted onset detectors (SuperFlux, ComplexFlux, and RNN), as shown in Table 1. However, Table 3 shows that some precision values of the alignment systems with the fusion method were lower than those with SuperFlux and ComplexFlux, possibly because SuperFlux and ComplexFlux have higher recall, leading to more choices of SDVs when DP matching is performed with SPVs to obtain better alignment. Taken as a whole, the proposed note-matching method achieves favorable results when combined with various onset detectors.

Table 4 provides the results of our experiment on the Cont dataset. The proposed method is compared with RB and CC only because we did not obtain results for JC with this dataset. To improve readability, Fig. 10 shows bar charts for the tolerance windows of 50 and 100 ms. The results of all our combinations were significantly favorable to those of RB, thereby demonstrating the robustness of the proposed note-matching method. In addition, our proposed note-matching method outperformed the CC method, except for the alignment systems with SuperFlux and ComplexFlux. This is because SuperFlux and ComplexFlux did not capture onsets sufficiently successfully in this dataset, as shown in Table 2. As for the CNN, although its precision and F-measure are low in Table 2, it still performed well for audio-to-score alignment. This is similar to the results of SuperFlux and ComplexFlux in the Bach10 dataset, where higher recall denoted more SDVs for better alignment. Thus, to some extent, the importance of recall is higher than that of precision in our alignment system; however, precision must be maintained. For instance, the fusion method had lower recall and considerably higher precision than did the CNN, as shown in Table 2. This led to generally greater alignment for the fusion method than for the CNN, as shown in Table 4. Furthermore, the overall performance of SDV3 was consistently favorable to that of SDV1 and SDV2 in the Cont dataset. In summary, fusion+ SDV3 had the highest precision in all tolerance windows for this dataset.

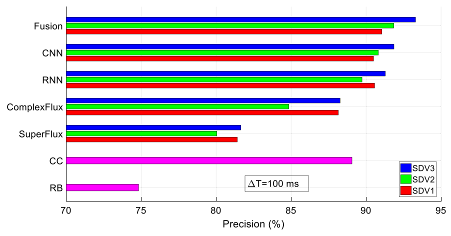
Table 4 Audio-to-score alignment results in Cont dataset

Method		25 ms (%)	50 ms (%)	75 ms (%)	100 ms (%)
RB		43.22	63.89	71.29	74.83
CC		65.44	84.02	87.66	89.06
SuperFlux	SDV1	62.09	75.89	79.88	81.41
	SDV2	60.52	74.31	78.56	80.05
	SDV3	**62.11**	**76.14**	**80.13**	**81.64**
ComplexFlux	SDV1	**69.61**	82.97	86.62	88.15
	SDV2	66.48	79.52	83.21	84.84
	SDV3	69.49	**83.08**	**86.80**	**88.27**
RNN	SDV1	66.53	85.29	88.97	90.57
	SDV2	66.18	84.34	88.32	89.73
	SDV3	**68.03**	**86.57**	**89.88**	**91.28**
CNN	SDV1	73.37	86.01	88.96	90.50
	SDV2	74.27	86.64	89.53	90.82
	SDV3	**74.71**	**87.70**	**90.66**	**91.86**
Fusion	SDV1	72.82	86.02	89.47	91.05
	SDV2	73.63	86.85	90.36	91.84
	SDV3	**75.72**	**88.76**	**92.01**	**93.30**

The bold types indicate the best result for each measure



(a) Tolerance window=50 ms



(b) Tolerance window=100 ms

Fig. 10 Audio-to-score alignment results in Cont dataset at the two tolerance windows of (**a**) 50 ms and (**b**) 100 ms.

Bach10 contains polyphonic music played on multiple instruments, whereas the Cont dataset contains more monophonic music. Although these two datasets have different properties, we observed a similar trend in both, namely that the overall performance of SDV3 was the best in all onset detectors. In addition, the alignment results revealed a positive correlation with the results of onset detection. To provide a more intuitive view of all methods, we merged the results obtained using the two datasets and re-evaluated the precision of the detected notes in tolerance window sizes ranging from 25 to 100 ms in 25 ms increments. To improve readability, the results of SDV1 and SDV2 are not presented. The obtained results are plotted in Fig. 11. All our systems achieved results favorable to those of the other methods included for comparison, namely RB and CC, in all tolerance windows. CNN + SDV3 outperformed RB by 20.88% and CC by 10.62% when the tolerance window was 25 ms. Fusion+SDV3 outperformed RB by 11.04% and CC by 6.03% when the tolerance window was 100 ms.

7.4 Experiment 3: online audio-to-score alignment

In experiment 3, Bach10 was used to evaluate the performance of the proposed score follower for online alignment. We adopted SDV3 for computation of the similarity matrix. Figure 12 presents our experimental results. We compared our score follower with four reference methods. Soundprism [18] is a system that uses a particle filter to infer the tempo and score position of each audio frame in online mode. The algorithms ISMIRon [8], BACKspeed, and FWDspeed [33] are derived from the

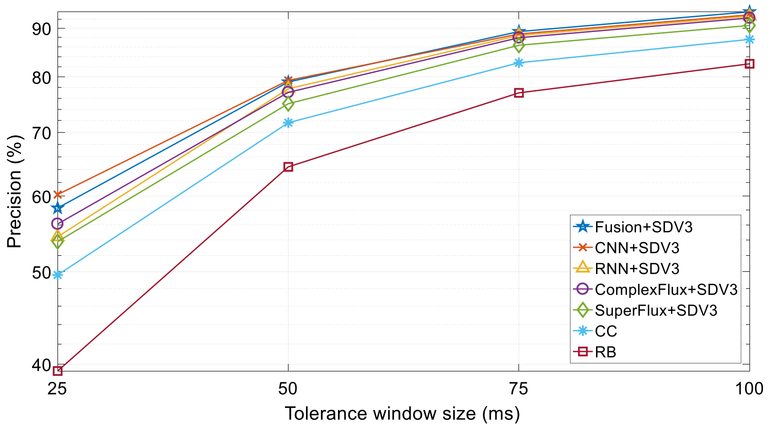


Fig. 11 Experimental results based on both datasets. The first 5 are based on the proposed algorithm, which compares favorably with the algorithms CC and RB proposed in the literature

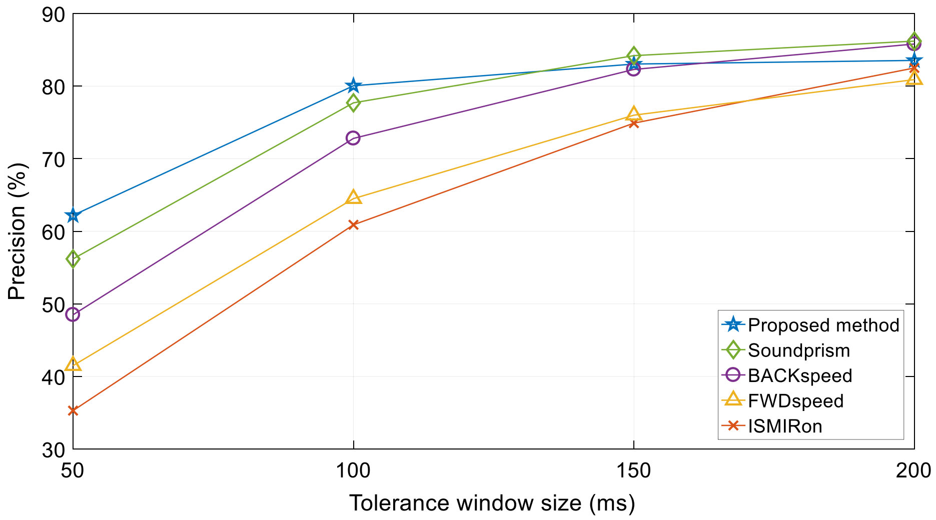


Fig. 12 Results of online audio-to-score alignment. The proposed method produces the best precision when the tolerance window size is less than or equal to 100 ms.

offline algorithm JC, which was introduced in the previous section. ISMIRon matches each performance frame with a score position associated with the minimum value of the accumulated cost, whereas BACKspeed and FWDspeed are two variants that estimate the DTW path in online mode by introducing reliable onset positions.

We determined the precision of detected notes in tolerance window sizes ranging from 50 to 200 ms in 50 ms increments, as shown in Fig. 12; the proposed system obtained the best results when the tolerance window was shorter than or equal to 100 ms. As the tolerance window was increased, the precision improvement of our method compared with the other methods became smaller because some onsets were falsely ignored by the employed onset detector, and thus the corresponding notes were not recognized. Although the precision of our method was slightly lower than that of Soundprism at tolerance window sizes of 150 and 200 ms, our method had the highest precision at 50 and 100 ms, and had the lowest average offset of all the compared methods. In addition, our method was able to detect 77% of all notes in 10-ms latency and 23% in 50-ms latency, leading to an average latency of only 19.2 ms. Therefore, our method is relatively suitable for online audio-to-score alignment. The success of this approach can be attributed to the high accuracy of our note-matching method and the reliability of local tempo estimation.

8 Conclusions and future work

We have proposed an algorithm that aligns an audio recording of polyphonic music to its corresponding score. The algorithm involves three steps: onset detection, similarity computation between an SDV derived from an audio onset and an SPV derived from the concurrent note-on event in the score, and a dynamic program to determine the optimal alignment. We have also proposed a fusion method for onset detectors and a note-matching method based on our MCQS. We reduce the complexity of similarity computation and interference from vibrato by considering spectrum variation around an onset. Our note-matching method is a direct comparison of SDVs and SPVs that is more reliable when the timbres in the audio differ from those specified in the music score. We implemented 15 alignment systems with combinations from five onset detectors and three SDVs and compared them with other audio-to-score alignment algorithms proposed in the literature. Our systems obtained higher alignment precision than did all compared methods. Finally, we proposed an online approach based on onset detection where musical notes can be detected within an average window of 19.2 ms. Our online version outperformed all the methods included for comparison when the tolerance windows were 50 and 100 ms.

Although the proposed method performed favorably in the experiments, there is still room for improvement. In future research, we will focus on improving various aspects of the alignment algorithm. Machine learning will be used to analyze the spectral patterns of the constant Q transform to produce a more advanced similarity matrix. Moreover, we intend to develop a musical accompaniment system for polyphonic music performances to demonstrate the feasibility of the proposed algorithm.

Acknowledgments This research is partially supported by Ministry of Science and Technology, ROC, under Grant no. MOST 104-2221-E-002-051-MY3.

Appendix

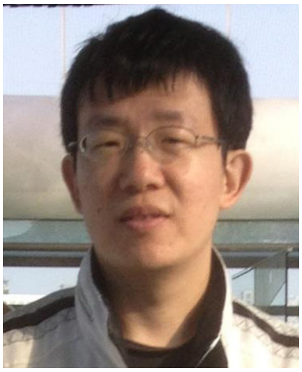
In order to make our presentation of the proposed framework clear, here we list symbols and their definitions as follows.

- w_m: The window size in frame number, where m is an integer
- θ: The thresholding parameter of the peak picking
- n_a: The frame index of the average time of a peak group
- Hv: The harmonic curve which derives from the harmonic component of a music recording
- Pv: The percussive curve which derives from the percussive component of a music recording
- n_p: The frame index of local maxima of Pv.
- n_h: The frame index of local maxima of Hv.
- dA_i: The spectrum difference around an onset i.
- ψ_i^m: The spectrum difference vector that derives from the spectrum difference dA_i, where m means the type of processing
- $\mathscr{P}_j^\ell$: The set of note pitches and overtones of a concurrence j in the score, where ℓ is the number of overtones.
- Ω: The overtone vector
- S: The similarity matrix of the input audio and the music score
- M: The number of detected onsets in the audio
- N: The number of concurrences in the score
- η: The local tempo coefficient

References

1. Arzt A, Widmer G, Dixon S (2008) Automatic page turning for musicians via real-time machine listening. Proceedings of European Conference on Artificial Intelligence (ECAI), p 241–245
2. Bello JP, Daudet L, Abdallah S, Duxbury C, Davies M, Sandler MB (2005) A tutorial on onset detection in music signals. IEEE Trans Audio Speech Lang Process 13:1035–1047
3. Böck S, Widmer G (2013) Maximum filter vibrato suppression for onset detection. Proceedings of the 16th International Conference on Digital Audio Effects, p 55–61
4. Böck S, Widmer G (2013) Local group delay based vibrato and tremolo suppression for onset detection. Proceedings of the 14th International Society of Music Information Retrieval Conference (ISMIR), p 361–366
5. Böck S, Korzeniowski F, Schlüter J, Krebs F, Widmer G (2016) madmom: a new Python audio and music signal processing library. Proceeding MM '16 Proceedings of the 2016 ACM on Multimedia Conference, p 1174–1178
6. Brown JC (1991) Calculation of a constant Q spectral transform. J Acoust Soc Am 89(1):425–434
7. Cai J, Guo Y, Wang H, Wang Y (2014) Score-informed source separation based on real-time polyphonic score-to-audio alignment and bayesian harmonic model. International Conference on Computational Intelligence and Communication Networks, p 672–680
8. Carabias-Orti JJ, Rodriguez-Serrano FJ, Vera-Candeas P, Ruiz-Reyes N, Canadas-Quesada FJ (2015) An audio to score alignment framework using spectral factorization and dynamic time warping. 16th International Society for Music Information Retrieval (ISMIR) Conference, p 742–748
9. Chen C-T, Jang J-SR, Liou W (2014) Improved score-performance alignment algorithms on polyphonic music. Proceedings of the 39th IEEE International Conference on Acoustics, Speech and Signal Processing, p 1365–1369
10. Chen C-T, Jang J-SR, Liou W-S, Weng C-Y (2016) An efficient method for polyphonic audio-to-score alignment using onset detection and constant Q transform. Proceedings of the 41st IEEE International Conference on Acoustics, Speech and Signal Processing, p 2802–2806
11. Cont A (2006) Realtime audio to score alignment for polyphonic music instruments, using sparse non-negative constraints and hierarchical HMMS. Proceedings of the 31st International Conference on Acoustics, Speech and Signal Processing, p 245–248
12. Cont A, Schwarz D, Schnell N, Raphael C (2007) Evaluation of real-time audio-to-score alignment. International Society on Music Information Retrieval, p 315–316
13. Dannenberg RB (1984) An on-line algorithm for real time accompaniment. Proceedings of the 1984 International Computer Music Conference, p 193–198
14. Dannenberg RB, Hu N (2003) Polyphonic audio matching for score following and intelligent audio editors. Proceedings of the 2003 International Computer Music Conference, San Francisco: International Computer Music Association, p 27–34
15. Degara-Quintela N, Pena A, Torres-Guijarro S (2009) A comparison of score-level fusion rules for onset detection in music signals. Proceedings of the 10th International Conference on Music Information Retrieval, p 117–121
16. Dixon S (2006) Onset detection revisited. Proceedings of the International Conference on Digital Audio Effects, p 133–137
17. Dorfer M, Arzt A, Widmer G (2017) Learning audio-sheet music correspondences for score identification and offline alignment. Proceedings of the International Society for Music Information Retrieval Conference, p 115–122
18. Duan Z, Pardo B (2011) Soundprism: an online system for score-informed source separation of music audio. IEEE J Sel Top Signal Process 5(6):1205–1215
19. Duxbury C, Bello JP, Davies M, Sandler MB (2003) A combined phase and amplitude based approach to onset detection for audio segmentation. Proceedings of the European Workshop on Image Analysis for Multimedia Interactive Services, p 275–280
20. Eyben F, Böck S, Schuller B, Graves A (2010) Universal onset detection with bidirectional long short-term memory neural networks. Proceedings of the 11th International Conference on Music Information Retrieval, p 589–594
21. Holzapfel A, Stylianou Y, Gedik AC, Bozkurt B (2010) Three dimensions of pitched instrument onset detection. IEEE Trans Audio Speech Lang Process 1517–1527
22. Hu N, Dannenberg RB, Tzanetakis G (2003) Polyphonic audio matching and alignment for music retrieval. Proceedings IEEE WASPAA, New Paltz, p 185–188
23. Joder C, Essid S, Richard G (2011) A conditional random field framework for robust and scalable audio-to-score matching. IEEE Trans Audio Speech Lang Process 19(8):2385–2397
24. Joder C, Essid S, Richard G (2013) Learning optimal features for polyphonic audio-to-score alignment. IEEE Trans Audio Speech Lang Process 21(10):2118–2128
25. Lacoste A, Eck D (2005) Onset detection with artificial neural networks. Proceedings of the International Conference on Music Information Retrieval

26. Lacoste A, Eck D (2007) A supervised classification algorithm for note onset detection. EURASIP J Appl Signal Process 153–166
27. Lerch A (2012) "Alignment". An introduction to audio content analysis: applications in signal processing and music informatics. Wiley, Hoboken, p 148–149
28. Müller M (2007) Music synchronization. Information retrieval for music and motion. Springer, p 85–108
29. Ono N, Miyamoto K, Kameoka H, Le Roux J, Uchiyama Y, Tsunoo E, Nishimoto T, Sagayama S (2010) Harmonic and percussive sound separation and its application to MIR-related tasks. Adv Music Inf Retr 274:213–236
30. Orio N, Schwarz D (2001) Alignment of monophonic and polyphonic music to a score. Proceedings 2001 ICMC, p 155–158
31. Orio N, Lemouton S, Schwarz D (2003) Score following: State of the art and new developments. Proceedings of the 2003 conference on New interfaces for musical expression, Montreal, Canada, p 34–41
32. Raffel C, Ellis DPW (2016) Optimizing DTW-based audio-to-MIDI alignment and matching. Proceedings of the 41st IEEE International Conference on Acoustics, Speech and Signal Processing, p 81–85
33. Rodriguez-Serrano FJ, Carabias-Orti JJ, Vera-Candeas P, Martinez-Muñoz D (2017) Tempo driven audio-to-score alignment using spectral decomposition and online dynamic time warping. ACM Trans Intell Syst Technol 8(2):1–20
34. Sako S, Yamamoto R, Kitamura T (2014) Ryry: a real-time score-following automatic accompaniment playback system capable of real performances with errors, repeats and jumps. Active Media Technology: 10th International Conference, AMT 2014, Warsaw, Poland, p 134–145
35. Salamon J, Gómez E, Ellis DPW, Richard G (2014) Melody extraction from polyphonic music signals: approaches, applications and challenges. IEEE Signal Process Mag 31(2):118–134
36. Schlüter J, Böck S (2013) Musical onset detection with convolutional neural networks. International Workshop on Machine Learning and Music (MML), Prague, Czech Republic, p 1–4
37. Schlüter J, Böck S (2014) Improved musical onset detection with convolutional neural networks. Proceedings of the 39th International. Conference on Acoustics, Speech and Signal Processing, p 6979–6983
38. Song X, Ming Z, Nie L, Zhao Y-L, Chua T-S (2016) Volunteerism tendency prediction via harvesting multiple social networks. ACM Trans Inf Syst 34(2):10:1–10:27
39. Tachibana H, Ono N, Kameoka H, Sagayama S (2014) Harmonic/percussive sound separation based on anisotropic smoothness of spectrograms. IEEE/ACM Trans Audio Speech Lang Process 22:2059–2073
40. Tian M, Fazekas G, Black DAA, Sandler M (2014) Design and evaluation of onset detectors using different fusion policies. Proceedings of the International Society of Music Information Retrieval, p 631–636
41. Ueda Y, Uchiyama Y, Nishimoto T, Ono N, Sagayama S (2010) HMM-based approach for automatic chord detection using refined acoustic features. Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, p 5518–5521
42. Wang S, Ewert S, Dixon S (2016) Robust and efficient joint alignment of multiple musical performances. IEEE Trans Audio Speech Lang Process 24(11):2132–2145



Chunta Chen is currently pursuing the Ph.D. degree in the School of Computer Science in Nation Tsing Hua University. He is currently working on audio analysis and intelligent music retrieval system development. His research interests include multimedia recommendation, content based music retrieval and machine learning.

Jyh-Shing Roger Jang (M’93) received his Ph.D. from the EECS Department at the University of California, Berkeley. He was a professor in the CS Dept. of Nation al Tsing Hua Univ., Taiwan, from 1995 to 2012. Since August 2012, he has been a professor in the CSIE Dept. of National Taiwan Univ., Taiwan. His research interests include machine learning and pattern recognition, with applications to speech recognition, music retrieval, image identification, and semiconductor manufacturing intelligence.