

Chi Hang Wong · Wai Man Szeto · Kin Hong Wong

Automatic Lyrics Alignment for Cantonese Popular Music

Abstract From lyrics-display on electronic music players and Karaoke videos to surtitles for live Chinese opera performance, one feature is common to all these everyday functionalities: temporal synchronization of the written text and its corresponding musical phrase. Our goal is to automate the process of lyrics alignment, a procedure which, to date, is still handled manually in the Cantonese popular song (Cantopop) industry.

In our system, a vocal signal enhancement algorithm is developed to extract vocal signals from a CD recording in order to detect the onsets of the syllables sung and to determine the corresponding pitches. The proposed system is specifically designed for Cantonese, in which the contour of the musical melody and the tonal contour of the lyrics must match perfectly. With this prerequisite, we use a dynamic time warping algorithm to align the lyrics. The robustness of this approach is supported by experiment results. The system was evaluated with 70 twenty-second music segments and most samples have their lyrics aligned correctly.

1 Introduction

Many popular song listeners find following lyrics in their written form when a song is being played an enjoyable experience. Many music players, including both hardware (e.g. Apple iPod) and software (e.g. Microsoft Media Player), feature sentence-by-sentence lyrics-display function. Besides, it is usual for Karaoke (a popular form of entertainment in Asia since 1980s) singers (and viewers as well) to expect lyrics to appear on screen at exactly the same moment as it is to be sung. Furthermore, surtitles¹ in a live Chinese opera performance have to be

Chi Hang Wong · Wai Man Szeto · Kin Hong Wong
Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
E-mail: {ch Wong1, wmszeto, kh Wong}@cse.cuhk.edu.hk

¹ Surtitles are the translations of opera libretto projected on screen *above* stage during performance [14]. During a live

displayed accurately at the correct time positions. This is a challenging application, as lyrics display has to be executed in real-time and the singing speed is entirely at the performer's own discretion; the system has to be fast, adaptive and thus, user-centered. All these demands are related to one thing: temporal synchronization of written texts and their corresponding musical phrases. It has been a slow and costly procedure in the music industry because all synchronizations are done manually. The system proposed here is our response to this problem (called lyrics alignment below) for Cantonese popular songs (Cantopop).

Cantonese is a major dialect in Southern China, in which about 120 million people use it daily. An important characteristic, which helps to automate the Cantopop lyrics alignment process, is that Cantonese is a tone language. *"In well over half of the languages of the world, it is possible to change the meaning of a word simply by changing the pitch level at which it is spoken. Languages that allow this are known as tone languages, and the distinctive pitch levels are known as tones or tonemes... Cantonese Chinese has six [tones]"* [8].

In analyzing the relationship between tone and melody, Marjorie categorized these 6 tones to three groups: high, mid and low, and showed that the lyrics were written to match the melodic contour of the musical phrase [5]. For example, if the lyrics consist of two Chinese characters: lou syu 老鼠 (meaning "rat")² in which a mid-pitch syllable is followed by a high-pitch syllable, and each syllable matches a musical note, the musical interval (pitch distance) of these two notes must be an ascending major 2nd (whole tone) such as "DO"- "RE", "RE"- "MI", etc. If the songwriter writes "FA"- "RE" (a descending minor 3rd), the lyrics become 老樹 (meaning "old tree"). Therefore, in order to convey the meaning of the lyrics accurately, the contour of the melody and that of the lyrics must

Cantonese opera performance in Hong Kong, the Chinese libretto is projected on screen instead of a translation.

² We transcribe the vowel(s) and consonant(s) of a Cantonese syllable by using the transcription system of Linguistic Society of Hong Kong (LSHK) [21].

match each other. We made use of this characteristic as one of the features of the lyrics in the lyrics alignment problem.

To align the lyrics with the accurate timings, the singing parts of the popular music need to be detected first. The singing part of the commercial popular music is defined as the *vocal segment*. It consists of the singing voice and other musical instruments such as guitar, keyboard and bass guitar. The *non-vocal segment* defines that the segment consists of musical instruments only. Since the vocal segment consists of both singing voice and musical instruments, it is difficult to determine whether the segment is vocal or not. Furthermore, the traditional automatic speech recognition system cannot be applied directly because the “background noise” from musical instruments is relatively high and the behaviour of singing is very different from that of speech, for example the voiced/unvoiced ratio [17] is increased significantly from speech to singing and the pace of singing is not steady. Therefore, processing commercial popular music signals is a challenging task.

Given a commercially available CD recording and the Cantonese lyrics of the corresponding song, our proposed system aligns the lyrics of each *sentence* (line) for a *section* (verse). A sentence³ defines one input line of the Cantonese lyrics. Typically, a section consists of 4-10 sentences while a sentence contains 4-10 characters. In other words, our system finds the start time and the end time of each lyrics sentence. Since our system is designed for commercial popular music rather than the synthesized audio or pure singing audio signals, our proposed system should be a practical and useful tool.

The major contributions of this paper are:

- We have extended and integrated existing techniques to form the following modules: (1) vocal signal enhancement module, (2) the onset detection module and (3) non-vocal pruning module. They have been applied to commercially available Cantonpop CDs, all contain a mixture of vocal and musical instrument signals, and the onsets and pitches are accurately detected.
- We made use of the tonal characteristic of Cantonese to develop an important feature for lyrics alignment. That is the contour of the musical melody and pitches of the lyrics must match each other. A dynamic time warping algorithm has been successfully applied to align the lyrics with the music signal.

As far as we know, this is the first lyrics alignment system for Cantonpop.

The flow of our system is depicted in figure 1 and the organization of the paper is as follows.

1. Our proposed vocal signal enhancement algorithm is used to suppress the signals of musical instruments and enhance the signal of the singing voice. (Section 3)

2. The start times/onsets of the syllables sung are detected by an onset detection method. (Section 4)
3. The non-vocal onsets are pruned by a singing voice detection classifier which classifies an onset whether is vocal or not. (Section 5)
4. The proposed features are extracted from the lyrics and the audio signal. The features extracted from the lyrics are called *lyrics features* while the features extracted from the audio signal are called *signal features*. (Section 6)
5. The start time and the end time of each lyrics sentence are obtained by the dynamic time warping algorithm which is an alignment algorithm to align an input sequence to a reference sequence. The reference sequence in our system is the lyrics features while the input sequence in our system is the signal features. (Section 7)

After inputting the Cantonese lyrics and the song to our system, the start time and the end time of each lyrics sentence are extracted. Experiments were performed to evaluate the system and the results will be shown in section 8. After that, a conclusion is given in section 9. Before describing the details of the system, a literature review on the addressed problem and some related systems are presented in the next section.

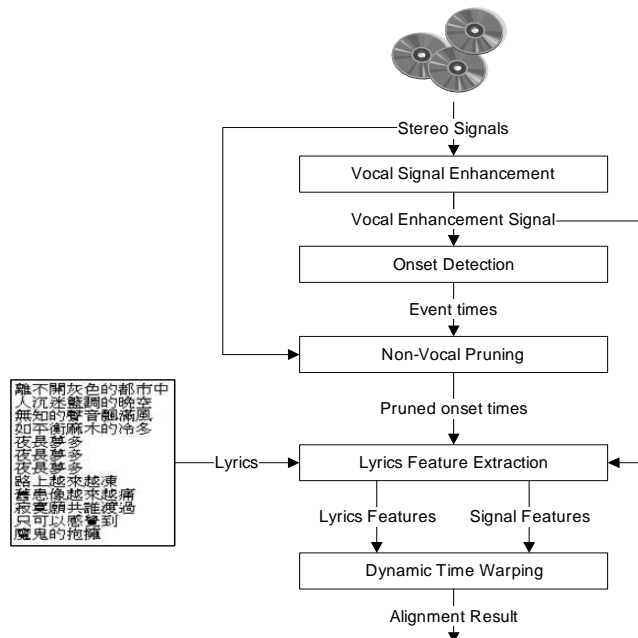


Fig. 1 The block diagram of our proposed system, it consists of five modules including vocal signal enhancement, onset detection, non-vocal pruning, lyrics feature extraction and dynamic time warping.

³ “line” in [29] is equivalent to “sentence” in this paper.

2 Literature Review

The lyrics alignment problem for Cantonpop is divided into certain subproblems, namely singing voice detection and singing transcription. Previous work on these aspects is reviewed here.

2.1 LyricAlly

LyricAlly [29] by Wang et al. is probably the first English lyrics sentence-level alignment system for aligning the textual lyrics to the music signals for a specific structure of popular songs. It first finds the beat of the music in the measure (bar) level and searches for the rough starting point and ending point of each section. Wang et al. defines the five different structural elements of popular music as sections, they are Intro(I), Verse(V), Chorus(C), Bridge(B) and Outro(O). LyricAlly also detects the presence of vocals in the song by using a singing voice detection technique and computes the estimated duration of each sentence of the lyrics by analyzing the duration of each phoneme. Lastly, LyricAlly combines all the information to align each sentence of the lyrics to the song by grouping or partitioning the detected vocal segments to match the estimated duration of each sentence. The current version of LyricAlly is limited to the songs with a specific structure “V-C-V-C-B-O” which covers nearly 40% of all popular songs by observation, and the meter of the songs is limited to 4/4 time signature. Moreover, the authors also point out that the section and vocal detectors are not good enough to handle real-life popular music. A crucial step in LyricAlly is to use the sum of the duration distribution of each phoneme in a sentence to predict the duration of the corresponding sentence being sung. In speech, each phoneme has a certain distribution of duration. However, the durations of phonemes in singing are different. They also depend on the time values of musical notes they belong to, and also the current tempo of the song. Therefore, the duration of a phoneme can vary considerably that may make it unreliable for lyrics alignment.

2.2 Singing Voice Detection

A singing voice detection method is to determine whether an input signal segment contains a singing voice or not. In the work of Adam and Ellis[3], they tried to estimate the music and the vocal segments for the popular music by using posterior probability features with the classifier based on Gaussian mixture model. Namunu et al. [19] proposed twice-iterated composite Fourier transform (TICFT) to detect the singing voice. Tat-Wan et al. [16] make use of the perceptual linear predictive coding (PLP) and the generalized likelihood ratio (GLR) distance to detect the singing voice boundaries, ICA-FX to

reduce the dimension of the features and Support Vector Machine to classify whether the segment is vocal or not. In [20], Tin et al. proposed to use the combination of harmonic content attenuation log frequency power coefficients (HA-LFPC) with HMM to do the classification. The system assumes that the key of the song is known.

2.3 Singing Transcription System

A singing transcription system is to estimate the MIDI pitches of the singing voice from the audio signals. In [7], Clarisse et al. performed a series of experiments to identify the problems of current transcription systems and proposed to use an auditory model based pitch detector called AMPEX (Auditory Model based Pitch Extractor) to transcribe the singing voice. Experiments show that the systems perform better for the melodies sung by humming than sung with lyrics. Another work, in [24], Matti and Klapuri used the fundamental frequency estimator called “YIN” algorithm, which was invented by de Cheveigné and Kawaharain in [9], to extract the pitches and voicing. They also used Hidden Markov Model (HMM) to model the note events (note transient, sustain and silence) and musicological model to track pitches of the singing voice. Note that in both systems, the input audio is a pure singing voice signal for query-by-humming (QBH) system to search and retrieve the music from the database, thus they cannot be applied directly on the real-life popular music as in our case. The difficulty on transcribing the singing voice from the popular music is the complexity of the song which includes different kinds of sounds such as the singing voice, the guitar and the drum. This problem will be addressed by the first module in the proposed system.

3 Vocal Signal Enhancement

The first module in our system is the vocal signal enhancement module. Given the stereo signals from a CD recording of a popular song, its objective is to suppress the signals of the musical instruments and to enhance the signal of the singing voice. The operating principle of this module is based on the mixing practice in popular music industry. In a popular song, the singing voice and the musical instruments are usually recorded separately into different tracks. Then, the music mixer adds different tracks together to become the final product. The industry has a common practice to mix the vocal track and some leading musical instrument tracks at the center position. “*The center is obvious in that the most prominent music element (usually the lead vocal) is panned there, as well as the kick drum, bass guitar and even the snare drum.*” [22] Mixing the track at the center position means that the signal is exactly the same for the left and right channels. Typically, there are only two channels in

an audio CD. Figure 2 shows an example of the recording setting. The singing voice and the drum signals are mixed at the center ($s_c(t)$) while the guitar and the violin are mixed at non-center ($s_{\bar{c},l}(t)$ and $s_{\bar{c},r}(t)$). The left channel $s_l(t)$ and the right channel $s_r(t)$ signals are defined as the following:

$$s_l(t) = s_c(t) + s_{\bar{c},l}(t) \quad (1)$$

and

$$s_r(t) = s_c(t) + s_{\bar{c},r}(t) \quad (2)$$

where t is the time variable, $s_c(t)$ is the center signal, $s_{\bar{c},l}(t)$ and $s_{\bar{c},r}(t)$ are the left channel and the right channel of non-center signal respectively. Typically, the singing voice belongs to the center signal $s_c(t)$.

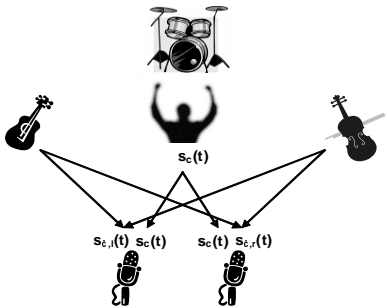


Fig. 2 Recording setting example: the singing voice and the drum signals are at the center while the guitar and the violin are the non-center signal.

Based on the above practice, a vocal signal enhancement algorithm method was proposed to enhance the vocal signal from the stereo recordings. Figure 3 shows the overall idea of enhancing the vocal signal. The “non-center estimation” and “center estimation” were used to extract the center-padded signal. The “bass and drum reduction” was used to enhance the vocal signal by reducing other center-padded musical instrument signals.

3.1 Non-center Signal Estimation

For the stereo signal, the left channel $s_l(t)$ and the right channel $s_r(t)$ signals have been defined as equations 1 and 2. Then by simple subtraction, the estimated non-center signal $\hat{s}_{\bar{c}}(t)$ is obtained:

$$\hat{s}_{\bar{c}}(t) = s_l(t) - s_r(t) = s_{\bar{c},l}(t) + (-s_{\bar{c},r}(t)) \quad (3)$$

Obviously, the center signal $s_c(t)$ is eliminated by simple subtraction. In fact, it is a common method in many commercial software and hardware such as Goldwave[13] to obtain the reduced vocal channel for a simple Karaoke system.

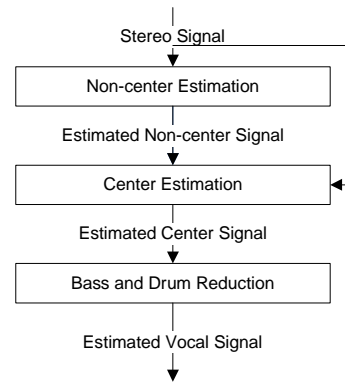


Fig. 3 Vocal signal enhancement block diagram. The algorithm is divided into three parts, they are non-center estimation, center estimation and bass and drum reduction.

3.2 Center Signal Estimation

We found that time domain methods cannot extract the center signal $s_c(t)$ from $s_l(t)$ and $s_r(t)$. Thus, non-linear spectral subtraction (NLSS) [28] is introduced to extract the center signal. The subtraction of the NLSS is operated in the magnitude spectrum domain. In the literature, NLSS is used to reduce the noise for speech recognition [4] by subtracting the magnitude spectrum of the signal from the average magnitude spectrum of the signal. In this work, we used the concept of the NLSS in which the subtraction is executed in the magnitude spectrum domain. For extracting the center signal, the system subtracts the magnitude spectrum of the original signal $s(t)$ from the magnitude spectrum of the estimated non-center which is obtained in the previous section (section 3.1).

By applying short-time Fourier transform \mathcal{F} on the non-center signal (equation 3) and the both channels of original signal, we get

$$\mathcal{F}\{\hat{s}_{\bar{c}}(t)\} = \hat{S}_{\bar{c}}(w) = S_{\bar{c},l}(w) + (-S_{\bar{c},r}(w)) \quad (4)$$

$$\mathcal{F}\{s_l(t)\} = S_l(w) = S_c(w) + S_{\bar{c},l}(w) \quad (5)$$

$$\mathcal{F}\{s_r(t)\} = S_r(w) = S_c(w) + S_{\bar{c},r}(w) \quad (6)$$

where w is the frequency variable, $\hat{S}_{\bar{c}}(w)$ is the spectrum of estimated non-center signal, $S_l(w)$ and $S_r(w)$ are the spectrums of the left and right channels respectively. In this paper, the window size is set to 125ms with 88% overlapping. Then by taking the absolute and approximation of both sides of equations 4, 5 and 6, the equations become

$$|\hat{S}_{\bar{c}}(w)| \approx |S_{\bar{c},l}(w)| + |S_{\bar{c},r}(w)| \quad (7)$$

$$|S_l(w)| \approx |S_c(w)| + |S_{\bar{c},l}(w)| \quad (8)$$

$$|S_r(w)| \approx |S_c(w)| + |S_{\bar{c},r}(w)| \quad (9)$$

where $|\hat{S}_{\bar{c}}(w)|$ is the magnitude spectrum of the estimated non-center signal, $|S_l(w)|$ and $|S_r(w)|$ are the magnitude spectrums of left and right channels respectively.

Next, we mix the spectrums of left and right channels as the following:

$$|S(w)| = 2|S_c(w)| + (|S_{\bar{c},l}(w)| + |S_{\bar{c},r}(w)|) \quad (10)$$

where $|S(w)|$ is the magnitude spectrum of the mixed signal.

By the method of spectral subtraction, the estimated modulus of the center signal can be obtained by subtracting the mixed signal from the estimated non-center signal in frequency domain:

$$\begin{aligned} |\hat{S}_c(w)| &= \frac{1}{2}|S(w)| - \frac{1}{2}|\hat{S}_{\bar{c}}(w)| \\ &\approx \frac{1}{2}(2|S_c(w)| + |S_{\bar{c},l}(w)| + |S_{\bar{c},r}(w)|) \\ &\quad - \frac{1}{2}(|S_{\bar{c},l}(w)| + |S_{\bar{c},r}(w)|) \\ &= |S_c(w)| \end{aligned} \quad (11)$$

where $|\hat{S}_c(w)|$ is the magnitude spectrum of the estimated center signal. Then by applying the inverse Fourier transform with magnitude $|\hat{S}_c(w)|$ and the original phase of the signal $s_l(t)$, i.e. $\angle \mathcal{F}\{s_l(t)\}$, we can obtain the estimated center signal $\hat{s}_c(t)$.

3.3 Bass and Drum Reduction

In pop music, besides the vocal, the center signal $s_c(t)$ also consists of some other lead musical instruments such as the bass guitar and drum. These two instruments usually are more the less stationary in a short period in the frequency domain while the vocal part is not.

Therefore, we segment $\hat{s}_c(t)$ into N segments with period T . Within each segment i (typically 2s), the average spectrum $|\bar{S}_i(w)|$ can be computed by averaging the frequency components in that segment. Then, we apply the method of spectral subtraction again to each segment i which is subtracted by the average spectrum $|\bar{S}_i(w)|$. Lastly, a highpass filter is used to filter the frequency components of the bass guitar. After that, we obtain the estimated vocal signal $\hat{s}_v(t)$.

4 Onset Detection

Onset Detection or Event Detection is to detect the start time of each event. For music transcription, onset detection detects the start time of each note played by the performer. For this system, the start time of each lyrics character located in the signal is found by analyzing the enhanced vocal signal $\hat{s}_v(t)$. The onset detection algorithm presented here is similar to the algorithm proposed by Scheirer [27] and Klapuri [15] but some modifications and post-processing are introduced. The algorithm is divided into three parts as shown in figure 4. First, the amplitude envelope of the signal is extracted. Then, a relative difference function [15] is used as a cue to detect the onsets. Lastly, simple peak picking operation, thresholding and omitting window operation are introduced in the post-processing module to extract the onsets.

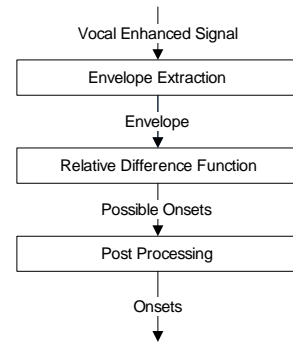


Fig. 4 Onset detection block diagram. The algorithm is divided into three parts, they are envelope extraction, relative difference function and post processing.

4.1 Envelope Extraction

Amplitude envelope is one of the cues to detect changes for human auditory system. The input signal is first rectified (by taking the absolute value), then the envelope is calculated by averaging the value with an onset window size w^{onset} as the following:

$$E_j = \frac{1}{w^{onset}} \sum_{t=t_j}^{t_j+w^{onset}-1} |s(t)| \quad (12)$$

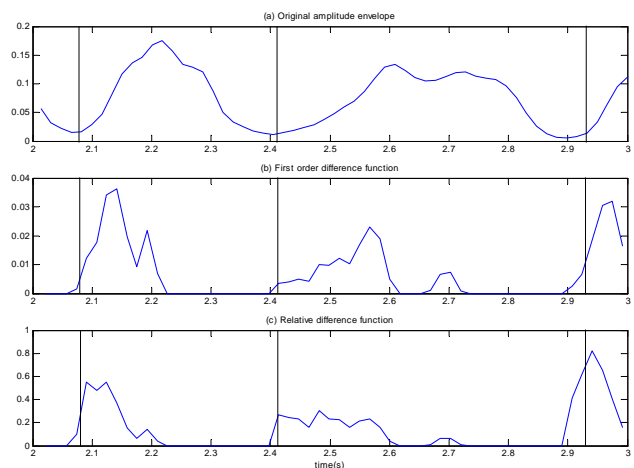


Fig. 5 (a) Original amplitude envelope, (b) first order difference function and (c) relative difference function. The vertical lines are the manually input onsets. The peak of the relative difference function has faster response than that of the first order difference function.

4.2 Relative Difference Function

Scheirer used the first order difference function δE_j to detect the changes for the envelope.

$$\Delta E_j = E_{j+1} - E_j \quad (13)$$

However, there are two problems that are described in Klapuri's master thesis [15]. First, the amplitude may need some time to increase to its maximum point but this point is too late from the start time of the event (shown in figure 5). Second, the signal may not always increase monotonically, so there are several local maximum values exist at the same event (shown in figure 5). So Klapuri proposed a method called first order relative difference function δE_j to handle these problems. It is calculated by ΔE_j divided by its original envelope value E_j . By simple mathematical manipulation, it is the same as the first order difference of the logarithm of the amplitude envelope ($\frac{ds(t)}{dt}/s(t) = \frac{d\log(s(t))}{dt}$). The first order relative difference function is defined as following:

$$\delta E_j = \log E_{j+1} - \log E_j \quad (14)$$

As shown in figure 5, the peak value is much closer the start time of the event and the global maximum can be clearly distinguished from other local maxima. So, the benefit of using the relative difference function is significant. For our system, just the positive value of the difference function is considered because the start time is just caused by the positive changes.

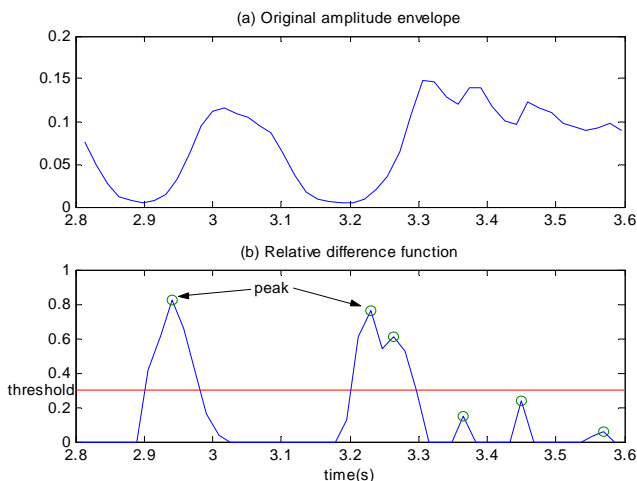


Fig. 6 Demonstration of peak picking operation (circle) and thresholding (horizontal line) in post processing. The figure shows (a) the original amplitude envelope and (b) the relative difference function and the operations in post processing. The algorithm finds the peaks first. Then the peaks below the threshold are pruned.

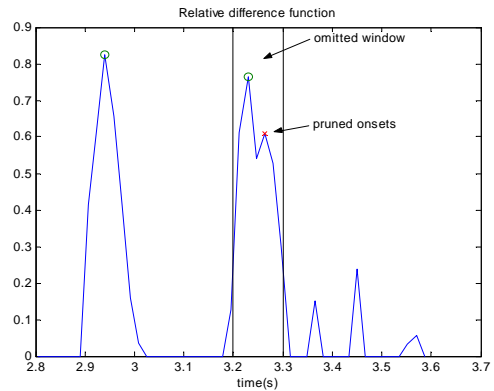


Fig. 7 Demonstration of omitting window operation. Within the omitting window (the region between two vertical lines), the lower value onset (cross) is pruned.

4.3 Post-Processing

Next, there are three more steps applying on the relative difference signal for extracting the onsets. First, a simple peak picking operation is applied to the relative difference signal. The peak picking operation picks all points that are larger than its neighbors as shown in figure 6. After applying the peak picking operation, all the points greater than a threshold ϵ_{onset} are considered as potential onsets as shown in figure 6. Lastly, if the potential onsets are too close with each other, the lower value onset(s) will be pruned because in reality the singer cannot sing too fast. The omitted window size w^{omit} is used to prune the onsets as shown in figure 7. Then $EventTime_k$ is the time of onset k . For optimal performance, the omitting window size, onset window size, and threshold are set to 150ms, 50ms and 0.05 respectively⁴.

5 Non-vocal Pruning

After applying onset detection, most onset times of singing words are extracted. However, there are some extracted onset times which are in fact not vocal. So, pruning non-vocal onsets are necessary in order to enhance the performance of the system. For pruning non-vocal onsets, we need a classifier to determine whether the onset is vocal or not. Six different types of features are chosen for vocal classification. We used these features as the inputs to a multiple-layer perceptron (MLP) neural network [2] to classify whether a segment is vocal or not. The six different types of features are explained below:

Spectrum flux Spectrum flux [18] is to measure the change of the spectrums between two consecutive frames. It is useful for vocal/non-vocal classification because vocal signals usually have greater changes between two

⁴ Due to limited space, the experimental result is put in the appendix available at <http://www.cse.cuhk.edu.hk/~khwong/demo/lyricsalign/demo.htm>

consecutive frames. Spectrum flux is defined as the 2-norm of their spectrum difference. Besides the original spectrum flux, the variance of the spectrum flux is also used as another feature for our system.

Harmonic coefficient For the voicing part of speech, the harmonics, which is measured by the energy of integer multiple of the fundamental frequency, are very rich. In [6], Chou showed that harmonic coefficients can capture this phenomenon. Given a discrete signal, harmonic coefficient is the maximum of the sum of its temporal autocorrelation sequence and its spectral autocorrelation sequence. Typically, vocal segments with a voicing part are relatively high in harmonic coefficients because the harmonic content is rich in the voicing part.

Zero crossing rate Three variations of zero crossing rate [26] are used as our features including the delta zero crossing rate, variance of delta zero crossing rate, and high zero crossing rate ratio [18].

Amplitude envelope The log amplitude envelope is discussed in the section 4.1. In addition to log amplitude envelope, our system uses the first and second order difference of the amplitude envelope as features. The amplitude envelope is used to differentiate silence and non-silence segments.

Mel-frequency cepstral coefficients (MFCC) Mel-frequency cepstral coefficients (MFCC) [6,12] and the first difference of MFCC are used as the features.

4-Hz modulation energy 4-Hz modulation energy [23] is a characteristic energy measure of speech signals. In [6], Chou stated that the 4-Hz modulation energy is effective for singing detection. For computing the 4-Hz modulation energy, energies of 40 perceptual channels of mel-frequency are first extracted. Discrete fourier transform with 1-Hz frequency resolution is applied on each channel. Then the summation of the channel energies of the 4-Hz is extracted to be the 4-Hz modulation energy feature. The 4-Hz modulation energy is relatively higher in the vocal segments.

Vocal segments and non-vocal segments probably have their distinguishable regions in the space of these features. In order to find the mapping between the feature space and the class labels (vocal/non-vocal), these features are inputted to MLP neural network. The network used in this paper has one hidden layers. The ‘tanh’ function is used as the activation function of the hidden nodes of the networks in the experiments. And the activation function of the output nodes is sigmoid function. The number of hidden units is chosen to be 9 after testing.

6 Lyrics Feature Extraction

In this section, we propose and describe the features which are to be used in the Dynamic Time Warping (DTW) algorithm to be introduced in the next section (section 7). These features are relative pitch features and

time distance features. The DTW algorithm uses them to align the lyrics with the correct timings.

Figure 8 shows the inputs and outputs of lyrics feature extraction module. The inputs are the lyrics, the event times which are obtained from the non-vocal pruning module described in section 5, and the vocal enhanced signal which is extracted by the vocal signal enhancement algorithm described in section 3. The outputs of the module are *lyrics features* and *signal features*. The lyrics features are the features extracted from the input lyrics while the signal features are extracted from the vocal enhanced signal and the event times. Then, the DTW algorithm aligns the signal feature vector to the lyrics feature vector.

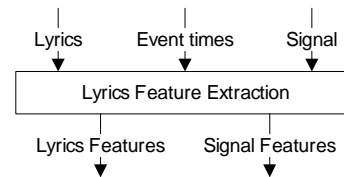


Fig. 8 Input-output diagram of lyrics feature extraction module. The lyrics features and the signal features are extracted by this module.

6.1 Features

6.1.1 Relative Pitch Feature

As described in section 1, in order to convey the meaning of the lyrics accurately, the contour of the melody and that of the lyrics must match each other. Therefore, the relative pitch feature of Cantonese syllables is significant for lyrics alignment. Figure 9 shows the idea of the relative pitch matching of a Cantonese song. In our system, we follow the categorization of 6 tones of Cantonese into 3 groups in [5]: high, mid and low pitches. Each group is called *lyrics pitch LP*.

For calculating the relative pitch feature of the lyrics feature vector (which contains the features extracted from the input lyrics), we can assign the pitch class to get the lyrics pitch LP_l (larger number, higher pitch) for each lyrics character l . Then the relative pitch feature of lyrics features LRP_l is calculated by applying a simple first order difference function except for the starting character of each sentence.

$$LRP_l = \begin{cases} 0, & \text{if starting word} \\ LP_l - LP_{l-1}, & \text{otherwise} \end{cases} \quad (15)$$

Figure 10 shows the block diagram of calculating the relative pitch feature of the signal features. First, the pitch extraction process is to extract the fundamental frequency of each event of the signal. This will be presented in section 6.2. After that, all the pitches (f_{q_k}) of

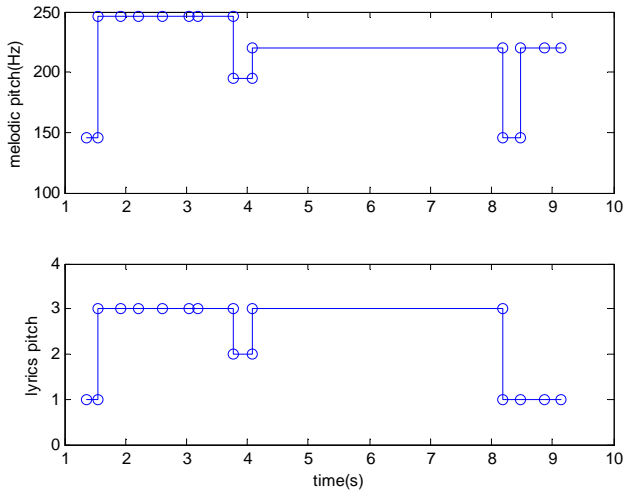


Fig. 9 The idea of relative pitch matching of the melody. Each lyrics character is denoted by the circle. The graph shows that the lyrics pitch is higher when the melodic pitch is higher.

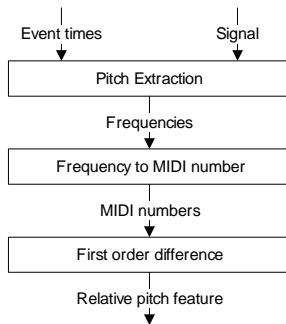


Fig. 10 Block diagram of calculating the relative pitch feature of the signal features. Pitch Recognition Algorithm is applied to extract the pitches from the signal according to the event times first. Then the frequencies are converted to MIDI number. Lastly, the first order difference is applied to get the relative pitch feature.

each event time are recognized. Then the frequencies are converted into MIDI numbers by the following equation:

$$MIDI_k = 69 + 12 * \log_2(fq_k/440) \quad (16)$$

where $MIDI_k$ is the MIDI number of event k , 69 is the MIDI number of A4 note and 440Hz is the fundamental frequency of A4 note.

Lastly, the first order difference is applied to the MIDI numbers to get the relative pitch feature of the signal:

$$SRP_k = \begin{cases} 0, & \text{if } \hat{e}v_k - \hat{e}v_{k-1} > \epsilon^{time} \\ MIDI_k - MIDI_{k-1}, & \text{otherwise} \end{cases} \quad (17)$$

where SRP_k and $\hat{e}v_k$ are the relative signal pitch feature of event k and estimated event time k , respectively. ϵ^{time} is the time threshold to separate the sentences. In this work, we chose 500ms as the time threshold.

6.1.2 Time Distance Feature

The time distance feature is another metric for the DTW algorithm. The rationale of using time distance feature is that the time distance between the last character of a sentence and the first character of the following sentence is generally longer than the time distance of the characters within a sentence. Therefore, the time distance feature is also an important measure for the DTW algorithm for time alignment.

The time distance feature of the lyrics feature and signal feature can be obtained directly by the following equations:

$$LD_l = \begin{cases} 4, & \text{if starting word} \\ 1, & \text{otherwise} \end{cases} \quad (18)$$

where LD_l is the time distance feature of the l^{th} character of the lyrics. The number 4 is chosen because the maximum difference between two relative pitches is also 4 (the relative pitch can be -2, -1, 0, 1 and 2).

$$SD_k = \begin{cases} 4, & \text{if } \hat{e}v_k - \hat{e}v_{k-1} > \epsilon^{time} \\ 1, & \text{otherwise} \end{cases} \quad (19)$$

where SD_k and ev_k are the time distance feature of event k and estimated event time k , respectively. And ϵ^{time} is the threshold to separate the sentences.

Finally, the relative pitch and the distance features are grouped together to become the lyrics features(L) and the signal features(S):

$$L_l = (LRP_l, LD_l) \quad (20)$$

$$S_k = (SRP_k, SD_k) \quad (21)$$

6.2 Pitch Extraction

There are two steps to extract the pitch from the signal and the event times. First, the fundamental frequency (f_0) detection algorithm called YIN in [9] is applied on each of successive frames to obtain the preliminary result. Then, a simple post-processing algorithm is proposed here to produce the final f_0 frequencies because f_0 detected by YIN algorithm may have *octave* errors. Two musical note is in octave if the f_0 of the upper note is double that of the lower note. The goal of the simple post-processing method is to overcome this problem. In pop music, the pitches of the melodies seldom change significantly for more than an octave. For example the melodic change from A4 (440Hz) to B5 (988Hz) is greater than an octave. So, if the pitch difference between two consecutive pitches is more than an octave, the detected pitch of the latter one needs to be modified to the pitch with an octave to the direction of the former one so that the pitch difference is below an octave. Then the octave error of YIN algorithm can be solved. For example, if two consecutive detected pitches are C4 and G5, then they become C4 and G4 after post-processing. If the melodic

change is actually greater than an octave, which rarely occurs in Cantopop, this post-processing step will cause two errors at maximum in the feature vector. The experiments in the section 8.2.3 showed that the alignment algorithm was robust to such kind of errors. Furthermore, to improve the performance of the algorithm, 5 consecutive windows (80% overlapping) were used instead of 1 window for extracting the pitch. The pitch with best voicing value within 5 consecutive windows was chosen as the recognized pitch. The voicing value, which is an output from YIN algorithm, is a confidence indicator of the reliability of the fundamental frequency estimate.

7 Lyrics Alignment

7.1 Dynamic Time Warping

Dynamic Time Warping (DTW) has been used widely in the area of automatic speech recognition [25, 1]. The DTW algorithm is a robust algorithm for aligning two sequences by evaluating the error function. In this work, the DTW algorithm is used to align the lyrics feature sequence (L , equation 20) to the signal feature sequence (S , equation 21) in order to find the optimum time alignments of the provided lyrics. Figure 11 shows the idea of the alignments from the DTW algorithm.

In DTW, the error matrix (E^{dtw}) between two sequence is computed first.

$$E_{i,j}^{dtw} = Distance(L_i, S_j) \quad (22)$$

where $Distance(\mathbf{v}_1, \mathbf{v}_2)$ is the distance function between two vectors, and L and S are the lyrics features and signal features, respectively. In this work, the distance metric is chosen to be the city block distance:

$$Distance(\mathbf{v}_1, \mathbf{v}_2) = \sum_{i=1}^N (|\mathbf{v}_1(i) - \mathbf{v}_2(i)|) \quad (23)$$

where N is the dimension of the vector \mathbf{v} and $\mathbf{v}(i)$ denotes the i^{th} dimension value of vector \mathbf{v} . In our case, N is equal to 2. Then the accumulated error matrix (EA^{dtw}) is calculated by:

$$EA_{1,1}^{dtw} = E_{1,1}^{dtw} \quad (24)$$

$$EA_{i,j}^{dtw} = \min(EA_{i,j-1}^{dtw} + E_{i,j}^{dtw} + w^{dtw}, \\ EA_{i-1,j}^{dtw} + E_{i,j}^{dtw} + w^{dtw}, \\ EA_{i-1,j-1}^{dtw} + E_{i,j}^{dtw} + \sqrt{2}w^{dtw}) \quad (25)$$

where w^{dtw} is the weighting factor against the feature vectors and it is set to 4, and the value $\sqrt{2}$ is to compensate the distance from the diagonal direction.

The accumulated error matrix is obtained from the minimum of three directions including left, bottom and bottom-left. For example, given the signal feature sequence $S = \langle (0, 4), (0, 4), (0, 4), (0, 4) \rangle$ and the lyrics feature sequence $L = \langle (0, 4), (0, 1), (0, 1) \rangle$, we calculate the

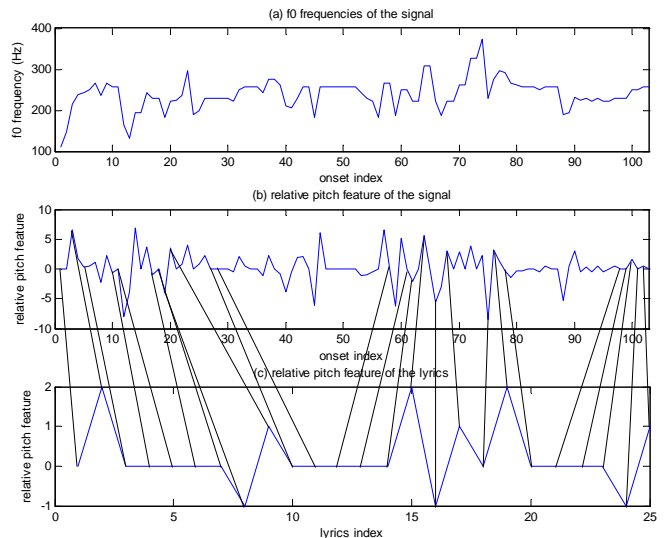


Fig. 11 (a) Detected f0 frequencies of the signal, (b) relative pitch features of the signal and (c) relative pitch features of the lyrics. The f0 frequencies of the signal are detected by the f0 detection algorithm described in section 6. The relative pitch features of the signal are computed by equations 16 and 17. The relative pitch features of the lyrics are obtained from equation 15. The DTW algorithm aligns the relative pitch features of the lyrics to that of the signal as shown by the lines. Although there are spurious onsets (onset indexes 34-56 and 81-95), the DTW algorithm aligns the lyrics robustly for all three sentences.

accumulated error matrix (EA^{dtw}) and the direction matrix as shown in figures 12(a) and (b) respectively. The direction matrix indicates that the direction gives the minimum accumulated error in each entry of the accumulated error matrix. Then, we backtrack the accumulated error matrix from the end to the starting point by the reverse direction of the direction matrix (the entries bracketed with “()” in figure 12(a)). Lastly, we choose the first hit of the lyrics feature in the backtracked path as the alignment between a lyrics feature and a signal feature (the entries bracketed with “[]” in figure 12(a)). The start time of a lyrics character is the onset of the signal feature which is aligned with the lyrics feature of that character. For example, if the onsets of the signal features in figure 12(a) are at the time 656ms, 1356ms, 1881ms, and 2394ms, the start time of the three corresponding lyrics characters of the lyrics feature sequence is 656ms, 1881ms, and 2394ms.

8 Experiments

8.1 Experimental Setup

To evaluate the accuracy of our system, experiments were performed on 14 different songs in 7 different albums as shown in table 1. All the albums are sung by

Lyrics\Signal	(0,4)	(0,4)	(0,4)	(0,4)
(0,4)	[(0)]	(4)	8	12
(0,1)	7	8.66	[(12.66)]	16.66
(0,1)	14	15.66	17.31	[(21.31)]
Lyrics\Signal	(0,4)	(0,4)	(0,4)	(0,4)
(0,4)	-	3	3	3
(0,1)	2	1	1	1
(0,1)	2	1	1	1

Fig. 12 DTW example with the signal feature sequence $S = \langle (0,4), (0,4), (0,4), (0,4) \rangle$ and the lyrics feature sequence $L = \langle (0,4), (0,1), (0,1) \rangle$. (a) The accumulated error matrix (EA^{dtw}). Backtracked DTW path contains the entries bracketed with “()”. The alignment between a lyrics feature and a signal feature is an entry bracketed with “[]”. (b) The direction matrix. The numbers “1”, “2” and “3” correspond to the diagonal, vertical and horizontal directions respectively.

different singers. The tempos of the songs vary from 56 to 160 beats per minute (bpm). Before performing the experiments, the songs were re-sampled from 44,100Hz to 8,000Hz by the software *Goldwave* [13]. The wave format is 8,000Hz sampling rate, 16-bit and stereo. There were 70 segments with 20 seconds long (total 1400 seconds). The total number of syllables sung was 2670.

Album	Singer	Tempo (Song 1)	Tempo (Song 2)
Real Feeling (1992)	Jacky Cheung	89	94
Beyond Life (1996)	Beyond	69	76
Can’t Relax (1996)	Sammi Cheng	56	88
Hacken Best 17 (1997)	Hacken Lee	74	68
Bliss (1999)	Eason Chan	70	92
Being (2004)	Paul Wong	91	160
Picassa’s Horse (2004)	Steve Wong	106	75

Table 1 Albums used in our experiment. The tempos of the songs are varying from 56 to 160 in the unit of beats per minute (bpm). There are 70 testing 20-second segments and 2670 syllables sung in total.

The lyrics were entered sentence by sentence manually and then lyrics pitch of each character was obtained automatically from the online Cantonese dictionary⁵ in [11]. The lyrics features were computed by equation 20.

⁵ Given a Chinese character, the dictionary returns one of the six Cantonese tones denoted by the integers from 1 to 6 specified by the transcription system of LSHK [21]. According

To evaluate the non-vocal pruning classifier, we trained the neural network classifier by the cross-validation training method to avoid overfitting. First, we marked all the vocal and non-vocal segments manually on the 70 segments. The training set, the validation set and the test set contained 35 segments, 25 segments and 20 segments respectively. All the three sets were disjointed in the song level, i.e. a song could only be either in the training set, the test set or the validation set. And also, within the set, number of vocal segments and number of non-vocal segments were the same in order to train the neural network fairly. The neural network classifier was trained and tested 50 times in order to obtain more accurate result since the weights of the networks were chosen randomly before training.

Before discussing the results, two metrics of accuracy are defined below. A sentence defines a group of characters which is segmented by the distance feature equal to 4. Assume (S_i^s, S_i^e) is the time range of actual duration in the songs of the i^{th} sentence. $(\hat{S}_i^s, \hat{S}_i^e)$ is the estimated time range of the i^{th} sentence by the system. The start time \hat{S}_i^s is the onset of the first character in the sentence i . The end time \hat{S}_i^e is set to the start time of the sentence $i + 1$ because it is probably acceptable that the lyrics of the i^{th} sentence is still being displayed during the gap between the i^{th} and the $(i + 1)^{th}$ sentences. The end time of the last sentence is set to the end time of its corresponding segment.

Two types of accuracy are defined to evaluate the system. The first type is “In-Range Accuracy”:

$$A_i^R = \frac{Range((S_i^s, S_i^e) \cap (\hat{S}_i^s, \hat{S}_i^e))}{Range((S_i^s, S_i^e))} \times 100\% \quad (26)$$

where A_i^R is “In-Range Accuracy” of i^{th} sentence and $Range(x, y) = y - x$. The rationale of the “In-Range Accuracy” is that the particular lyrics must be displayed when the singer is singing that lyrics. For example, if the duration of a sentence is 4 seconds (which is the typical duration), 80% In-Range Accuracy means that 3.2 seconds of the lyrics sentence is displayed when the singer is singing that 4-second sentence.

The second type of accuracy is “Duration Accuracy”:

$$A_i^D = \frac{Range((S_i^s, S_i^e) \cap (\hat{S}_i^s, \hat{S}_i^e))}{Range((S_i^s, S_i^e) \cup (\hat{S}_i^s, \hat{S}_i^e))} \times 100\% \quad (27)$$

where A_i^D is “Duration Accuracy” of i^{th} sentence and $Range(x, y) = y - x$. The rationale of the “Duration Accuracy” is that the duration of particular lyrics displayed must be the same as the duration of the lyrics the singer sung.

Figure 13 shows the graphical explanation of both accuracies. The numerators of both accuracies, which is the intersection region, are the same (figure 13(a)).

to [5], tone numbers 1 and 2 belong to high pitch, 3 and 5 belong to mid pitch, and 4 and 6 belong to low pitch.

The difference between both accuracies is the denominators. The denominator of “In-Range Accuracy” is the actual interval (figure 13(b)). This accuracy measures how much the estimated interval is correct during the singer singing that sentence. On the other hand, the denominator of “Duration Accuracy” is the union region (figure 13(c)). This accuracy shows how much the system estimates the duration of the actual time range correctly.

Using both “In-Range Accuracy” and “Duration Accuracy” is to ensure the system can be evaluated properly. According to Figure 13, “Duration Accuracy” is always smaller than or equal to “In-Range Accuracy”. At first glance, it seems that “In-Range Accuracy” is unnecessary. However, as mentioned before, it is probably acceptable that the lyrics of the current sentence is still being displayed during the gap between the current and the next sentences so the start time of the next sentence is used as the end time of the current sentence in the system. As a result, the end time of each sentence is not estimated accurately and it is usually greater than the actual end time. The performance of the system would be underestimated if only “Duration Accuracy” is used. On the other hand, “In-Range Accuracy” can settle this issue but if the estimated interval is wider than the actual interval of a lyrics sentence, its “In-Range Accuracy” is 100%. However, it is very unlikely that such case will cause the overall “In-Range Accuracy” to be overestimated because this wide estimated interval probably shortens its adjacent estimated intervals and the gap between two sentences is usually short. Thus, 100% accuracy for this particular sentence certainly decreases the accuracy its adjacent sentences. To take the advantages of both metrics, both “In-Range Accuracy” and “Duration Accuracy” are included in evaluation.

8.2 Results and Discussion

8.2.1 Performance of vocal signal enhancement, onset detection, and non-vocal pruning

Comparing to the manually found onsets, the onset detection module performed with the hit rate (number of true onsets detected / number of true onsets) 89%, and the false alarm rate (number of onsets spuriously detected / number of onsets detected) 56%. The reason for the high false alarm rate is that the vocal enhancement method could not remove all the non-vocal instruments, thus there were many non-vocal onsets. Therefore, non-vocal pruning was introduced to handle this issue. The classification accuracy of the non-vocal pruning classifier is about 80% (75% for non-vocal segments, 81% for vocal segments) after vocal signal enhancement. The network classified vocal segments better than non-vocal segments because the variation between non-vocal segments (silence, guitar, bass, drum, etc.) is larger than that between vocal segments. Result also showed that

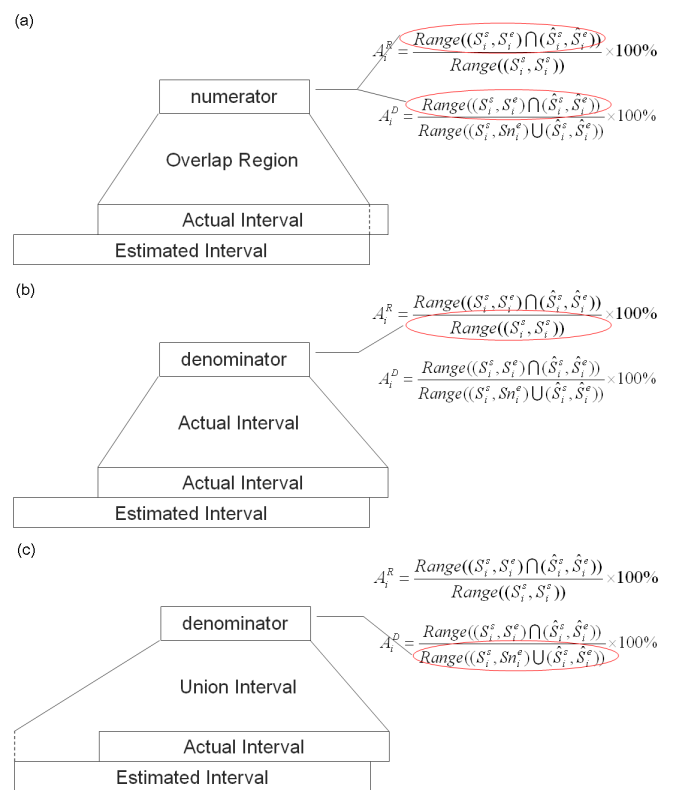


Fig. 13 Graphical explanation of In-Range Accuracy A^R and Duration Accuracy A^D . (a) The numerators of both accuracies are the duration of the intersection region of the actual interval and the estimated interval. On the other hand, (b) the denominator of the In-Range Accuracy is the duration of the actual interval while (c) that of the Duration Accuracy is the duration of the union interval of the actual interval and the estimated interval.

the classification accuracy using the vocal enhanced signals was (about 3%) better than that using the original signals. Since the vocal enhancement algorithm reduced the non-vocal signals significantly and maintained the level of vocal signals, the vocal enhancement algorithm was effectively acted as the preprocessing step before applying the classifier.

8.2.2 Benchmark performance of DTW

To evaluate the benchmark performance of the system, the manually found onsets and pitches were used. According to equation 21, the system computed the benchmark signal features S from all these onsets and pitches. The lyrics features L and the benchmark signal features S were applied on the dynamic time warping algorithm.

Table 2 shows the benchmark performance of the system. The average of “In-Range Accuracy” was about 94%, the system could not align perfectly(100%) because relative pitch features were used in both of the lyrics features and the signal features. In some cases, for example, the two successive characters in the lyrics match the

musical interval “DO”-“RE” in the melody, if there is “RE”-“MI” nearby, DTW may incorrectly match these two characters to “RE”-“MI” because both “DO”-“RE” and “RE”-“MI” are ascending major 2nds and have the same relative pitch feature.

The average of “Duration Accuracy” was about 75%. It was much lower than “In-Range Accuracy” because the system used the start time of the next sentence as the end time of the current sentence so the end time of each sentence was not estimated accurately. In real application, the system is acceptable if it can align the lyrics about 80% in “In-Range Accuracy”.

	In-Range Accuracy	Duration Accuracy
	A^R (%)	A^D (%)
mean	94.30	75.38
min	67.09	49.56
max	99.75	93.51
standard deviation	7.43	9.28

Table 2 Alignment accuracy of the benchmark performance.

8.2.3 Robustness of the DTW algorithm

To evaluate the robustness of the DTW algorithm, three kinds of noises were added to the benchmark signal features, the noises were “semitone noise”, “extra onset noise” and “pruning onset noise”.

The “semitone noise” defines adding some semitone errors probabilistically to the benchmark signal features. For example, the MIDI number of the first onset is 69 originally, then ± 1 , ± 2 or ± 3 semitone(s) error is added probabilistically, thus the MIDI number becomes either 66, 67, 68, 70, 71 or 72. The “semitone noise” was used to simulate the pitch detection error and observe the behaviour of the DTW algorithm. The experiments were performed with different probability to add the “semitone noise” from 0.05 to 0.5. For example, if the probability is 0.5, there is 50% chance adding ± 1 , ± 2 or ± 3 semitone(s) to an onset. For each probability value, we tested the DTW algorithm 50 times to find the average performance of each song in order to get a more accurate result.

Figure 14 shows the “In-Range Accuracy” and the Duration Accuracy after adding the “semitone noise”. The DTW algorithm aligned the lyrics robustly. The result was similar to the benchmark performance of the system, thus the DTW algorithm could tolerate the errors which were introduced in the pitch detection module of the system.

The “spurious onset noise” defines adding spurious onsets to the benchmark signal features. For example, there were 40 original onsets in the benchmark signal features,

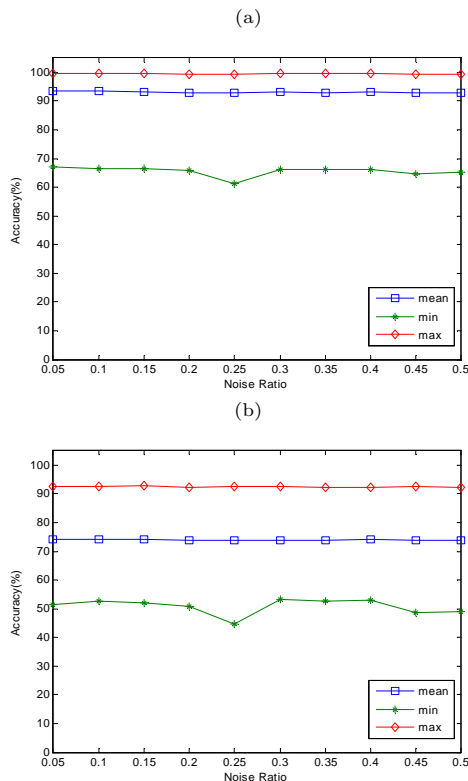


Fig. 14 Alignment accuracy (a)In-Range Accuracy A^R and (b)Duration Accuracy A^D of adding semitone noise.

4 spurious onsets (10% noise ratio) are added randomly while the pitches of these spurious onsets are chosen as the same as the previous onsets. For instance, a spurious onset is added between the 3rd and 4th onsets, the pitch of this spurious onset is the same as that of 3rd one. The “spurious onset noise” was used to simulate the false alarm errors which were introduced from the onset detection algorithm. Similar to the previous experiments, the experiments were performed with different noise ratios from 0.05 to 0.5. For example, if the probability is 0.1 and the number of onsets is 40, $40 \times 0.1 = 4$ spurious onsets would be added. For each noise ratio, we also tested the DTW algorithm 50 times.

Figure 15 shows the “In-Range Accuracy” and the Duration Accuracy after adding the spurious onsets. The DTW algorithm could align the lyrics robustly even the noise ratio was 1, i.e. 20 spurious onsets with 40 original onsets. The accuracy dropped from 93% to 88%, 5% dropped after 50% spurious onsets, thus the DTW algorithm could compensate the false alarm errors which were introduced from the onset detection algorithm of the system.

The “pruning onset noise” defines pruning the onsets from the benchmark signal features. For example, there were 40 original onsets in the benchmark signal features, 4 onsets (10% noise ratio) are deleted from the bench-

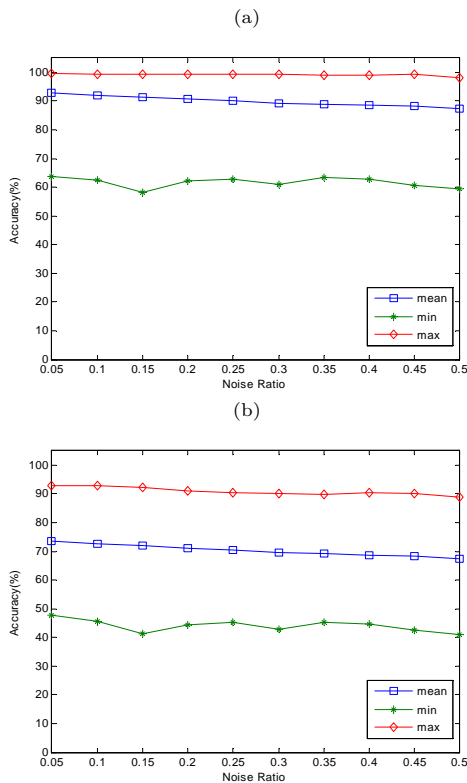


Fig. 15 Alignment accuracy (a)In-Range Accuracy A^R and (b)Duration Accuracy A^D of adding spurious onset noise.

mark signal features randomly. The “pruning onset noise” was used to simulate the missing onset errors from the onset detection algorithm. Similar to all previous experiments, the experiments were performed with different noise ratios from 0.05 to 0.5. We also tested the DTW algorithm 50 times for each noise ratio.

Figure 16 shows the “In-Range Accuracy” and the Duration Accuracy after pruning the onsets. The DTW algorithm could align the lyrics robustly (about 80% accuracy) if the noise ratio was not too large (≤ 0.25), but the alignments were bad when the noise ratio was larger than 0.25. The accuracy dropped from 91% to 58%, 33% dropped after pruning 50% onsets.

To conclude, the DTW algorithm could align the lyrics robustly even if the pitch detection module contains some semitone errors, the onset detection module contains some false alarm errors but not too many missing onset errors.

8.2.4 Overall Performance

Two data sets were applied on the proposed system. Data set *A* contains 20 segments which were not used in training the non-vocal pruning classifier in Section 8.1. These 20 segments are from the 4 songs in the albums “Beyond Life” and “Bliss” in Table 1. The range of the song tempo is from 69 to 92. In addition to these 20 segments, data

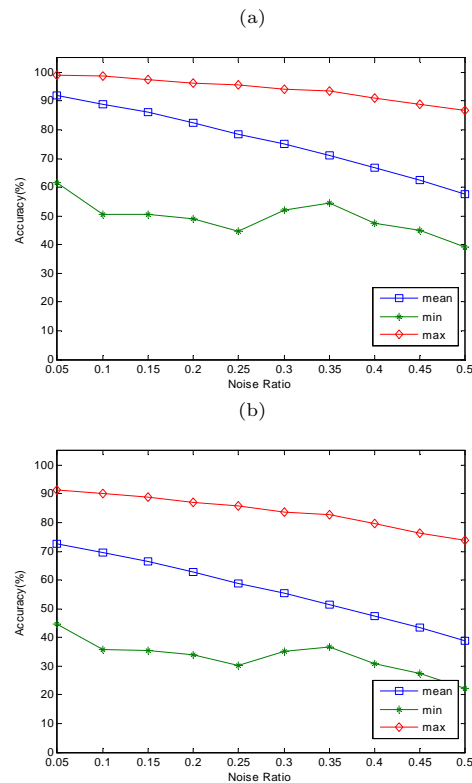


Fig. 16 Alignment accuracy (a)In-Range Accuracy A^R and (b)Duration Accuracy A^D of pruning onset noise.

set *B* includes the 50 segments in the training and validation sets for training the non-vocal pruning classifier so there are 70 segments in set *B*. The range of the tempo in set *B* is from 56 to 160. The system first enhanced the vocal signal by applying the vocal enhancement algorithm. Next, the potential onsets were detected by the onset detection algorithm. By using the non-vocal pruning module, some non-vocal onsets were pruned from the pool of the potential onsets. Lastly, the signal features *S* were obtained from the lyrics feature extraction module. Applying the signal features *S* (automatically computed by the system) and lyrics features (automatically computed by using the input lyrics) on the DTW algorithm, the overall alignment result was obtained.

Tables 3 and 4 show the overall alignment result of our system without and with non-vocal pruning module respectively. The “In-Range Accuracy” of the system without non-vocal pruning module was 82.36% for set *A* and 73.51% for set *B* (table 3) while that of the system with non-vocal pruning module was 85.76% for set *A* and 80.24% for set *B* (table 4). The accuracy was increased 3.4% for set *A* and 6.73% for set *B*. Thus the non-vocal pruning module was able to boost the performance of the system. The increase in the accuracy in set *B* was more significant than that in set *A* because set *B* includes the segments used in training the non-vocal pruning classifier. Moreover, we found that the system could not align

the fast songs well so the accuracy for set A was greater than that for set B (e.g. song 2 in the album “Being” in table 1) because the time distance between two consecutive sentences were very short which violated the assumption of our time distance feature, and the relative pitch matching criteria (mentioned in section 1) may be lost for fast-pace songs [5]. But in general, most of the songs are not that fast (160bpm).

	In-Range Accuracy A^R (%)		Duration Accuracy A^D (%)	
	Set A	Set B	Set A	Set B
mean	82.36	73.51	60.93	51.83
min	49.81	33.15	33.64	17.86
max	99.85	100.00	77.44	81.23
standard deviation	14.29	17.28	12.79	15.68

Table 3 Alignment accuracy of the overall performance in which the system did not apply the non-vocal pruning module. Set A contains 20 segments which were not used in training the non-vocal pruning classifier. Set B contains 70 segments which include all segments in set A and 50 segments used in training the classifier.

	In-Range Accuracy A^R (%)		Duration Accuracy A^D (%)	
	Set A	Set B	Set A	Set B
mean	85.76	80.24	64.63	58.26
min	63.01	40.51	39.00	22.70
max	99.42	100.00	81.51	81.68
standard deviation	11.70	14.78	11.46	14.57

Table 4 Alignment accuracy of the overall performance. Set A contains 20 segments which were not used in training the non-vocal pruning classifier. Set B contains 70 segments which include all segments in set A and 50 segments used in training the classifier.

9 Conclusion

We built a system to align the lyrics of Cantonese popular music. Firstly, our proposed system enhances the vocal part in commercial CD recordings to estimate the pure vocal signals by our proposed vocal signal enhancement algorithm. Then the start times/onsets of the characters sung are detected by the onset detection method. Since many non-vocal onsets are detected, they are pruned by the singing voice detection classifier which classifies an onset whether it is vocal or not. After that, the proposed features are extracted from the lyrics and the audio signals. Lastly, the start time and the end time of each lyrics sentence are obtained by the dynamic time warping algorithm.

Analysis of the modules was performed in depth to find the bottlenecks of the system. Onset detection was

very sensitive that it was able to detect most vocal onsets. However, it also detected many non-vocal onsets (false alarms), thus non-vocal pruning module was introduced to tackle this problem. The performance of the singing voice detector in the non-vocal pruning module was satisfactory (about 80% accuracy). The result showed that the non-vocal pruning module was effective to boost the performance of the system and the proposed features (time distance and relative pitch feature) were effective to align the lyrics. To evaluate the DTW algorithm and find the critical issue affecting the overall system performance, simulations of different kinds of noises including missing vocal onsets, spurious vocal onsets and incorrect pitches were carried out. The error of missing vocal onsets was critical (10% miss deducted 6.5% performance), the effect of the spurious vocal onsets was smaller (10% false alarms deducted 1% performance) and the incorrect pitches had smallest effect (nearly no performance deduction for 50% of the incorrect pitches), thus we concluded that the DTW algorithm was robust to tackle the lyrics alignment problem. Lastly, the overall performance of the system was also measured. The system was able to align the lyrics with about 80% In-Range Accuracy. This accuracy measures the ratio between the duration of lyrics sentence displayed during the lyrics sentence being sung and the actual duration of the sentence. The demonstrations of our system can be available at <http://www.cse.cuhk.edu.hk/~khwong/demo/lyricsalign/demo.htm>.

The proposed system can be probably applied to the lyrics alignment problem of other tone languages such as Mandarin if the lyricist has to write lyrics to match the relative pitches of melodies. But the proposed system is limited to one singing voice, thus the system cannot work with duets. Also, the input lyrics are assumed to be segmented line by line thus the system cannot work with unsegmented lyrics. Some Chinese characters can be pronounced in Cantonese with different tones which depend on their meanings, but the proposed system does not encounter this type of characters. If the lyrics of a song are written in English, the system cannot determine the relative pitches of these lyrics.

There are several directions for improving the system. Our system was limited to signal segments (about 20 seconds) of which the duration is as long as a section defined in [29]. The section detection method in [29] could be used to apply to the song and the lyrics. So, the song and the lyrics are divided into different sections. Since the duration of a section is as long as a segment, the proposed system can be used to align the lyrics to each section and the lyrics of the complete song can be aligned. Thus, the segment limitation can be overcome.

Another improvement is that it is possible to apply the beat detector so that the time distance features can be adapted to the speed of the song. And also, singing voice detection can be improved by using more high-level information such as the key of the song and the beat of

the song. Those high-level information can be obtained automatically by some existing beat detection systems and key detection systems. Moreover, the alignment result and the result of the singing voice detection can be combined to estimate the end time of each sentence to increase the Duration Accuracy.

Our system aligned the lyrics sentence by sentence accurately. It is extensible to align the lyrics character by character rather than sentence by sentence with a semi-automatic method described as follows. First, our system aligns the lyrics sentence by sentence. Then the user adjusts the boundaries of each sentence manually. Lastly, the DTW algorithm can be applied to each sentence with our proposed lyrics features to align each character. Another extension is the real-time/online lyrics alignment which adapts to the singer's performance. It is essential for the application of surtitles in Cantonese operas. All modules in our system can be performed in online mode except the DTW module. An online DTW algorithm such as [10] should replace the offline DTW algorithm used in this paper in order to align lyrics in real-time.

Acknowledgements We are thankful to the reviewers for their valuable comments.

References

1. W. H. Abdulla, D. Chow, and G. Sin. Cross-words reference template for dtw based speech recognition systems. In *IEEE TENCON*, pages 1576–1579, October 2003.
2. Steve Lawrence Adam Berenzweig, Daniel P. W. Ellis. Using voice segments to improve artist classification of music. In *AES 22nd International Conference*, 2002.
3. Adam L. Berenzweig and Daniel P.W. Ellis. Locating singing voice segments within music signals. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, volume W2001, pages 1–4, October 2001.
4. Steven F. Boll. Suppression of acoustic noise in speech using spectral subtraction. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-27(2):113–120, April 1979.
5. Marjorie K. M. Chan. Tone and melody in cantonese. In *Berkeley Linguistic Society, Proceedings of the Thirteenth Annual Meeting*, pages 26–37, 1987.
6. Wu Chou and Liang Gu. Robust singing detection in speech/music discriminator design. In *Proc. ICASSP*, pages 865–868, May 2001.
7. L. P. Clarisse, J. P. Martens, M. Lesaffre, B. De Baets, H. DeMeyer, and M. Leman. An auditory model based transcriber of singing sequences. In *Proc. ISMIR*, pages 116–123, October 2002.
8. David Crystal. *The Cambridge Encyclopedia of Language*. Cambridge University Press, 2nd edition, 1997.
9. A. de Cheveigné and H. Kawahara. Yin, a fundamental frequency estimator for speech and music. *Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002.
10. Simon Dixon. Live tracking of musical performances using on-line time warping. In *Proc. of the 8th Int. Conference on Digital Audio Effects (DAFx05)*, pages 92–97, September 2005.
11. Ying hao Chiang, Kwok Fan, and Tze wan Kwan. A chinese talking syllabary of the cantonese dialect (<http://humanum.arts.cuhk.edu.hk/lexis/canton2/>).
12. J. M. Hunt, M. Lenning, and P. Mermelstein. Experiments in syllable-based recognition of continuous speech. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '80*, volume 5, pages 880–883, April 1980.
13. GoldWave Incorporated. Goldwave version 5.06 (<http://www.goldwave.com>).
14. Michael Kennedy. *The Oxford Dictionary of Music*. Oxford University Press, 2nd edition, 1994.
15. Anssi Klapuri. Automatic transcription of music. Master's thesis, Tampere University of Technology, April 1998.
16. Tat-Wan Leung, Chong-Wah Ngo, and Rynson W. H. Lau. Ica-fx features for classification of singing voice and instrumental sound. In *17th International Conference on Pattern Recognition (ICPR'04)*, volume 2, pages 367–370, August 2004.
17. Alex Loscos, Pedro Cano, and Jordi Bonada. Low-delay singing voice alignment to text. In *Proceedings of International Computer Music Conference 1999*, 1999.
18. Lie Lu, Hao Jiang, and HongJiang Zhang. A robust audio classification and segmentation method. In *Proc. of the 9th ACM International Conference on Multimedia*, pages 203–211, 2001.
19. Namunu Chinthaka Maddage, Kongwah Wan, Changsheng Xu, and Ye Wang. Singing voice detection using twice-iterated composite fourier transform. In *2004 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1347–1350, 2004.
20. Tin Lay Nwe, Arun Shenoy, and Ye Wang. Singing voice detection in popular music. In *Proceedings of the 12th annual ACM international conference on Multimedia*, pages 324–327, 2004.
21. Linguistic Society of Hong Kong. Cantonese romanization (<http://cpct92.cityu.edu.hk/lshk/>).
22. Bobby Owsinski. *Mixing Engineer's Handbook*. Thomson Course Technology, November 1999.
23. J. Pinquier, J. Rouas, and R. Andr e Obrecht. Robust speech/music classification in audio documents. In *International Conference on Spoken Language Processing*, volume 3, pages 2005–2008, September 2002.
24. Matti P. Ryyänänen and Anssi P. Klapuri. Modelling of note events for singing transcription. In *Proc. ISCA Tutorial and Research Workshop on Statistical and Perceptual Audio Processing*, October 2004.
25. Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. In *IEEE Transactions on Acoustics, Speech, and Signal Processing*, volume ASSP-26, pages 43–49, February 1978.
26. Eric Scheirer and Malcolm Slaney. Construction and evaluation of a robust multifeature speech/music discriminator. In *Proc. ICASSP*, pages 1331–1334, April 1997.
27. Eric D. Scheirer. Tempo and beat analysis of acoustic musical signals. *Journal of Acoustic Society of America*, (1):588–601, 1998.
28. Saeed V. Vaseghi. *Advanced Signal Processing and Digital Noise Reduction*. Wiley, 2000.
29. Ye Wang, Min-Yen Kan, Tin Lay Nwe, Arun Shenoy, and Jun Yin. Lyrically: Automatic synchronization of acoustic musical signals and textual lyrics. In *MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia*, pages 212–219, October 2004.