

Unisoner: An Interactive Interface for Derivative Chorus Creation from Various Singing Voices on the Web

Keita Tsuzuki^{*1} Tomoyasu Nakano^{**2} Masataka Goto^{***3} Takeshi Yamada^{***4} Shoji Makino^{***5}

^{*} Graduate School of Systems and Information Engineering, University of Tsukuba, Japan

^{**} National Institute of Advanced Industrial Science and Technology (AIST), Japan

^{***} Faculty of Engineering, Information and Systems, University of Tsukuba, Japan

¹tsuzuki[at]mmlab.cs.tsukuba.ac.jp ^{2,3}{t.nakano, m.goto}[at]aist.go.jp

⁴takeshi[at]cs.tsukuba.ac.jp ⁵maki[at]tara.tsukuba.ac.jp

ABSTRACT

This paper describes *Unisoner*, an interface for assisting the creation of derivative choruses in which voices of different singers singing the same song are overlapped on one common accompaniment. In the past, it was time consuming to create such choruses because creators have to manually cut and paste fragments of singing voices from different singers, and then adjust the timing and volume of every fragment. Although several interfaces for “mashing up” different songs have been proposed, no mash-up interface for creating derivative choruses by mixing singing voices for the same song has been reported. *Unisoner* enables users to find appropriate singers by using acoustic features and metadata of the singing voices to be mixed, assigning icons of the found singers to each phrase within a song, and adjusting the mixing volume by moving those icons. *Unisoner* thus enables users to easily and intuitively create derivative choruses. It is implemented with several signal processing techniques, including a novel technique that integrates F_0 -estimation results from many voices singing the same song to reliably estimate F_0 without octave errors.

1. INTRODUCTION

Vocal covers, derivative of existing original songs, are common in the age of digital music production and sharing [1]. Many amateur singers sing the same song and upload their singing voices to video sharing services. Those vocal covers are called “Me Singing”, and 1.7 million “Me Singing” videos have been uploaded on a popular video sharing service *YouTube*¹, and 665,000 videos have been uploaded on a Japanese video sharing service, *Niconico*². These vocal covers make it possible for people to listen to and compare voices of different singers singing the same song. Since vocal covers are so popular, many (amateur) artists have provided karaoke versions to make it easier to create vocal covers.

Copyright: ©2014 Keita Tsuzuki et al. This is an open-access article distributed under the terms of the [Creative Commons Attribution 3.0 Unported License](http://creativecommons.org/licenses/by/3.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

¹ <http://www.youtube.com>

² <http://www.nicovideo.jp>

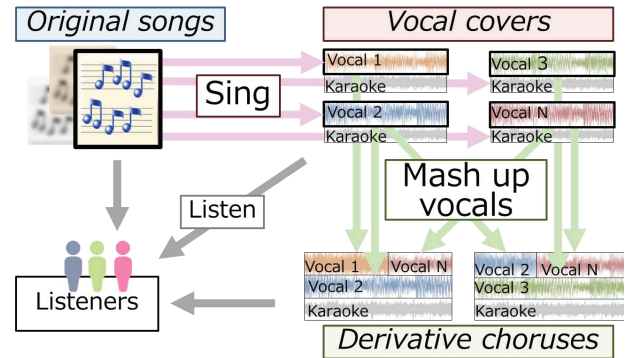


Figure 1. Relationship among original songs, vocal covers, derivative choruses, and listeners. Various singers sing the same song to create vocal covers. From these vocals, derivative choruses are created. Many listeners enjoy not only the original songs, but also the vocal covers and derivative choruses.

Some creators have started creating derivative works of such vocal covers by mixing (mashing up) them along with one common accompaniment. We call this type of music *derivative chorus*. Figure 1 shows the relationship among original songs, vocal covers, and derivative choruses. Approximately 10,000 derivative choruses have been uploaded on Niconico, and some derivative choruses have received more than 1 million views³.

Derivative choruses are similar to Eric Whitacre’s “Virtual Choir”⁴. Virtual Choir was created by mixing singing voices that were purposely recorded and uploaded for this collaborative choir. In contrast, though, derivative choruses simply reuse existing vocal covers that are not intended to be mixed with other vocals.

Listeners can enjoy derivative choruses in the following ways:

Listen to different expressions of derivative choruses

Famous original songs tend to have several derivative choruses. Even if the original song is the same, the vocal covers used and their arrangement are different in each derivative chorus. Listeners can enjoy comparing such different vocals and arrangements.

³ A derivative chorus at <http://www.nicovideo.jp/watch/sm5132988> has more than 1.9 million views.

⁴ <http://ericwhitacre.com/the-virtual-choir>

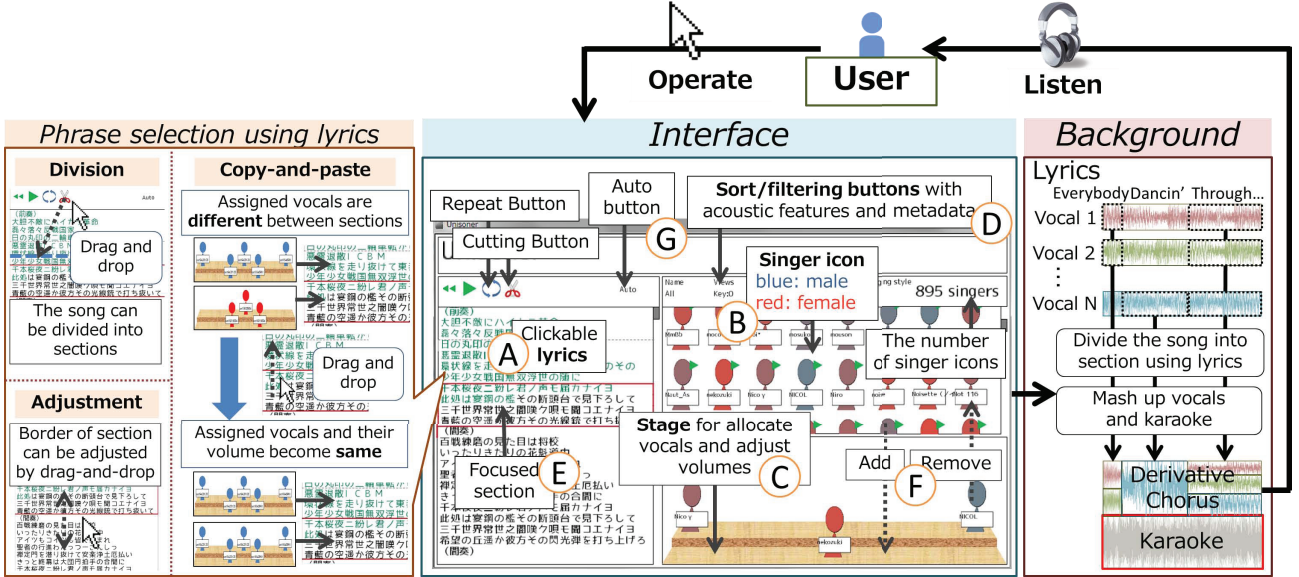


Figure 2. Overview of Unisoner and the interaction between user and Unisoner.

Compare characteristics of singers Listening to several vocal covers at the same time allows listeners to notice differences in singing style, vocal timbre, etc.

Discover favorite singers Derivative choruses give listeners a chance to discover singers they like from the vocal covers used in the choruses. Some creators of derivative choruses mash up vocal covers to highlight their favorite singers.

Creators of derivative choruses can also enjoy the derivative creation.

However, it is not easy to create derivative choruses. First, it is necessary to extract singing voices from vocal covers by suppressing their karaoke accompaniments. Second, since the extracted singing voices are not temporally aligned, it is time-consuming to synchronize them with a karaoke accompaniment. Third, creators must use a waveform-editing tool, such as Digital Audio Workstation (DAW), to manually cut and paste fragments of singing voices from different singers.

We therefore propose an easy-to-use interface, *Unisoner*, that enables end users without musical expertise to create derivative choruses. Unisoner overcomes the difficulties described above by automating the synchronization tasks necessary for derivative choruses and providing intuitive mixing functions. This work forms part of the emerging field of creative Music Information Retrieval (MIR), where MIR techniques are used for creative purposes. In this field, there have been interfaces for creating mash-up music by connecting loops corresponding to musical pieces [2], creating mash-up music automatically [3], and creating mash-up dance videos [4], yet no interface for derivative choruses has been reported.

In addition to Unisoner, we also propose a vocal training interface that leverages various vocal covers. Since amateur singers have difficulty improving their singing skill, vocal training interfaces, such as an interface to analyze a singer's voice alone [5] and an interface to compare two singing voices [6], have been proposed. Our training inter-

face allows a user to compare his or her voice with a wide variety of vocal covers by visualizing them. The user can choose a favorite vocal cover using metadata such as the number of views on a video sharing service, and compare the fundamental frequency (F_0) of the chosen vocal with the F_0 of the user's voice so that the user can sing more like the favorite vocal cover.

Demonstration videos of Unisoner are available at <http://mmlab.cs.tsukuba.ac.jp/%7etsuzuki/icmcsmc14>.

2. UNISONER: INTERACTIVE DERIVATIVE CHORUS CREATION INTERFACE

Unisoner enables users to create derivative choruses easily, and allows for the simultaneous listening of various vocal covers and derivative choruses. We assume audio signals have been extracted from a set of desired videos on YouTube or NicoNico. We call the resulting audio signal the *accompanied vocal*, and the vocal audio signal after vocal extraction (see Section 3) the *suppressed vocal*.

2.1 Interface of Unisoner

Figure 2 shows an overview of Unisoner. Unisoner displays each vocal cover as an icon that represents each singer (the singer icon). The user can assign each vocal cover to phrases and adjust the volume simply by dragging and dropping singer icons. Moreover, music-synchronized lyrics, given in advance, enable the user to assign each vocal cover to certain phrases easily.

The smooth assignment of each vocal cover to phrases is important for efficient creation of derivative choruses. Unisoner dynamically synthesizes the chorus according to the user's operations, allowing the user to check the output chorus instantly without stopping the music. The creation of derivative choruses in real-time with Unisoner can be regarded as an example of active music listening [7]. Unisoner and standard tools are compared in Figure 3.

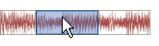
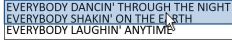
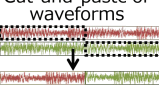
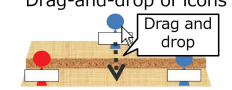


	Waveform-editing tools	Unisoner
Selection of phrases	Waveform based 	Lyrics based 
Assignment of vocals	Cut-and-paste of waveforms 	Drag-and-drop of icons 
Volume control of vocals	Precise control using sliders or knobs 	Simple control using icons 
Filtering and sorting of vocals	Singers' name File name	Acoustic features Metadata

Figure 3. Comparison of waveform-editing tools and Unisoner.

2.2 Three functions of Unisoner

Unisoner features the following three main functions.

1) Lyrics-based phrase selection Users must be able to intuitively select desired phrases to efficiently create derivative choruses. Phrase selection on conventional waveform-editing tools is inefficient because it is difficult to select correct phrases just by looking at waveforms. However, it is time-consuming for users to listen to each fragment of singing voice. Unisoner can select and jump to the phrase of interest by leveraging the song lyrics (marked Ⓐ in Figure 2).

Users can divide a song into sections that include multiple phrases and assign vocal covers to sections. Operations related to song lyrics and sections are illustrated in left figure of Figure 2. In those operations, the copy-and-paste function of volume and panning information along with drag-and-drop is a unique feature of Unisoner. This is clearly useful when a user wants to use the same vocal cover at the same volume on a different section, such as when equalizing the assigned vocal cover on the first and second verse.

As a way to use clickable lyrics, skipping the playback position [8] and selecting the position for recording [9] has been proposed. However, selecting sections and enabling the user to edit derivative choruses based on lyrics is a novel idea regarding the usage of clickable lyrics.

2) Assignment and volume control of vocal covers using icons Waveform-editing tools enable music creators to adjust the volume of the left and right channels of each suppressed vocal in detail, but this becomes increasingly cumbersome as the number of vocal tracks increases. Unisoner represents each suppressed vocal as an icon (Ⓑ in Figure 2), colored according to the estimated gender-likeness of the singer (as explained in Section 3.5), so the user can intuitively understand how each suppressed vocal is sung and how high the volume of each suppressed vocal is. The overall volume of each suppressed vocal can be adjusted by moving the singer icon to the front or back, and the volume balance between two channels can be adjusted

by moving the singer icon left or right on the stage (Ⓒ in Figure 2). The balance between the left and right channels is decided automatically so that the total energy is evenly divided between the two channels. These forms of volume control and the assignment of vocal cover to phrase using icons can be done in real-time without stopping the music. The real-time assignment of vocal covers assists a user’s trial-and-error approach to creation.

3) Sorting and filtering using metadata and acoustic features The sorting and filtering of vocal covers allow a user to explore thousands of vocals. Unisoner can sort and filter vocal covers by acoustic features and metadata.

Specifically, the sorting criteria we used are the similarity of singing style, calculated from F_0 and ΔF_0 ; and voice timbre, calculated from the Mel Frequency Cepstral Coefficient (MFCC). Singing style is related to several features, such as dynamics of F_0 contour, voice power, and voice timbre. We used F_0 and ΔF_0 to explain the dynamics F_0 in this paper. ΔF_0 is an important factor for discriminating singing voices from speech voices [10], and has been used for singer identification [11]. Since the singers essentially sing the same melody for each song, the straight F_0 value is useful for expressing the dynamics of F_0 . Criteria obtained from metadata are the singer’s name⁵, the number of views, and the number of Mylists (“favorites” annotated by users).

Filtering criteria are the gender-likeness of vocal cover and the key difference from the original song, which are both obtained from acoustic features. Acoustic features used in sorting and filtering are explained in Section 3. These various forms of sorting and filtering can be done by clicking a button on Unisoner (Ⓓ in Figure 2).

2.3 Typical use of Unisoner

By clicking the lyric (Ⓐ in Figure 2), a user can change the current playback position to focus on a certain section. A vocal cover will be added to the specified section of choice by dragging-and-dropping the corresponding singer icon (Ⓑ in Figure 2). The volume of each vocal cover can be adjusted by moving the singer icon. For creation of derivative choruses with specific features, such as a derivative chorus with only male singers, the filtering and sorting functions are essential. The “Auto button” (Ⓒ in Figure 2) can be employed when the user lacks a creative spark. Unisoner automatically divides the song into sections and randomly assigns vocal covers into each section when the Auto button is clicked.

2.4 Application of a derivative chorus for vocal training

Unisoner can also be used for vocal training; since most vocal covers use the same musical structure, singers can learn how to sing from existing derivative works. A proposed vocal training interface (Figure 4) utilizes various vocal covers of the same song as a reference to help users recognize their singing characteristics, which is important

⁵ Uploaders’ names are currently used for substitute of singer’s name.

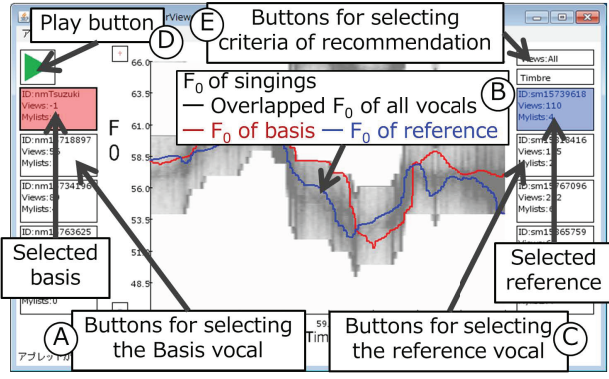


Figure 4. Screenshot of the proposed vocal training interface.

for improving singing skill. To achieve this, visualization of the F_0 of singing voices is effective. Conventional vocal training interfaces [5, 6] also visualize the F_0 of a singing voice for training. Our proposed interface visualizes F_0 of the user’s vocal, F_0 of various vocal covers, and the overlapped F_0 of thousands of vocal covers at the same time.

Because many vocal covers have been uploaded on the Web, recommendation of an appropriate vocal cover is necessary to find a good set of references. Our proposed vocal training interface recommends vocal cover based on the similarity of singing style and vocal timbre, and shows the number of views and the number of Mylists for the referred vocal. Users can listen to and compare both their own and similar singing voices in addition to illustrative metadata.

2.4.1 Operation of proposed vocal training interface

The vocal training interface can be used by the following steps. This interface helps users recognize their particular singing specialty by comparing various vocal covers which are similar.

Selection of basis vocal A user selects the vocal cover that will be used as a search reference. The user can click buttons on the left side of the display (A in Figure 4) or can drag and drop a file with data of their voice. The number of views and Mylists are displayed on buttons. F_0 of the selected vocal cover (the “basis vocal”) is indicated by the red line in the center (B in Figure 4). In addition, overlapped F_0 lines of all singing examples are shown in black lines. The more vocal covers sung with that F_0 in that time frame, the darker the overall contour becomes.

Selection of reference vocal Candidate vocal covers which are close to the basis vocal with respect to certain criteria are displayed on the right-side buttons (C in Figure 4). Users can select a reference by clicking these buttons, after which the F_0 of the reference is indicated by the blue line (B in Figure 4). When the play button (D in Figure 4) is clicked, the basis vocal is played from the left channel, with the reference vocal played from the right channel.

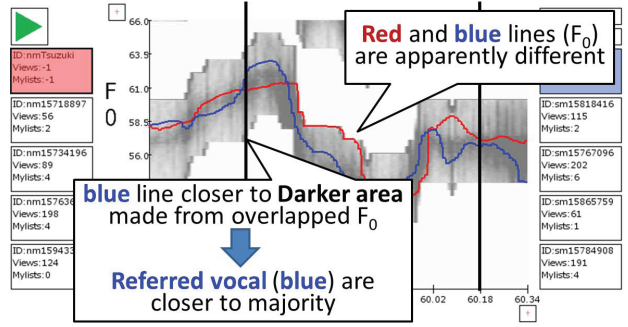


Figure 5. Comparison of F_0 of basis vocal, that of reference vocal, and overall F_0 of all singers. F_0 of basis vocal differs from that of both reference vocal and the majority of all singers.

Selection of criteria for recommendation The criteria for recommendation can be changed by clicking the upper right buttons (E in Figure 4). Users can select the recommendation criteria from the similarity of voice timbre, the similarity of singing style, and the overall similarity, which includes all of these (methods to calculate these similarities are described in Section 3.4). Moreover, users can filter the recommendation results by the number of views, enabling comparison between a basis vocal and references which are both close to the basis vocal and popular.

2.4.2 Recognizing characteristics of a user’s singing voice using the proposed interface

By visualizing a particular F_0 contour, a user can easily see the differences between vocal covers and can get an idea of how to sing by listening. In a situation such as that of Figure 5, the red line (the user’s singing voice) and blue line (reference vocal) are clearly different. Comparing these two lines and the black lines in the background, it is apparent that the blue line is closer to the area where the black lines are concentrated (the dark black area). Since black lines indicate the trend in F_0 , it can be said that this user sings differently from that of the majority of singers. This enables understanding of deviations from the norm in this singing example. After this analysis, the user can listen to the reference and practice singing to adjust the pitch.

3. IMPLEMENTATION OF UNISONER

We implemented Unisoner by developing a new F_0 estimation method that utilizes various vocal covers of the same song. Unisoner is also based on various methods, such as karaoke accompaniment suppression, similarity calculation between suppressed vocals, and gender-likeness estimation of suppressed vocals.

In Unisoner, all accompanied and suppressed vocals are sampled at 16 kHz, and they are monophonic. We assume that the karaoke accompaniments used in vocal cover are given, because these are open to the public in many cases⁶.

⁶ Hamasaki *et al.* reported that many VOCALOID song writers publish karaoke versions of their songs [1], for example.

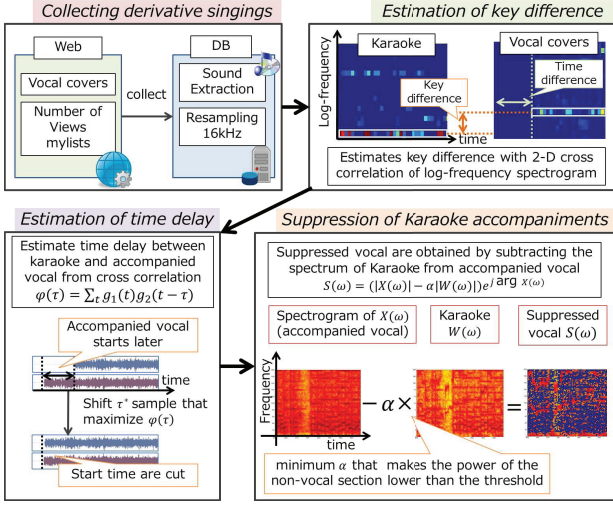


Figure 6. The preprocessing flow. Vocal covers are collected from the Web and resampled into 16 kHz signals. The key difference and time delay from the original song are then estimated. Last, accompaniments included in the vocal covers are suppressed.

3.1 Songs and data used in Unisoner

We chose an original song that has the largest number of vocal covers in Nico Nico⁷. We collected videos of those vocal covers as well as the karaoke version of the original song. 4,941 vocal cover videos were collected from Nico Nico in total, and the numbers of views and Mylists attached to each video were also collected. However, the collected singing videos included some videos which were inappropriate for analysis, such as remixed songs of the original song. We filtered out singing videos that were more than 15 seconds shorter or longer than the original song to avoid this problem. As a result, 4488 vocal cover videos were used for Unisoner and the signal processing described below.

3.2 Preprocessing

The estimation of key and time differences from the original song and the suppression of karaoke accompaniments make up the preprocessing steps in Unisoner. These steps are illustrated in Figure 6. For computational efficiency, the key difference is first estimated with broad time resolution, after which the time delay is estimated at a finer resolution. The first estimation is done with a hop time of 100 ms, and the second estimation is done with a hop time of 62.5 μ s or 1 sample.

Key difference and rough time delay estimation The key difference from the original song is estimated by calculating the two-dimensional cross correlation of a log-frequency spectrogram of accompanied vocal and karaoke accompaniments. This calculation has to be done in two dimensions because the differences of both key and time must be calculated simultaneously. Log-frequency spectrograms, calculated by getting the inner product with

a windowed frame, are used because the differences in keys will be described as linear differences by using log-frequency. We use a Hanning window of 2,048 samples and a hop size of 1,600 samples. The log-frequency spectrogram is calculated in the range of 1 to 128 in MIDI note number (8.7 to 13289.7 Hz) and 1 bin is allocated for each note number. The MIDI note number f_M can be calculated by the following equation when f_{Hz} is the frequency in Hz:

$$f_M = 12 \log_2 \left(\frac{f_{Hz}}{440} \right) + 69. \quad (1)$$

The estimation result is limited to between ± 6 MIDI notes of the original song. Note that many singers sing the exact same melody as the original, and that our method is invariant to octave choice by the singer. Compared to the hand-labeled key difference, 96 out of 100 singing samples were estimated correctly. There were 50 singing samples with a key difference and the other samples were in the same key as the original song. Pitch-shifted karaoke accompaniments are used for the following preprocessing on accompanied vocal with a different key. Audacity⁸ is used to make pitch-shifted audio.

Estimation of precise time delay The time delay between accompanied vocal $g_1(t)$ and karaoke accompaniment $g_2(t)$ is estimated in samples using the cross correlation function $\phi(\tau)$

$$\phi(\tau) = \sum_t g_1(t)g_2(t - \tau), \quad (2)$$

where t and τ represent samples. By shifting each accompanied vocal by τ^* samples, which maximizes $\phi(\tau)$ as follows

$$\tau^* = \underset{\tau}{\operatorname{argmax}} \phi(\tau), \quad (3)$$

the start time of all accompanied vocals can be cut to match the karaoke. τ^* is limited to a range of ± 50 ms of the estimated delay calculated in the previous step.

The median difference between hand-labeled samples and the estimated time delay in the 100 singing samples, where the same samples are used as for the evaluation of key difference, was 34.6 ms.

Suppression of karaoke accompaniments Karaoke accompaniments in accompanied vocal are suppressed by spectral subtraction [12]:

$$H(\omega) = |X(\omega)| - \alpha|W(\omega)|, \quad (4)$$

$$S(\omega) = \begin{cases} 0 & (H(\omega) \leq 0) \\ H(\omega)e^{j \arg X(\omega)} & (\text{otherwise}), \end{cases} \quad (5)$$

where $X(\omega)$ and $W(\omega)$ are spectra of the accompanied vocal and karaoke accompaniments. α is a parameter, describing the weight for subtracting the karaoke, and j is the imaginary unit.

The quality of suppressed vocal is sensitive to the choice of α . Thus, an appropriate α for each accompanied vocal must be estimated before suppression. To determine α , the karaoke accompaniment is temporarily suppressed

⁷ A song at <http://www.nicovideo.jp/watch/sm15630734> is chosen.

⁸ <http://audacity.sourceforge.net>

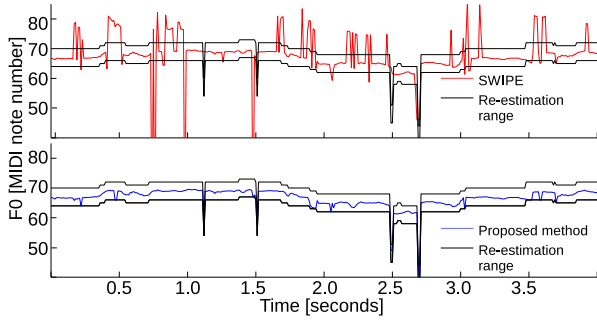


Figure 7. Upper: F_0 of suppressed vocal from estimation in SWIPE (red line). Lower: Estimated F_0 of suppressed vocal using the proposed method (blue line). Black lines in both figures show the range of results determined from the trend of F_0 illustrated in Figure 8 and used in the proposed method. Variance in the estimated result was reduced by properly determining the estimation range.

with $\alpha = 1$, and non-vocal sections are estimated from the result of suppression. These sections are classified as an area where power is lower than the pre-defined threshold (average power of the full mix). Power is calculated with a Hanning window of 1,024 samples, FFT with 2,048 samples, and a hop size of 512 samples.

After estimation of non-vocal sections, karaoke accompaniment is suppressed on the longest non-vocal section for each α , which increases by 0.1 from 0 to 5. Minimum α , where the power of singing voice after suppression is lower than the threshold, are treated as an appropriate α for accompaniment suppression of the whole song. 1% of the power of voice before suppression is currently used as a threshold. Note that $|S(\omega)|$ tends to be smaller in the non-vocal section than in the vocal section, no matter what α is, because accompanied vocal and karaoke accompaniments have almost the same signal in a non-vocal section. To determine α and suppress karaoke accompaniment, a Hanning window of 2048 samples, FFT calculated with 4096 samples, and a hop size of 1024 samples are used.

3.3 F_0 estimation method integrating various vocal covers

An effective F_0 estimation method for suppressed vocal is needed to enable sorting according to a singers' singing style. We used the SWIPE algorithm [13] as a base method for F_0 estimation. It is calculated with a time resolution of 10 ms and frequency resolution of 0.1 MIDI notes.

Though SWIPE is a highly precise method, estimation errors sometimes occur (such as in the upper plot of Figure 7) when it is applied to suppressed vocal. In this research, we propose an F_0 estimation method that leverages various vocal covers. We assume that even if each estimation result includes some errors, the trends appearing in results will be close to the true F_0 value. If this is true, the precision of the F_0 estimation should be improved by searching for the most feasible value around the trend. This method for evaluating the range of estimation is an important method for improving efficiency, and could also be applied to other F_0 estimation methods.

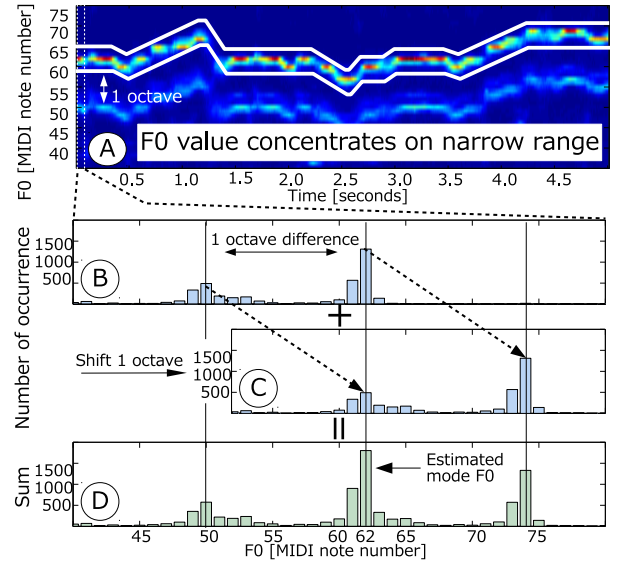


Figure 8. A: Distribution of F_0 values 0 to 5 seconds after prelude. The estimation results from 4488 suppressed vocals were used for this figure. A sharp peak surrounded by a white line can be seen in each time frame. B: Histogram of the appearance of the F_0 value in the first 100 ms of Figure A. Two peaks separated by the distance of a 12 note number can be seen. C: Histogram of Figure B shifted by 12 notes (1 octave). D: Sum of Figures B and C. Mode F_0 is an F_0 value with maximum sum.

Range estimation of F_0 In Figure 8, A shows the distribution of F_0 values for each suppressed vocal. The majority of estimation results are concentrated within a narrow range of values (indicated by the white lines). Two peaks are shown in the figure, and these peaks can be considered as the F_0 of suppressed vocals sung like the original song (\pm octave errors). This result suggests that reliable F_0 estimation can be done for each suppressed vocal by integrating various F_0 estimation results.

In Figure 8, B shows a histogram of the first 0.1 seconds of A. Peaks at MIDI note numbers 50 and 62, which have a 1 octave difference, can be observed. Assuming that the true F_0 value of each suppressed vocal is near the trend (peak), we regard the most frequently appearing F_0 , considering the 1 octave difference, as the *mode F_0* of that frame. The mode F_0 can be calculated by adding the number of occurrences of an F_0 value and the occurrence of F_0 values that are 1 octave (12 notes) lower than that F_0 from the lowest to the highest (sum of B and C in Figure 8), and then selecting the F_0 value that has the maximum sum (62 in D of Figure 8).

Re-estimation of F_0 F_0 for each frame is re-estimated by limiting the estimation range around the mode F_0 after calculating the mode F_0 in every frame. However, it is possible that vocal covers may be sung 1 octave higher or lower than the original (for example, when a male singer sings a song originally recorded by a female singer). To counteract this, the distance between the F_0 value of the first estimation and mode F_0 , mode $F_0 + 1$ octave, and

mode $F_0 - 1$ octave are calculated. This distance, D , between the estimated F_0 and mode F_0 is then calculated by

$$D = \sum_t \sqrt{(f_0(t) - f_{\text{mode}}(t))^2}, \quad (6)$$

where t is an index for the time frame, $f_0(t)$ indicates the estimated F_0 , and $f_{\text{mode}}(t)$ indicates mode F_0 .

In re-estimation, ± 3 semitones from the selected candidates was used as the estimation range. Properties such as time and frequency resolution were same with those for the initial estimation. The lower plot in Figure 7 shows the re-estimated F_0 , with the black lines showing the estimation range. Comparing the estimations, we can see that the variance of the estimated result has decreased from that of the initial estimation.

3.4 Similarity calculation method between vocal covers

To sort acoustic features (Section 2.2), we calculate the similarity between the suppressed vocals by using the Earth Movers Distance (EMD) [14] between their Gaussian Mixture Models (GMMs).

Voice timbre similarity MFCCs were used as a feature.

The frame length was 25 ms and the hop time was 10 ms for the calculation. The lower 12 dimensions, excluding the DC components, were used.

Singing style similarity F_0 and ΔF_0 were used.

Overall similarity MFCC, F_0 , and ΔF_0 were used.

3.5 Gender-likeliness estimation method for vocal cover

Each singer's gender-likeliness is estimated from the estimated probability of a two-class (male- and female-class) Support Vector Machine (SVM) [15]. Unisoner sets the color of the singer icon according to the estimated probability of each class. 12-dimensional MFCCs are calculated using a 25-ms frame length and a 10-ms hop time. MFCCs and ΔF_0 from 30 suppressed vocals of another song (15 male, 15 female) are used for training. The male- and female-likeliness are calculated by taking the median of the estimated probability over all frames in each class. The duration between the beginning of the first verse and the end of the first chorus is regarded as a vocal section and used for both training and estimation.

This method is based on the technique used in Songrium [1]. Songrium uses ΔF_0 and LPMCC (mel-cepstral coefficients of LPC spectrum) from *reliable frames* [11] as a feature for SVM. Unisoner, however, uses MFCC as a feature, since MFCC is a common feature for gender estimation [16] and its usefulness in gender recognition tasks of speech has been verified [17].

3.6 Chorus synthesis

Unisoner dynamically synthesizes a derivative chorus according to the assignments of vocals to sections. The location of a singer icon in the interface determines the volume.

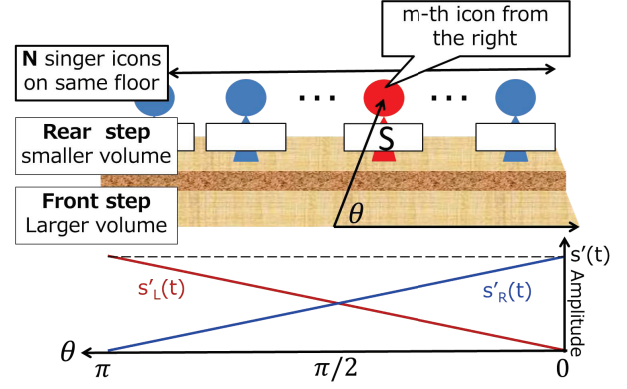


Figure 9. Upper: Parameters to decide volume. Lower: Amplitude variation of each suppressed vocal in left speaker ($s'_L(t)$) and right speaker ($s'_R(t)$) corresponding to θ .

Determination of overall volume Overall volume is first calculated. The bottom-right area of the Unisoner display resembles a stage with two-step stairs and a user can place each singer icon on the stage (Figure 9) to assign each corresponding suppressed vocal to a section and adjust the volume of the suppressed vocal. Let the waveforms of suppressed vocal S be $s(t)$. The adjusted waveforms $s'(t)$ are then calculated as

$$s'(t) = \begin{cases} s(t) & (S \text{ locates on front step}) \\ \frac{1}{2}s(t) & (S \text{ locates on rear step}). \end{cases} \quad (7)$$

Determination of angle for panning The angle for panning each suppressed vocal is then determined. When N singer icons are located on the same floor and a singer icon of suppressed vocal S is the m -th singer icon from the right (from the user's view), the localization angle θ of S is determined by

$$\theta = \begin{cases} \frac{m}{N+1}\pi & (N \neq 1 \text{ and } S \text{ is on front step}) \\ \frac{m-1}{N-1}\pi & (N \neq 1 \text{ and } S \text{ is on rear step}) \\ \pi/2 & (N = 1), \end{cases} \quad (8)$$

where θ takes the range $[0, \pi]$, as shown in Figure 9. This equation was designed to locate suppressed vocals on the front step near the center, and to make the number of suppressed vocals equal on the left and right sides.

Determination of final volume Last, the waveforms of left and right channels, $s'_L(t)$ and $s'_R(t)$, are determined from $s'(t)$ and θ as follows:

$$s'_L(t) = \frac{\theta}{\pi}s'(t), \quad s'_R(t) = (1 - \frac{\theta}{\pi})s'(t). \quad (9)$$

3.7 Other data needed for implementation

A map between the lyrics and the waveform of suppressed vocal, used for lyrics-based phrase selection (section 2.2), and timing for dividing the song, used for automatic synthesis of the chorus (section 2.3), are needed to implement Unisoner. These data are currently prepared by the user, although the application of existing techniques such as lyric alignment [8] or chorus detection [18] could make these user tasks unnecessary in the future.

4. DISCUSSION

Unisoner was designed for users unfamiliar with the creation of music or software for creating music (such as waveform-editing tools). Derivative choruses are well suited for users who are learning to create music using our interface because each vocal cover is itself a complete music piece. Users can create music easily only by selecting vocals. Unisoner can also be considered an Augmented Music-Understanding Interface [19], since one function of derivative choruses is to support the analysis of singer characteristics. Trial usage of the Unisoner (chorus creation interface and vocal training interface) suggests that it would be a useful tool for amateur users.

Vocal covers can be regarded as a kind of database of cover songs. There are several databases for cover songs, such as the SecondHandSongs dataset⁹, which are linked to the Million Song Dataset [20]. An advantage of vocal covers compared to the usual cover songs is that most vocal covers of a song are sung in the same tempo and the same musical structure as the original song. Thus, they are useful for examining how people listen to songs or what makes songs more appealing. Signal processing techniques for vocal covers, such as those introduced in this paper, may have a potential as a basis of such examination.

5. CONCLUSIONS

In this paper we proposed Unisoner, which enables a user to easily and intuitively create derivative choruses by simply dragging and dropping icons. Another key feature of Unisoner is phrase selection using lyrics. Unisoner should improve the efficiency of creating derivative choruses compared with using conventional waveform-editing tools. To realize Unisoner, several signal processing methods have been implemented. Among these methods is a new F_0 estimation method that improves precision by considering the trend of each vocal cover's F_0 . Even though each F_0 contains some errors, our method is able to overcome those errors.

In our future work, we will continue to improve the precision of each signal processing method and interface for utilizing vocal covers. For example, we will consider the use of features other than ΔF_0 or MFCCs for estimating the similarity between vocal covers.

Acknowledgments

We thank Masahiro Hamasaki, and Keisuke Ishida for handling the videos from Niconico; and Matt McVicar, and Graham Percival for helpful comments and proofreading. This work was supported in part by OngaCrest, JST.

6. REFERENCES

- [1] M. Hamasaki, M. Goto, and T. Nakano, "Songrium: A music browsing assistance service with interactive visualization and exploration of a Web of Music," in *Proc. WWW 2014*, 2014.
- [2] N. Tokui, "Massh!—a web-based collective music mashup system," in *Proc. DIMEA 2008*, 2008, pp. 526–527.
- [3] M. Davies, P. Hamel, K. Yoshii, and M. Goto, "AutoMashUpper: An automatic multi-song mashup system," in *Proc. ISMIR 2013*, 2013, pp. 575–580.
- [4] T. Nakano, S. Murofushi, M. Goto, and S. Morishima, "DanceReProducer: An automatic mashup music video generation system by reusing dance video clips on the web," in *Proc. SMC 2011*, 2011, pp. 183–189.
- [5] D. Hoppe, M. Sadakata, and P. Desain, "Development of real-time visual feedback assistance in singing training: a review," *J. Computer Assisted Learning*, vol. 22, pp. 308 – 316, 2006.
- [6] T. Nakano, M. Goto, and Y. Hiraga, "Mirusinger: A singing skill visualization interface using real-time feedback and music CD recordings as referential data," in *Proc. ISMW 2007*, 2007, pp. 75–76.
- [7] M. Goto, "Active music listening interfaces based on signal processing," in *Proc. ICASSP 2007*, 2007, pp. IV–1441–1444.
- [8] H. Fujihara, M. Goto, J. Ogata, and H. G. Okuno, "LyricSynchronizer: Automatic synchronization system between musical audio signals and lyrics," *IEEE J. Selected Topics in Signal Processing*, vol. 5, no. 6, pp. 1251–1261, 2011.
- [9] T. Nakano and M. Goto, "VocaRefiner: An interactive singing recording system with integration of multiple singing recordings," in *Proc. SMC 2013*, 2013, pp. 115–122.
- [10] Y. Ohishi, M. Goto, K. Itou, and K. Takeda, "Discrimination between singing and speaking voices," in *INTERSPEECH*, 2005, pp. 1141–1144.
- [11] H. Fujihara, M. Goto, T. Kiatahara, and H. G. Okuno, "A modeling of singing voice robust to accompaniment sounds and its application to singer identification and vocal-timbre-similarity-based music information retrieval," *IEEE Trans. ASLP*, vol. 18, no. 3, pp. 638 – 648, 2010.
- [12] S. F. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Trans. ASSP*, vol. 27, no. 2, pp. 113–120, 1979.
- [13] A. Camacho, "SWIPE: A sawtooth waveform inspired pitch estimator for speech and music," Ph.D. dissertation, University of Florida, 2007.
- [14] Y. Rubner, C. Tomasi, and L. J. Guibas, "The earth mover's distance as a metric for image retrieval," *International J. Computer Vision*, vol. 40, no. 2, pp. 99–121, 2000.
- [15] C. Chih-Chung and L. Chih-Jen, "LIBSVM: A library for support vector machines," *ACM Trans. Intelligent Systems and Technology*, vol. 2, no. 3, pp. 1 – 27, 2011.
- [16] B. Schuller, C. Kozielski, F. Weninger, F. Eyben, G. Rigoll *et al.*, "Vocalist gender recognition in recorded popular music," in *Proc. ISMIR 2010*, 2010, pp. 613–618.
- [17] T. Vogt and E. André, "Improving automatic emotion recognition from speech via gender differentiation," in *Proc. LREC 2006*, 2006.
- [18] M. Goto, "A chorus-section detection method for musical audio signals and its application to a music listening station," *IEEE Trans. ASLP*, vol. 14, no. 5, pp. 1783 – 1794, 2006.
- [19] —, "Augmented music-understanding interfaces," in *Proc. SMC 2009 (Inspirational Session)*, 2009.
- [20] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proc. ISMIR 2011*, 2011, pp. 591–596.

⁹ <http://labrosa.ee.columbia.edu/millionsong/secondhand>