SPEECH TO SINGING SYNTHESIS SYSTEM

Mingqing Yun, Yoon mo Yang, Yufei Zhang

Department of Electrical and Computer Engineering
University of Rochester

ABSTRACT

This paper describes a speech-to-singing synthesis system that can synthesize a singing voice from an input speaking voice. The system controls two acoustic features that determine the difference between speaking and singing voices: the fundamental frequency (F0) and the phoneme duration. By changing the pitch of the speaking voice into the pitch of the singing voice, we can successfully synthesis the singing melody. Then after modifying the phoneme duration, the speech can be synthesis to a singing voice. The system finally generates a singing voice that preserves the timbre of the speech voice but has the singing-voice features. Experimental results show that this system can convert speech voices into singing voices whose timbre is almost the same as the original singing voices.

Index Terms— pitch detection, speaking to singing synthesis, phase vocoder

1. INTRODUCTION

The synthesis of the singing voice is the artificial production of human-like singing voice. In our daily lives, people sing songs to express their emotions, whether they are in happy or sad mood. But not everyone can sing in a correct way. Some people are naturally tone deaf. So in this paper, we are going to introduce a speech-to-singing synthesis system to help people sing gracefully.

In Saitous study[1] , a speaking voice of reading lyrics could potentially be converted into a singing voice by manually controlling its three acoustic features: the fundamental frequency, phoneme duration, and spectrum. The fundamental frequency controls the pitch of the sound, the duration controls the tempo and the spectrum controls the timbre. Since our goal is to convert a human voice directly to a singing version, we only need to modify the pitch and phoneme duration of the speech signal without changing the timbre of the speech signal. To realize the synthesis, we first detect the pitch of the speech signal and then modify the duration of each phoneme along with changing the pitch.

The rest of this paper is organized as follows; Section 2 introduces the technique methods we used to detect pitch and how we modified it along with duration. Section 3 outlines the experiment process and the result. Section 4 concludes the whole paper.

2. SPEECH TO SINGING SYNTHESIS METHOD

Our speech to singing synthesis system has the following input and output:

INPUT: Speaking lyrics and singing voice, where the singing and speaking signal are from the same person.

OUTPUT:Synthesized singing voice.

The steps of conversion are described as follows:

- Detect the pitch of the speech voice and the singing voice
- Convert the pitch of the speech signal to the pitch of the singing signal as well as modifying the the duration of the pitch-shifted speech signal

The following sub-sections briefly introduces the methods we use.

2.1. Pitch detection

Pitch detection is the fundamental part of this project. Here we use YIN algorithm to detect the pitch.[2] The main idea of the YIN algorithm is to estimate the fundamental frequency (F0) of speech or musical sounds. It is based on the well-known autocorrelation method with a number of modifications that combine to prevent errors. The calculation procedure are as following:

1. Difference equation

Instead of using the conventional method, which is the autocorrelation method, to detect pitch, the YIN algorithm introduces a difference equation:

$$d_t(\tau) = \sum_{j=1}^{W} (x_j - x_{j+\tau})^2 \tag{1}$$

where τ is the lag time.

When a signal amplitude increases with time, the peak of autocorrelation function will grow with the lag time rather than remain constant. The difference function has the advantage that it is immune to the change, as amplitude changes

cause period-to-period dissimilarity to increase with lag in all cases.

2. Cumulative mean normalized difference function

The difference above has one limitation that it must choose the zero-lag dip instead of the period dip. Even set a lower limit is not helpful. To solve this problem, equation (4) tries to dividing each value of the old by its average over shorter-lag values.

$$d'_t(\tau) = \begin{cases} 1, & if \tau = 0 \\ \frac{d_t(\tau)}{\frac{1}{\tau}\sum_{j=1}^{\tau} d_t(j)}, & otherwise \end{cases} \quad (2)$$

This step have several benefits. First, it reduces excessively high errors. Secondly, it could remove the upper frequency limit of the search range, which solves the zero-lag dip problem appeared in step 1. Thirdly, it normalizes the function for the next error-reduction step.

3. Absolute threshold Set an absolute threshold and choose the smallest value of that gives the minimum of $d_t(\tau)$ smaller than that threshold. If none is found, the global minimum is chosen instead.

4. Parabolic interpolation

The following steps are good enough to detect pitch. But sometimes, the pitch may not be detected very accurately. To make the pitch be modified very accurate, the YIN algorithm decided to use parabolic interpolation to "fine-tune" the pitch detection.

2.2. Pitch-shifting and tempo change

2.2.1. Timescale modification and Phase vocoder

Tempo change and pitch-shifting are the most important steps in this system. The very first thing that we tried was using a phase vocoder to modify the tempo and pitch of each phoneme. However, the final result of using the phase vocoder was not satisfactory. Therefore, we had to find another way to modify tempo or pitch. What we decided was to use another technique for tempo change and use the phase vocoder only for the pitch-shifting since using the phase vocoder to change both of them makes an unstable output.

The fundamental technique that should be used for timescale modification is a block processing where you chop the input signal into short segments that have the same length. Re-sampling these segments individually would not work since it would change its pitch as well. To resolve this issue, the Hann window should be used to segment the input signal. It will smooth out each segment such that discontinuities at the end of segments can be handled. Each short segment has a fixed length N, usually in the range of 50 to 100 milliseconds of audio material. Also, overlap-add technique should be used during the process to reduce the discontinuities more. What actually makes the timescale modification in this algorithm is the difference between a synthesis hop size H_s and an analysis hop size H_a. The synthesis hop size is often fixed as $H_s = \frac{N}{2}$ or $\frac{N}{4}$ while the analysis hop size is defined as $H_a = \frac{H_s}{\alpha}$ where α is a stretching factor. Therefore, each analysis segment of the input signal is spaced by H_a:

$$x_m(r) = \begin{cases} x(r + mH_a), & if r \in [1 : N] \\ 0, & otherwise \end{cases} \quad (3)$$

Then, the time-scale modified output signal y of this OLA method:

$$y(r) = \sum_{m \in Z} y_m(r - mH_s), \quad (4)$$

where the synthesis segment y_m of Equation (4) is defined as

$$y_m(r) = \frac{w(r)x_m(r)}{\sum_{n \in Z} w(r - nH_s)}. \quad (5)$$

Please note that $w(r)$ of Equation (5) indicates the Hann window that has the same length as $x_m(r)$. The multiplication in the nominator of Equation (5) represents pointwise computation. And the summation in the denominator normalizes the frame such that it prevents amplitude fluctuation in the output signal. We applied this technique to the speech audio file and stretch the time phoneme by phoneme using the text files that contain the time durations of the phonemes.

The second step of 2.2.1 process is using the phase vocoder to changed the pitch of each phoneme from the time stretched audio file. Like the OLA technique we used for timescale modification, we changed the pitches of the audio file phoneme by phoneme. If we let β be a pitch change factor of each phoneme, the first step is to re-sample the signal of the phoneme that you want to change its pitch with $\frac{f_s}{\beta}$. The next step is to take STFT (Short-time Fourier transform) on the signal and interpolate the spectrogram to achieve time-scale factor which is 1 in this case. For the interpolation process the first step that we need to do is to find time t in the original signal's spectrogram that corresponds to a frame in the interpolated spectrogram. Then, by using its left and right frames, t_1 and t_2, we compute λ:

$$\lambda = \frac{t - t_1}{t_2 - t_1}. \quad (6)$$

Finally, each frame of the linearly interpolated spectrum will have a new maginitude $|Y[k]|$:

$$|Y[k]| = (1 - \lambda)|X_1[k]| + \lambda|X_2[k]|. \quad (7)$$

The last step, phase reconstruction, is the main point of a phase vocoder since it makes sure that the phase change coherent during its synthesis process. We can achieve this coherence by making phase advance from one frame to its next frame of the interpolated spectrogram be the same as the phase advance from $\angle X_1[k]$ to $\angle X_2[k]$:

$$\angle(Y[k]) = \angle(X_1[k]) - \angle(X_2[k]) + \angle(Y_o ld[k]). \quad (8)$$

2.2.2. PSOLA

The PSOLA algorithm is a method which introduced in [3]. It is based on the hypothesis that the input sound is characterized by a pitch, as, for example, the human voice and monophonic musical instruments. The algorithm is composed of two phases: the first phase analyses the pitch of the sound, and the second phase synthesizes a time-stretched version by overlapping and adding time segments extracted by the analysis algorithm.

1. Analysis algorithm

(a) Determination of the pitch period P(t) of the input signal and of time instants (pitch marks) t_i. These pitch marks are in correspondence with the maximum amplitude or glottal pulses at a pitch-synchronous rate during the periodic part of the sound and at a constant rate during the unvoiced portions. In practice P(t) is considered constant P(t) = P(t_i) = t_i+1 - t_i on the time interval $(t_i, t_i + 1)$.

(b) Extraction of a segment centered at every pitch mark ti by using a Hanning window with length $L_i = 2P(t_i)$ (two pitch periods) to ensure fade-in and fade-out.

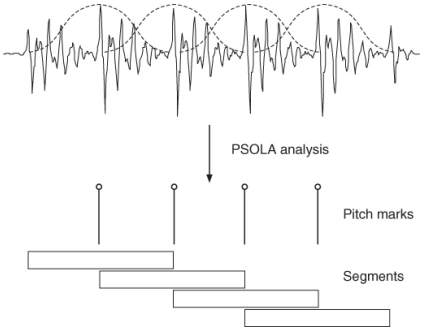


Fig. 1. Diagram of pitch mark analysis

2. Synthesis algorithm

(a). Choice of the corresponding analysis segment i (identified by the time mark t_i) minimizing the time distance $|at_i - \tilde{t_k}|$

(b). Overlap and add the selected segment. Notice that some input segments will be repeated for $a > 1$ (time expansion) or discarded when $a < 1$ (time compression).

(c). Determination of the time instant $\tilde{t}_{k+1}$ where the next synthesis segment will be centered, in order to preserve the local pitch, by the relation $\tilde{t}_{k+1} = \tilde{t}_k + \tilde{P}(\tilde{t}_k) = \tilde{t}_k + P(t_i)$

3. EXPERIMENT

3.1. Dataset

In this project, we use the NUS Sung and Spoken Lyrics Corpus (NUS-48E corpus for short), 48 English songs the lyrics of which are sung and read.The sampling rate of the signals is 44100. All singing recordings have been phonetically transcribed with duration boundaries. We decided to use this data set since it is not easy not only to find a way to detect each phoneme of each song but also to categorize phonemes for us now.

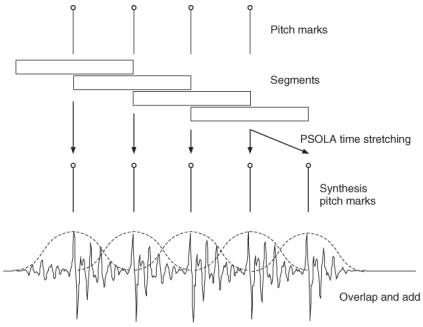
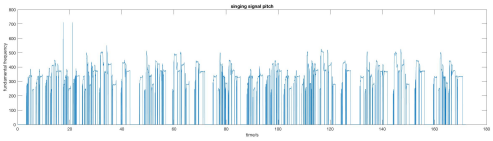


Fig. 2. Diagram of for time stretching synthesis

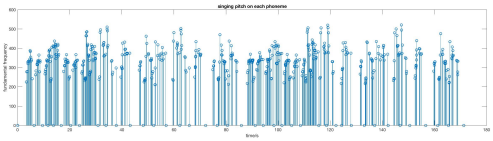
3.2. Pitch detection

As mentioned in section 2, we use YIN algorithm to detect the pitch of both speech signal and singing signal. The time interval between two adjacent estimates is 0.01s. The integration window size is 0.0464. The lowest and the highest possible possible F0 is 200 and 2000 HZ. The the threshold of dips of d_prime is 0.1.

Figure 3 shows the detected pitch of the target singing signal. Figure 3.(a) is the pitch of the whole signal and Figure3.(b) is the pitch of every detected phoneme. As for speech signal, Figure 4 shows the detected pitch of the target singing signal. Figure 4.(a) is the pitch of the whole signal and Figure4.(b) is the pitch of every detected phoneme.

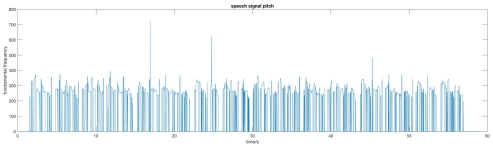


(a) pitch of the singing signal

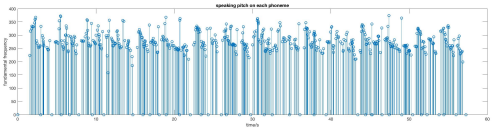


(b) pitch of each phoneme in the singing signal

Fig. 3. Comparison of the pitch of singing signal and the pitch of each phoneme in the singing signal



(a) pitch of the speech signal



(b) pitch of each phoneme in the speech signal

Fig. 4. Comparison of the pitch of speech signal and the pitch of each phoneme in the speech signal

3.3. pitch fitting and tempo change

3.3.1. Timescale modification and Phase vocoder

For this OLA-based timescale modification technique, 0.045 seconds which was used as a fixed frame length. And for the synthesis hop size, we set it to half of the frame length, which is about 0.0225 seconds. As it is explained in Section 2.2.1, the analysis hop size is determined by the strecthing ratio of the phoneme that we want to change. Figure 5 shows the spectrograms of the 422nd phoneme of the song that we used, which is "ow". The first plot shows the spectrogram of the speech audio signal, the second shows that of the converted signal generated by the OLA-based technique and phase vocoder and the last shows that of the singing signal.

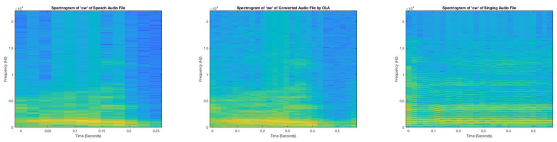


Fig. 5. Spectrum of speech phoneme, synthesized phoneme by OLA and Phase Vocoder and singing phoneme

Figure 5 shows that the converted signal is actually stretched in terms of time scale, but its spectrogram shows a different trend.

3.3.2. PSOLA

The duration and pitch of speech signals are modified by the PSOLA algorithms. The modification can be divided into three cases. First case is dealing with "silence". The synthesized signal will have a same duration length silence as the singing signal at the same phoneme position. Second case is changing the pitch and duration at feasible phoneme. we calculate the pitch shifting factor for each phoneme based on the pitch detection results of singing and speech signals, and time stretching factor based on the duration of each phoneme of speech and singing signals. If the time stretching factor is in the range of 0.5 to 5 and pitch shifting factor is in the range of 0.5 to 5, the pitch and duration of corresponding phoneme will be changed by PSOLA algorithm. At the same time, we use autocorrelation to find fundamental frequency and then find the pitch markers of the phoneme. If the time stretching factor or the pitch shifting factor can't meet the condition mentioned above, the current analysis phoneme will be append to next analysis phoneme. The two joint phoneme will be regarded as a whole one and be tested with the conditions mentioned above again. The joint phoneme will be modified until it satisfies the conditions.

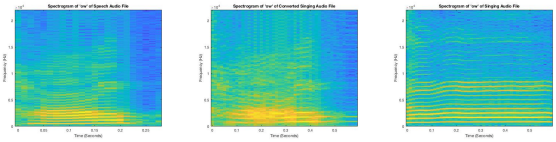


Fig. 6. Spectrum of speech phoneme, synthesized phoneme by PSOLA and singing phoneme

The window size and hop size for fundamental frequency estimation and PSOLA algorithm is 0.02 and 0.01 seconds respectively. Figure 6 indicates the performance of PSOLA algorithm. From left to right is the spectrum of speech 'ow' phoneme, synthesized 'ow' and singing 'ow'. The results indicates that the modification of pitch and duration by PSOLA will introduce some artifacts. Also, it don't have high continuity between each phonemes.

3.4. final audio

The synthesized signal is achieved by performing time stretching and pitching shifting on each phoneme. The pitch shifting and time stretching can be hear clearly. However, there are some artifacts at some phonemes. Further improve should focus on considering the formants and vibrato difference.

4. CONCLUSION

We succeeded to synthesize a singing voice from a speaking voice. Based on the given duration and detected pitch of each phoneme, the speech signal was time stretched and pitch shifted phoneme by phoneme by the combination of timescale modification and phase vocoder and PSOLA, respectively. We achieved good synthesis results at some sentences. The system needs more improvements in continuity, formant and vibration. Moreover, it is required to make this system less dependent on the data set we have been using for this project so the users can use it with any songs which can be improved by finding a way to detect the time steps of phonemes for any songs.

5. REFERENCES

[1] Takeshi Saitou, Masataka Goto, Masashi Unoki, and Masato Akagi, "Speech-to-singing synthesis: Converting speaking voices to singing voices by controlling acoustic features unique to singing voices," pp. 215 – 218, 11 2007.

[2] Alain de Cheveign and Hideki Kawahara, "Yin, a fundamental frequency estimator for speech and music," vol. 111, pp. 1917–30, 05 2002.

[3] Eric Moulines and Francis Charpentier, "Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones," *Speech Commun.*, vol. 9, no. 5-6, pp. 453–467, Dec. 1990.