Arbitrary Modification of Speech Characteristics in Segmental Durations

Kyeomeun Jang, Jiaying Li, Yinuo Wang

***Abstract*—This paper presents a program which allows users to arbitrarily modify the duration of any portion of a given speech signal without changing the essential properties (e.g., pitch contour, power spectrum, etc.) of the original signal. Arbitrary modification means that users can change the duration of any region of the signal by specifying the starting and ending time for modification and the target duration of the specified interval, which can be either a fixed value of duration in the time domain or a scaling factor of the original duration. In addition, modification of multiple intervals is to be permitted, which means both the time duration and the target (duration or scaling factor) can be defined as the sequences in time.**

***Index Terms*—Time-scale modification, arbitrary segmental speech modification, algorithm evaluation, user interaction**

I. Introduction

With a history of over 80 years, algorithms for time-scale modification (TSM) of speech have significant applications in speech data compression [1], music signal playback and so on [2]. The most common studied methods are time domain modification, frequency domain modification, and model based modification.

Although there are lots of well-studied time-scale modification algorithms, the need for arbitrary modification is still not satisfied. Users always hope to modify their files wherever they want during speech editing and audio mixing.

In this paper, our team completed a program which allows arbitrary modification of the duration of any portion of a given speech signal without changing the essential properties. With a well-designed graphical user interface, users can input a "contour" which defines the durational expansion or contraction factor as a function of time and get their expected modified speech. After various evaluations, our program is highly compatible and robust with powerful functions.

II. Theoretical Foundation

To achieve the goal of modifying the speech duration, we need to modify the length of the speech signal without changing other speech characteristics such as speech contour and pitches. Generally, there are three main methods:

- Time domain modification: OLA, SOLA, SOLAFS, TD-PSOLA, WSOLA and so on;
- Frequency domain modification: MSTFTM and so on;
- Model based modification: Phase Vocoder, Sinusoidal Modeling and so on.

The basic idea of OLA is to divide the speech into several consecutive non-overlapping frames, and then repeat or discard some of the speech frames to slow down or speed up the speech rate. OLA simply repeats or discards speech frames, which results in discontinuous waveform between two adjacent frames and leads to the broken pitch, resulting in poor speech quality.

Synchronous Waveform Superposition Method (SOLA) [3] is proposed to solve the discontinuity of the waveform. The algorithm is mainly divided into two stages: framing and synthesis. The framing stage completes the windowing task of the original speech signal. In order to reduce the discontinuity phenomenon, windowing and smoothing are generally performed at the same time . SOLAFS [4], TD-PSOLA [5], WSOLA [6] and other SOLA algorithms are based on SOLA, and they all have some different improvements.

The most representative method in the frequency domain method is MSTFTM (Modified Short-Time Fourier Transform Magnitude) [7], [8], which is based on the short-time Fourier transform. According to the principle of least mean square error, the short-time Fourier transform amplitude spectrum of a time-domain signal is found to approximate the spectrum of an ideal variable-speed signal.

The basic idea of model based method is to establish a model of the speech signal, and then modifying the relevant parameters of the model according to the needs to achieve the purpose of changing the pitch or duration of a speech. The phase vocoder method decomposes the speech into several sinusoidal signals through a band-pass filter, and then adjust the amplitude and phase of the sinusoidal signal to expand or compress the speech signal [9]. As for the sinusoidal modeling method, it is quite similar to the phase vocoder method. It needs to estimate the instantaneous amplitude, instantaneous frequency, instantaneous phase and other parameters of the model. However, the algorithm complexity is too high, especially comparing to its synthesis performance.

In this project, we will choose three typical algorithms for comparison: Phase vocoder, SOLAFS and WSOLA.

A. Phase Vocoder [9]–[12]

There are two stages for phase vocoder. In the first stage, prominent peaks are identified, and the frequency axis is divided into "regions of influence" dominated by each peak. In the second stage, the regions around each peak are shifted, or translated, to new locations, thus achieving the desired frequency modification.

Suppose that the frequency of the input speech signal $x(n)$ is ω, then the expression of $x(n)$ can be defined as:

$$x(n) = Ae^{j\omega n + j\phi} \tag{1}$$

Denote $h(n)$ as the phase vocoder analysis window, and $H(\Omega)$ is the Fourier transform for $h(n)$ at frequency Ω. The STFT of $x(n)$ at time t_a^u and frequency Ω is:

$$X(\Omega, t_a^u) = AH(\Omega - \omega)e^{j\phi} \tag{2}$$

Suppose the speech signal around ω is shifted by $\Delta\omega$, then the short-term output signal at time t_a^u and frequency Ω corresponding to $Y(\Omega, t_a^u)$ is:

$$y_u(n) = h(n)Ae^{j\phi}e^{j(\omega+\Delta\omega)n} \tag{3}$$

Suppose that there are R samples between two frames, the hop size for the phase vocoder is $\Delta\omega R$. Simply detect the pitch by searching the maximum magnitude in its left and right neighbors. For a target time scaling factor β which gives the desired compression or expansion rate:

- Large FFT sizes, low sampling rate: $\Delta\omega = \omega(\beta - 1)$;
- For other cases: $\Delta\omega = \alpha\omega^2 + \omega(\beta - 1)$, where α is a quadratic frequency mapping coefficient.

Once the $\Delta\omega$ is known, we can shift the peaks and adjust the phases. This can be accomplished by simply multiplying by a complex hop size. The rotations should be cumulated from one frame to the next in order to maintain phase-coherence from one frame to the next:

$$Z_{u+1} = Z_u\Delta\omega_{u+1}R \tag{4}$$

where R is the phase-vocoder hop size.

B. SOLA [3], [4], [8]

Denote the input speech signal as $x[n]$, and split it into overlapping windows $x_m[n]$ with a fixed length W, and separated by a fixed analysis distance S_A:

$$x_m[n] = \begin{cases} x[mS_A + n] & \text{for } n = 0, ..., W-1 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

For target compression or expansion rate β, $\beta > 1$ represents expansion and $\beta < 1$ represents compression. The scaling factor $\alpha = 1/\beta$, has opposite meaning. The windows have a new synthesis distance S_S, which is defines as $S_S = \beta S_A$. The synthesis shift used for each window $x_m[n]$ is allowed to vary by an amount k_m in order to maximize the similarity of the data in the overlapping regions before the overlap-add step. Suppose output modification frames:

$$y_m[n] = y[mS_S + k_m + n] \tag{6}$$

then $y_m[n]$ can be recursively defined by:

$$\begin{cases} \beta_m[n]y_m[n] + (1 - \beta_m[n])x_m[n] & n = 0, ..., W_{OV}^m - 1 \\ x_m[n] & n = W_{OV}^m, ...W - 1 \end{cases} \tag{7}$$

where $W_{OV}^m = k_{m-1} - k_m + W - S_S$.

C. SOLAFS [4]

SOLAFS is an improvement of SOLA. During the analysis, the windwos are chosen as:

$$x_m[n] = \begin{cases} x[mS_A + k_m + n] & \text{for } n = 0, ..., W-1 \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

and the output $y_m[n] = y[mS_S + n]$ is recursively formed by:

$$\begin{cases} \beta[n]y_m[n] + (1 - \beta[n])x_m[n] & n = 0, ..., W_{OV} - 1 \\ x_m[n] & n = W_{OV}, ...W - 1 \end{cases} \tag{9}$$

where $W_{OV} = W - S_S$ is the number of points in the overlap region.

A common used similarity measure is the normalized cross-correlatino between x and y in the overlap region:

$$k_m \leftarrow \max_{0 \le k \le K_{max}} R_{xy}^m[k] \tag{10}$$

where K_{max} is the maximum allowable shift from the window's initial starting position, and

$$R_{xy}^m[k] = \frac{r_{xy}^m[k]}{\sqrt{r_{xy}^m[k]r_{yy}^m}} \tag{11}$$

where

$$r_{xy}^m[k] = \sum_{n=0}^{W_{OV}-1} x[mS_A + k + n]y[mS_S + n] \tag{12}$$

$$r_{xx}^m[k] = \sum_{n=0}^{W_{OV}-1} x^2[mS_A + k + n] \tag{13}$$

$$r_{yy}^m = \sum_{n=0}^{W_{OV}-1} y^2[mS_S + n] = x[mS_A + \tau_m + n] \tag{14}$$

where $\tau_m = k_{m-1} + S_S - S_A$. Suppose $0 \le \tau_m \le K_{max}$, at the m^{th} shift, k_m should be determined by

$$\begin{cases} \tau_m = k_{m-1} + (S_S - S_A) & \text{if } 0 \le \tau_m \le K_{max} \\ \max\limits_{0 \le k \le K_{max}} R_{xy}^m[k] & \text{otherwise} \end{cases} \tag{15}$$

D. WSOLA [6], [13]

Based on SOLA algorithm, the STFT of input speech signal $x(n)$ after adding a window becomes:

$$X(\omega, m) = \sum_{n=-\infty}^{+\infty} x(n + m)w(n)e^{-j\omega n} \tag{16}$$

WSOLA uses three measures for the best similar waveform searching algorithm:

- Cross-correlation coefficient;
- Normalised cross-correlation coefficient;
- Cross AMDF coefficient.

Among the three algorithms, WSOLA has the theoretical highest performance.

III. Algorithm Design

In this project, our team designed the code to execute the signal modification based on users' input and three different algorithms. Users can choose start times, end times, target duration (scaling factors or desired duration), and algorithms (SOLAFS, PV-TSM, WSOLA) in the graphic interface panel looks like Fig. 1. A user input contour example is shown as

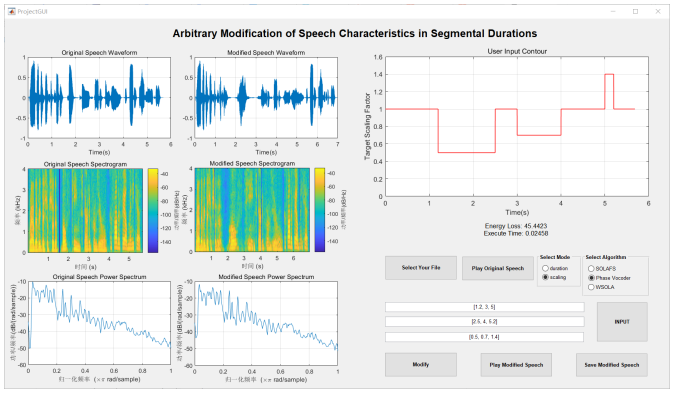

Fig. 1. Graphic User Interface Panel

Fig. 2. The input contour can be segmented and input as the duration arrays.

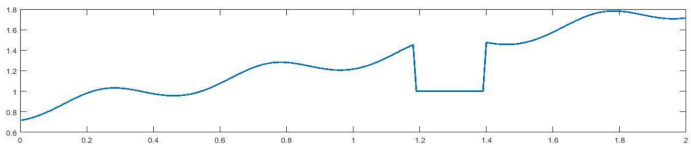

Fig. 2. Contour Input Example. The x-axis indicates the time, and the y-axis represents the scaling factor

The main signal modification process is implemented in the seg_modify function, and the system pipeline is shown in Fig. 3. First, the digital speech signal is segmented into multiple arbitrary fragments based on the user-specified start times and end times and then the time domain input is transformed into discrete samples. For the target duration, our system allows two different input types. Specifically, in the duration mode, user can determine the desired duration of each selected segment defined by start time and end time, and in the scaling mode, user's input of target duration represents the scaling factor of the original signal duration. But both two types of input will be unified as the scaling factor for the visualization as user input contour in the GUI and internal level algorithm implementation.

Then, each block of the segmented signal will be inserted into the modification algorithm that the user chose. After executing the time-scale modification, the modified segments are concatenated with the rest of the unselected original segments to build up the output speech signal.

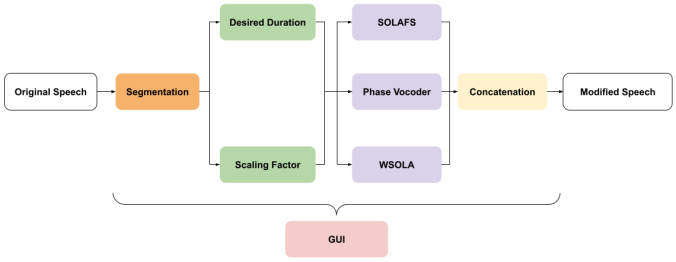

Fig. 3. System Workflow

IV. Result and Evaluation

A. Experiment Result

All the above three methods are implemented and tested in our project, and Fig. 4 shows an experiment result from one of our test cases where the scaling factor is 1.5 from 2-3 seconds, and 0.5 from 4.5-5 seconds. All of them could modify the speech segmental duration as well as keep the original speech characters. We plotted the waveform, spectrogram and power spectral density spectrum of original and modified speech signals below to show the good performance of these three algorithms in time-scale speech signal modification.

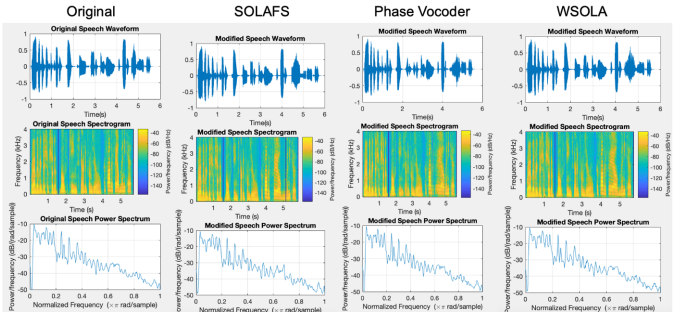

Fig. 4. Modification Result of Different Algorithms

B. Performance Evaluation

Although each of the above algorithms can satisfy the challenge requirements, there still are some differences among them, which need to be illustrated by using some quantitative comparisons. Specifically, we use three indicators, energy loss, execution time and difference of power spectral density, to evaluate their corresponding performance.

Signal energy is a commonly used measurement in speech signal processing. Traditionally, the energy of a signal is defined using the square of the signal magnitude. Here, we define energy loss as the difference between the original speech and the modified speech signal:

$$E_{loss} = E_{in} - E_{out} = \sum_{n=-\infty}^{+\infty} |x_i(n)|^2 - \sum_{n=-\infty}^{+\infty} |x_o(n)|^2 \quad (17)$$

The power spectral density (PSD) provides an easier way of representing the distribution of signal frequency components than the DFT. We used Welch's method [14] to estimate the

PSD of the speech signal and compute the difference between the original signal PSD and the modified signal PSD.

In addition, as a signal processing algorithm, execution efficiency is another crucial indicator we need to consider. Therefore, we implemented an experiment to compare the performance of different algorithms by using the above indicators and controlling all other variables the same, and the experiment results are shown in Table. I. Summarized results

TABLE I
PERFORMANCE COMPARISON OF DIFFERENT ALGORITHMS

	SOLAFS	Phase Vocoder	WSOLA
Energy Loss (dB)	-26.1494	55.8216	-1.5239
Difference of PSD (W/Hz)	0.1220	0.1106	0.0743
Execution Time (ms)	6.86	6.38	14.73

are listed as follows:

- WSOLA is the most robust model among the three algorithms. There was the smallest energy loss and power spectrum density change. Although it yields an accurate result, the execution time is longer than the other models depending on the delta values.
- SOLAFS has greater energy loss and difference in power spectrum density than WSOLA. However, it was more computationally efficient than WSOLA as the execution time was less than half of the WSOLA's.
- Phase Vocoder turns out to be the weakest model. The execution time and power spectrum density differences were similar to those of SOLAFS, but there was a large energy loss.

V. CONCLUSION

In this project, we completed the arbitrary speech modification algorithm and finished the graphical user interface design, which improves the accessibility and usability of our program. After reliable evaluation and assessments, we obtain the conclusion that among the three algorithms, WSOLA algorithm has the highest performance while SOLAFS and Phase Vocoder algorithm are more computationally efficient. Because of the support of multiple algorithms, our program has more adaptability and has very encouraging potential in the application environment.

Due to the limited time and space, we didn't finish achieving more algorithms. In future work, it can be considered to add more algorithms on the existing basis and further optimize the interaction of the graphical user interface.

VI. APPENDIX

Team Project GitHub Repository Link:
https://github.com/allenwang-git/ECE6255-team-project

REFERENCES

[1] J. Driedger and M. Müller, "A review of time-scale modification of music signals," *Applied Sciences*, vol. 6, no. 2, p. 57, 2016.

[2] J. Driedger, "Time-scale modification algorithms for music audio signals," *Master's thesis, Saarland University*, 2011.

[3] J. Makhoul and A. El-Jaroudi, "Time-scale modification in medium to low rate speech coding," in *ICASSP'86. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 11. IEEE, 1986, pp. 1705–1708.

[4] D. Hejna and B. R. Musicus, "The solafs time-scale modification algorithm," *Bolt, Beranek and Newman (BBN) Technical Report*, 1991.

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

[6] W. Verhelst and M. Roelands, "An overlap-add technique based on waveform similarity (wsola) for high quality time-scale modification of speech," in *1993 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2. IEEE, 1993, pp. 554–557.

[7] D. Griffin and J. Lim, "Signal estimation from modified short-time fourier transform," *IEEE Transactions on acoustics, speech, and signal processing*, vol. 32, no. 2, pp. 236–243, 1984.

[8] S. Roucos and A. Wilgus, "High quality time-scale modification for speech," in *ICASSP'85. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 10. IEEE, 1985, pp. 493–496.

[9] J. Laroche and M. Dolson, "New phase-vocoder techniques are real-time pitch shifting, chorusing, harmonizing, and other exotic audio modifications," *Journal of the Audio Engineering Society*, vol. 47, no. 11, pp. 928–936, 1999.

[10] M. Portnoff, "Implementation of the digital phase vocoder using the fast fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, no. 3, pp. 243–248, 1976.

[11] M. Dolson, "The phase vocoder: A tutorial," *Computer Music Journal*, vol. 10, no. 4, pp. 14–27, 1986.

[12] D. P. Ellis, "A phase vocoder in matlab," *Columbia University (http://www. ee. columbia. edu/dpwe/resources/matlab/pvoc/)*, 2002.

[13] J. Driedger and M. Müller, "Tsm toolbox: Matlab implementations of time-scale modification algorithms." in *DAFx*. Citeseer, 2014, pp. 249–256.

[14] P. Welch, "The use of fast fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms," *IEEE Transactions on audio and electroacoustics*, vol. 15, no. 2, pp. 70–73, 1967.