Time-Stretching and Pitch-Scaling of Speech Signals

ECE 6255

Digital Speech Processing

Spring 2017

Group 3

William Morlan

Alex Parisi

Vasundhara Rawat

Analysis

Time stretching is the process of changing the speed or duration of an audio signal without affecting its pitch. Conversely, pitch shifting is the process by which the pitch is modified without affecting the speed. Certain methods are able to affect both these properties simultaneously. These processes are used to match the pitches and tempos of two pre-recorded clips for mixing when the clips cannot be reperformed or resampled.

The simplest and most intuitive way of changing pitch would be to resample the signal. In this method, the signal is essentially reconstructed from the given samples before being sampled again at the required rate. When the new samples are played at the original sampling frequency, the audio clip sounds faster or slower. Unfortunately, the frequencies in the sample are always scaled at the same rate as the speed, transposing its perceived pitch up or down in the process. In other words, slowing down the recording results in the pitch being lowered while speeding it up raises the pitch. This is often times undesirable as the two effects cannot be varied independently of the other.

To preserve audio signal's pitch when applying time scaling, many time-scale modification (TSM) procedures follow a frame-based approach [1]. The first step is to split the signal into short analysis frames of fixed length. These analysis frames are designed to be separated by a fixed number of samples, otherwise known as the analysis hopsize. The analysis frames are then temporally relocated to have a synthesis hopsize that differs from the analysis hopsize. The new frames are then used to reconstruct the signal with the time scale modified. This frame relocation method results in a modification of the signal's duration by a stretching factor. This method has the added advantage of preventing undesirable artifacts such as phase discontinuities or amplitude fluctuations that might be have resulted otherwise.

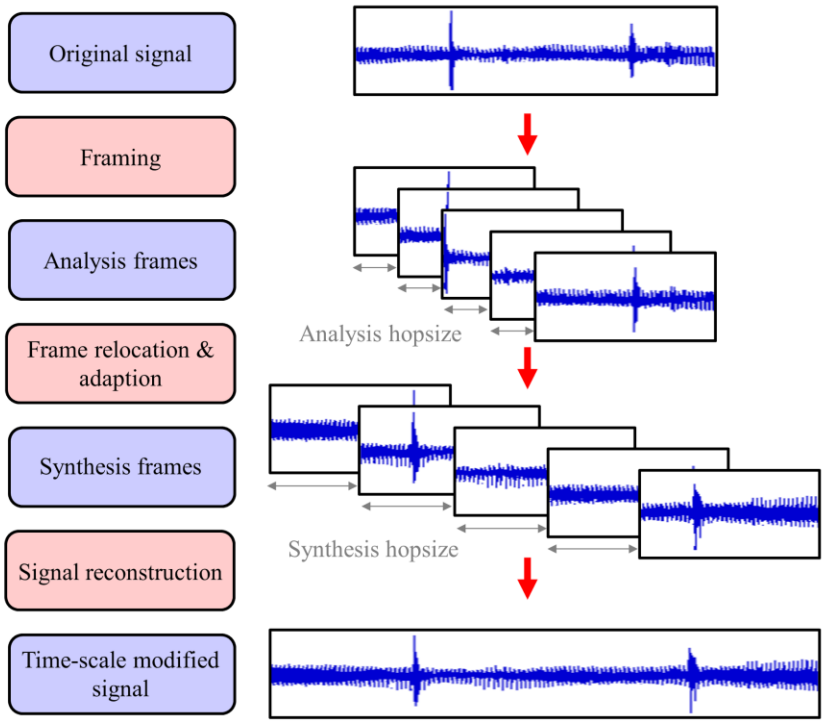

Fig1. Frame - based approach of many TSM procedures

One way of stretching the length of a signal without affecting the pitch is to build a phase vocoder in the manner of Flanagan, Golden, and Portnoff. The steps include:

- computing the instantaneous frequency/amplitude relationship of the signal using the STFT, which is the discrete Fourier transform of a short, overlapping and smoothly windowed block of samples
- applying some processing to the Fourier transform magnitudes and phases (like resampling the FFT blocks)
- performing an inverse STFT by taking the inverse Fourier transform on each chunk and adding the resulting waveform chunks, also called overlap and add (OLA) [2].

Another method for time stretching relies on the spectral model of the signal. In this method, peaks are identified in frames using the STFT of the signal, and sinusoidal "tracks" are created by connecting peaks in adjacent frames. The tracks are then re-synthesized at a new time scale. This method can yield good results on both polyphonic and percussive material, especially when

the signal is separated into sub-bands. However, this method is more computationally demanding than other methods [3].

Rabiner and Schafer introduced a technique called time-domain harmonic scaling [4] also called the synchronized overlap-add method (SOLA). It utilizes pitch detection algorithm to find the period of a given section of the wave and crossfades one period into another. The performance of this method is often faster than the phase vocoder but fails when the autocorrelation inaccurately measures the period of a signal (in case of complex harmonics).

Implementation

We implemented our two-stage phase vocoder in MATLAB with help from the DSP System Toolbox - this should be included in any MATLAB r2016b installation provided by Georgia Tech's OIT. We designed two functions - one that would extract the desired block operations information, and another to process the extracted signal. Our block operations file performed the operations starting with those that occur latest in the audio signal - this is because any time-stretching or -shrinking will affect the overall length of the signal, and will throw off the indexing of any blocks that would occur after. This block operation will also "chop-up" the original signal into the specified section in which we want to process.

This information is then fed into the phase vocoder function. This function actually contains two instances of the phase vocoder, one in which we perform pitch-scaling, and the other in which we perform time-stretching/shrinking. We perform the time-stretching/shrinking second because if the sampling frequency is changed, as it does in pitch-scaling, after time-stretching it applied, it will not only change the pitch, but also the duration. To avoid this problem, we perform pitch-scaling first and get any changes to the sampling frequency out of the way.

For more information regarding the MATLAB files or how they were designed, please look at the README file included with this report.

Results

The program appears to work as designed, based on non-scientific observations by this humble engineer. The signal lengthens to the appropriate rate, and the pitch is set to roughly the intended level. The program works correctly with both a single block (demo here), multiple blocks (demo here), time-stretching (demo here), pitch scaling (demo here), or a combination of all of the above (demo here). However, the program does not completely account for the case where a block begins or ends in the middle of a word (demos here); in such a case, the audio sounds mangled and broken (as an experiment, try changing the pitch of your own voice in the middle of a word), almost certainly resulting in the loss of whatever the original message was. Therefore, it is necessary to choose the boundary of the blocks carefully, such that the boundaries coincide with the end of a word or sentence; it is therefore usually necessary to analyze the signal to be modified in an audio analysis program such as Audacity, in order to accurately determine the appropriate boundaries for the blocks.

Additionally, it is also necessary to estimate the pitch of the unmodified block, as determined by Matlab. Given that speech signals are rarely monotone, the estimate for the original pitch may vary based on which program is used. The program will perform the appropriate pitch shift relative to whatever pitch matlab perceives for the original signal; for example, if matlab detects a pitch of 50 mels, and the operation file specifies 100 mels for that block, then the pitch of the block will be doubled. Any further work on this project will involve more accurate methods of determining the original pitch, therefore allowing the pitch to scale more accurately relative to the op file.

References

[1] Jonathan Driedger and Meinard Müller (2016). "A Review of Time-Scale Modification of Music Signals". *Applied Sciences*. **6** (2): 57. doi:10.3390/app6020057

[2] Jont B. Allen (June 1977). "Short Time Spectral Analysis, Synthesis, and Modification by Discrete Fourier Transform". *IEEE Transactions on Acoustics, Speech, and Signal Processing*. ASSP-25 (3): 235–238.

[3] McAulay, R. J.; Quatieri, T. F. (1988), "Speech Processing Based on a Sinusoidal Model" (PDF), *The Lincoln Laboratory Journal*, **1** (2): 153–167

[4] David Malah (April 1979). "Time-domain algorithms for harmonic bandwidth reduction and time scaling of speech signals". *IEEE Transactions on Acoustics, Speech, and Signal Processing*. ASSP-27 (2): 121–133.