A VHDL Implementation of Voice Morphing using Phase Vocoder Algorithm

Rameshwari Tarunkar
D.I.M.A.T.
Raipur

Balram Timande
D.I.M.A.T.
Raipur

ABSTRACT
The phase vocoder is a tool used to perform time-stretching and pitch-shifting on recorded sounds. Voice morphing is the process of producing intermediate or hybrid voices between the utterances of two speakers. It can also be defined as the process of gradually transforming the voice of one speaker to that of another. Like image morphing, speech morphing aims to preserve the shared characteristics of the starting and final signals, while generating a smooth transition between them. This paper describes the Phase vocoder method of voice morphing. We extract the feature difference of source and targeted voice applies this phase different to the source voice.

General Terms-speech morphing, ST-FFT, vocoders

Keywords- Voice morph, Phase vocoder, Speech

I. INTRODUCTION

Speech is usually characterized as voiced, unvoiced or transient forms. Voiced speech is produced by an air flow of pulses caused by the vibration of the vocal cords. The resulting signal could be described as quasi-periodic waveform with high energy and high adjacent sample correlation. The two broad categories of pitch-estimation algorithms are time-domain algorithms and frequency-domain algorithms. Time domain algorithms attempt to determine pitch directly from the speech waveform. Frequency-domain algorithms use some form of spectral analysis to determine the pitch period [1].

Voice morphing is the transition of one speech signal into another. Voice morphing technology enables a user to transform one person's speech pattern into another person's pattern with distinct characteristics. Image morphing, speech morphing preserve the shared characteristics of the starting and final signals, while generating a smooth transition between them. Speech morphing is analogous to image morphing. In image morphing the in-between images show one face smoothly changing its shape and texture until it turns into the target face. In Speech morphing one speech signal should smoothly change into another, keeping the shared characteristics of the starting and ending signals but smoothly changing the other properties.

Voice morphing allows speech model to be duplicated and an exact copy of a person's voice be completed were then we can say something to the operator wishes it to speak come out in the voice of someone-else. Voice morphing plays a great role in military psychological fighting and subversion. It is mainly used in conjunction by using voice morphing recorded telephone conversations we can use as evidence in courts. Voice morphing is a powerful combat zone weapon which can be used to give false information to the opponent group, come out from their own leader. In the last few years many papers have addressed the issue of voice morphing using different signal processing techniques [2-3]. Most methods developed were single-scale methods based on the interpolation of speech parameters and modeling of the speech signals using formant frequencies [4], Linear Prediction Coding Cepstrum coefficients, Line Spectral Frequencies and harmonic-plus-noise model parameters [5]. Other methods are based on mixed time- and frequency-domain methods to alter the pitch, duration and spectral features. The methods suffer from absence of detailed information during the extraction of formant coefficients and the excitation signal which results in the limitation on accurate estimation of parameters as well as distortion caused during synthesis of target speech.

II. METHODOLOGY

The phase-vocoder, see Flanagan and Golden (1966) is a frequency-domain technique based on the Short Time Fourier Transform (STFT). The STFT characterizes the spectral behavior of a signal x (n) along time and can be written as:

$$X(n,k) = \sum_{m=-\infty}^{m-\infty} x(m).h(n-m).e^{-j\frac{2\pi mk}{N}}$$

Where (n, k) is the complex time-varying spectrum with the frequency bin k and time index, n. At each time index, the input signal (m) is weighted by a finite length window $h(n-m)$ and then computed its spectrum. The most common understanding of the phase-vocoder technique is the filter bank interpretation. This filter bank is a parallel bank of N band-pass filters with the following impulse response

$$h_k = h(n).e^{-j\frac{2\pi nk}{N}}, \; k=0,1,\ldots\ldots N-1$$

Which means the same filter shape is shifted along frequency by $2\pi/N$ radians steps. The band-pass signal of band k, (n) is obtained by filtering the input signal x (n) with filter $hk(n)$ as showed in the following expression

$$y_k(n) = x(m) * h_k(n)$$

$$= \sum_{m=-\infty}^{m-\infty} x(m).h_k(n-m)$$

$$= \sum_{m=-\infty}^{m-\infty} x(m).h(n-m).e^{\frac{2\pi(n-m)k}{N}}$$

$$= e^{j\frac{2\pi(n)k}{N}} \sum_{m=-\infty}^{m-\infty} x(m).e^{-j\frac{2\pi(m)k}{N}}.h(n-m)$$

$$= e^{j\frac{2\pi(n)k}{N}} X(n,k)$$

And the output signal y (n) is the sum of all band-pass signals

$$y(n) = \sum_{k=0}^{k=N-1} x(m).h_k(n)$$

$$= \sum_{k=0}^{k=N-1} X(n,k).e^{-j\frac{2\pi(n)k}{N}}$$

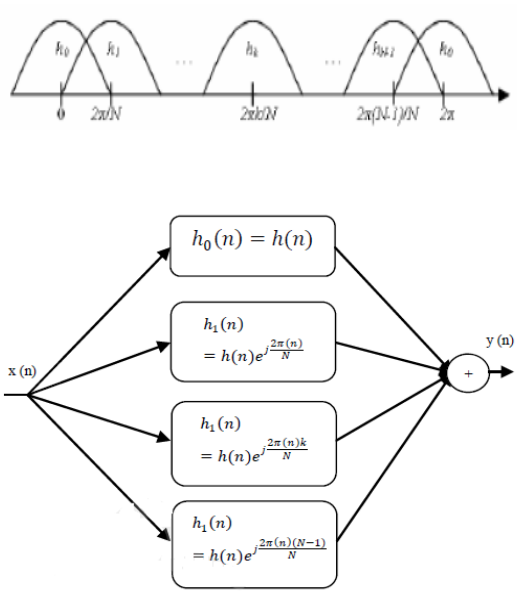

Figure1: Filter bank descriptions of the short time Fourier transform.

The frequency bands on the top show the displacement of each of the band-pass filters. Thus, for a certain time index n, y (n) will be defined by an N length vector containing an estimation of each band's energy

$$y_k(n) = X(n,k)e^{j\frac{2\pi(n)k}{N}}$$

$$= |X(n,k)|.e^{j\varphi x(n,k)}.j\frac{2\pi(n)k}{N}$$

$$|y_k(n)| = |X(n,k)|$$

$$\varphi_{y_k} = \varphi_x(n,k) + \frac{2\pi(n)k}{N}$$

Before synthesis, spectral modifications can be applied to the resulting analysis data. If we term the modified data $y_k^{mod}(n)$, the synthesis window w(n), the synthesis hop size H, and $y_m(n)$ is the Inverse Fourier Transform of the modified short time spectra

$$y_m(n) = 1/N \sum_{k=0}^{k=N-1} y_k^{mod}(n).e^{j\frac{2\pi(n)k}{N}}$$

$$= 1/N \sum_{k=0}^{k=N-1} \left(X(mH,k).e^{j\frac{2\pi(m)k}{N}}\right)^{mod} .e^{j\frac{2\pi(n)k}{N}}$$

The resulting synthesis signal s (n) can be written as the overlap and add of the windowed $ym(n)$'s.

$$s(n) = \sum_{m=-\infty}^{m=\infty} w(n-m.H).y_k(n)$$

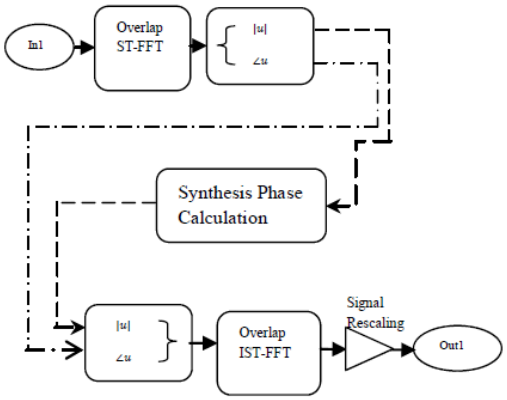

Figure2: Flow diagram of Voice morphing

G etting Sound into the Phase Vocoder

The first thing we need to do is to be able to is to access our sound file directly in our pfft~ sub-patch and perform an FFT directly on the sound, without sending it into any fftin~ inlets of the pfft~. Why do we need to do it this way? Since the whole point of a phase vocoder is to perform time stretching and compression on an existing sound, we need to be able to access that sound directly via a buffer~ object and perform the FFT at any given playback speed and overlap. In order to have independent transposition and playback speed, a phase vocoder needs independent playback of the sound to be transformed for each FFT overlap (in our case 4). Each playback frame needs to be synchronized with its respective FFT frame. Therefore we cannot send a single copy of the sound we wish to transform into the pfft~, but need to play a slice of the sound into each of the four overlap frames. Since we cannot send one slice of the sound into the pfft~ object (it keeps track of its input history and considers its input to be continuous sound, not individual slices of sound), we cannot use the fftin~ object inside the pfft~, but must do our FFT processing using the fft~ object. The fft~ object performs a full-spectrum FFT (i.e. mirrored), so we consequently need to make the pfft~ object behave the same way, so the fft~ can work in sync with the FFT frames processed inside the pfft~ object. We need to make a few changes to the default way that pfft~ deals with the STFT.

We first need to tell the pfft~ object to process full-spectrum FFT frames, instead of the default "spectral frame" which is half the FFT size (up to half the Nyquist). This is easily accomplished by adding a non-zero fifth argument to the pfft~ object. Because the full-spectrum argument is the fifth argument, we must supply all the other arguments before it,

including the fourth argument, the start onset, which will be set to the default value of zero.

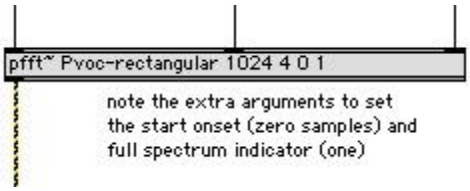

Fig. 3 – Additional Full-Spectrum Argument to the pfft~ Object

Next, because the fftin~ and fftout~ objects perform the FFT calculation at zero phase with respect to the FFT (the first sample in the windowed frame sent to the FFT is the middle of the window), and the traditional fft~ and ifft~ objects perform the FFT 180 degrees out of phase, we need to make sure any fftin~ and fftout~ objects in our patch have the same FFT phase offset used in the fft~ objects. We do this by specifying a phase offset to the fftin~ and fftout~ objects. A phase value of 0.5 means 180 degrees out of phase, so this is the value we want. While we do not need the fftin~ object in our pfft~, we can still make use of the convenience of the fftout~ object in order to get the properly windowed and overlap-added result out of our pfft~. The automatic windowing in the fftout~ object should behave like our manual windowing with the fft~ objects.

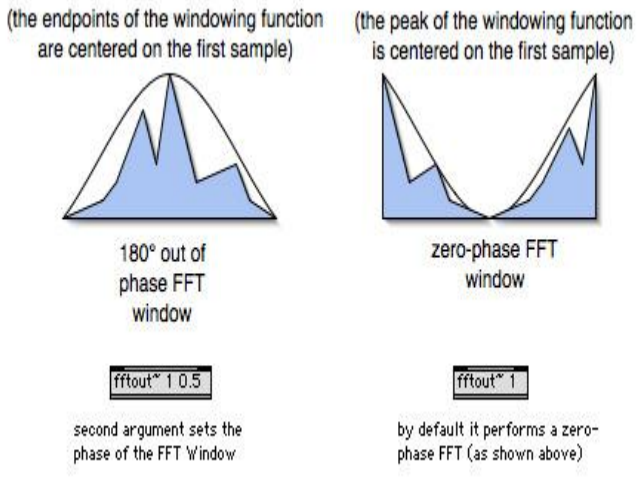

Fig. 4 – Zero-Phase FFT Window

Now we are ready to start constructing our phase vocoder sub-patch.

III. SIMUATION RESULTS

The program for FPGA is developed using the concept of phase vocoder in VHDL. The simulation is done on modelsim6.3f.

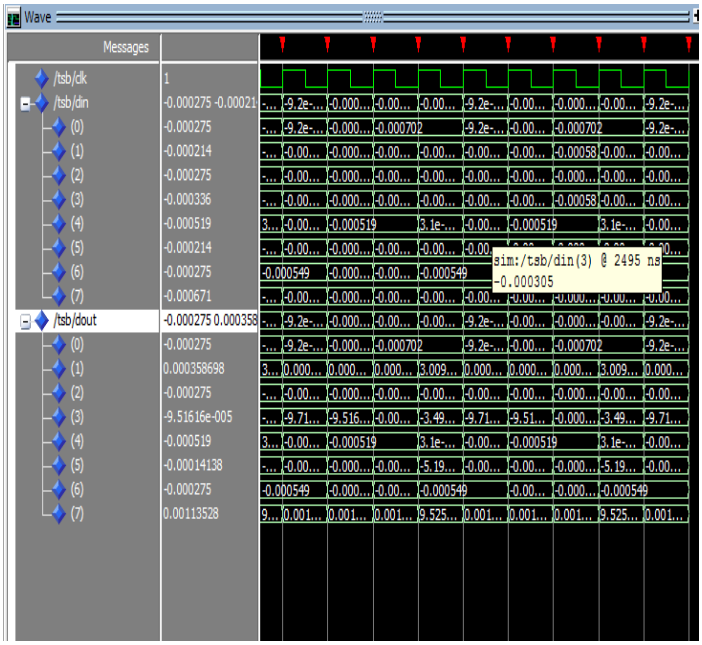

Figure below showing the numeric results in waveforms:

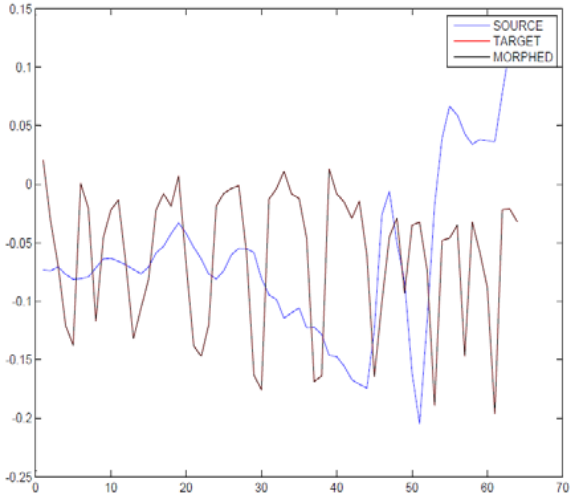

Figure 4: Waveforms of Voice Signal

IV. CONCLUSION

In this study, a new speech morphing algorithm is presented. The aim is to produce natural sounding hybrid voices between two speakers, uttering the same content. The algorithm is based on the concept phase vocoding, where the pitch of source voice is shifted by the phase difference between source and target Voice. Results show the effectiveness of work. To sum up what we've just covered, the basic structure of the phase vocoder requires four overlapping pairs of buffer reads and four overlapping pairs of FFTs. The buffer/FFT pairs are exactly one quarter frame apart. The FFT pairs allow us to calculate the phase differences between where we are in a sound and where we were a quarter frames ago. Then, for each of the four pairs, we simply add their respective phase difference to the previous phase from a full frame before, accumulating our running phase. We do all this with Cartesian coordinates (real and imaginary values making use of complex arithmetic) using fft objects inside a pfft~ that is

running in full-spectrum mode with a 180-degree phase offset for the fftout object so that it runs in phase with the fft~ objects.

V. REFRENCES

[1] Allam Mousa, Speech Segmentation in Synthesized Speech Morphing Using Pitch Shifting, *the International Arab Journal of Information Technology, Vol. 8, No. 2, April 2011*.

[2] D.H.Klatt "Review of Text-to-Speech Conversion for English" Journal of the Acoustical Society of America Vol.67.pp.971-995. 1980.

[3] S. S. Agrawal. N. Pinto. R. Verma. A. S. Sarma and K.N. Stevens, "Synthesizing high Quality Hindi Speech Using KLSY88."Journal of Acoustic Society of India. Vol.20 No. 1-4, 1992.

[4] R. Verma, A. S. Sarma , K.N. Stevens, S. S. Agrawal, N. Shrotiya, "On the Development of Text-to speech System for Hindi," proceedings of ICPhS 95.Stockholm.1995

[5] D.H.Klatt "software for a cascade/parallel format synthesizer" Journal of the Acoustical Society of America **82**(3):737-93.

[6] Improved Phase Vocoder Time-Scale Modification of Audio Jean Laroche, and Mark Dolson IEEE Transactions on Speech and Audio Processing, Vol. 7, no.3 May 1999.

[7] Traditional (?) Implementations of a Phase-Vocoder Daniel Arfib, et. al. Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-00) December, 2000.

[8] "An Efficient phase vocoder of voice morphing using VHDL", International Journal of Digital Application & Contemporary research Website: www.ijdacr.com (Volume 1, Issue 3, October 2012)