Time-Scale Modification of Audio Signals Using the Dual-Tree Complex Wavelet Transform

APPROVED BY
SUPERVISING COMMITTEE:

Brian L. Evans, Supervisor

Bruce Pennycook

Russell F. Pinkston

Dedicated to my parents, Harry and Sandra Livingston in appreciation of their unwavering support along every step of my academic, professional and personal path.

Time-Scale Modification of Audio Signals Using the Dual-Tree Complex Wavelet Transform

by

Jeffrey B. Livingston

REPORT

Presented to the Faculty of the Graduate School of
The University of Texas at Austin
in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE IN ENGINEERING

THE UNIVERSITY OF TEXAS AT AUSTIN
December 2006

TIME-SCALE MODIFICATION OF AUDIO SIGNALS USING THE DUAL-TREE COMPLEX WAVELET TRANSFORM

JEFFREY B. LIVINGSTON M.S.E.

The University of Texas at Austin, 2006

Supervisor: Brian L. Evans

ABSTRACT

The use of the wavelet transform in place of the short-time Fourier transform (STFT) in the phase-vocoder algorithm for time-scaling audio signals has been investigated in the past, motivated by the fact that the wavelet transform offers variable time-frequency resolution that can efficiently and precisely capture audio signal information in a manner very well-matched to human auditory perception characteristics. Despite this, little has emerged in the audio processing literature likely due to inherent limitations of traditional forms of the wavelet transform, such as lack of phase information from the discrete wavelet transform (DWT), and high computational cost and lack of available inverse transform implementations for the continuous wavelet transform (CWT). In this paper, a new wavelet transform based phase-vocoder algorithm is presented that uses a new form of the DWT, the Dual-Tree Complex Wavelet Transform, DT-CWT, which overcomes many of the deficiencies of the older DWT forms.

A preliminary implementation of the algorithm in Matlab resulted in output that was time-stretched as desired, but with the addition of erroneous frequency components due to instantaneous frequency estimation errors caused by the insufficiently narrow octave band frequency resolution of the fully decimated DT-CWT. Use of the wavelet packet transform, (WPT) with approximately 1/3rd octave logarithmically spaced subbands instead of the octave band, fully decimated DT-CWT is proposed as a solution to remedy the artifacts resulting from inadequate frequency resolution.

Table of Contents

Abstract . . . . **iv**

Table of Contents. . . . . **v**

List of Tables and Figures. . . . . **vii**

1.0 Intoduction. **1**

2.0 Overview of Time-Scaling Techniques. **1**

- 2.1 Time Domain Techniques: Time-Segment Processing. **1**
  - 2.1.1 Overlap-add (OLA) **2**
  - 2.1.2 Synchronous Overlap-Add (SOLA) **2**
  - 2.1.3 Time-Domain Pitch-Synchronous Overlap and Add (TD-PSOLA) . . . **3**
- 2.2 Signal Model Analysis/Synthesis. **3**
- 2.3 Phase-Vocoder Time-Scaling Algorithm. **5**
  - 2.3.1 STFT Analysis/Synthesis. **5**
  - 2.3.2 Time-Scale Modification. **6**
  - 2.3.3 Performance Considerations of the STFT Based Phase-Vocoder. **9**
    - 2.3.3.1 Window Size: Frequency vs Temporal Resolution. **9**
    - 2.3.3.2 Effects of Different Window Types **10**
    - 2.3.3.3 Window Overlapping. **11**
    - 2.3.3.4 STFT Phase-Vocoder Artifact Causes. **12**
    - 2.3.3.5 Phase-Vocoder Improvements. **14**
    - 2.3.3.6 Overview of STFT Phase-Vocoder Shortcomings. **15**

3.0 Wavelet Transform Based Phase-Vocoder. **16**

- 3.1 Wavelet Transform Analysis for Audio Signals. **16**
- 3.2 Continuous Wavelet Transform (CWT) **17**
- 3.3 Discrete Wavelet Transform (DWT), and its Limitations. **18**
- 3.4 Dual-Tree Complex Wavelet Transform (DT-CWT) **20**
  - 3.4.1 DT-CWT Properties for Phase-Vocoder Time-Scaling. **20**
  - 3.4.2 Kingsbury's Dual-Tree Complex Wavelet Transform (DT-CWT(K)) **21**

3.4.2.1 DT-CWT Filter Design for Shift Invariance and Perfect Reconstruction (PR) **22**

3.5 Using the DT-CWT in the Phase-Vocoder Algorithm. **24**

4.0 DT-CWT Phase-Vocoder Algorithm Implementation **27**

4.1 Input Signal Considerations for DWT Analysis. **27**

4.2 DT-CWT Calculation. **28**

4.3 Analysis, Time-Scale Modification and Resynthesis Stages. **29**

4.4 Handling Boundary Effects Between Successive Transform Windows **29**

5.0 Evaluation of the Algorithm Implementation. **30**

5.1 Discussion of Possible Sources of Artifacts. **31**

5.2 Discussion of Possible Remedies of Artifacts. **34**

5.3 Complexity Comparisons of Phase-Vocoder Implementations. **36**

6.0 Conclusion. **37**

7.0 Future Work. **38**

Bibliography. **39**

Vita. **42**

List of Tables and Figures

Figures

1 Analysis filterbank structure of the DT-CWT. 22

2 Illustration of shift-invariance of the DT-CWT: Wavelet and scaling function components at scale 1 to 4 of 16 shifted step responses for (a) DT-CWT and (b) real DWT. 24

3 Spectrogram of 1 second 500Hz sinewave (left). Spectrogram of 500Hz sine wave time-stretched by a factor of 2 by the DT-CWT phase-vocoder (right). 32

4 Spectrograms of a 500Hz sine wave (44.1 kHz sampling rate) time-stretched by a factor of two by the STFT phase-vocoder with 1024 sample window, 4x overlap factor (left), STFT phase-vocoder with 128 sample window, 4x overlap factor (middle), and DT-CWT phase-vocoder (right). 33

5 Critical Bandwidth and Musical Intervals versus Frequency. The Minor Third (quarter octave) curve corresponds to 25%, 25%, 50% octave partitioning. 35

Tables

1 Complexity of transforms used in each of the phase-vocoder implementations. N refers to the transform window size, L refers to filter length. 36

1.0 Introduction

In the simplest terms, time-scale modification refers to changing the duration of a signal without modifying its perceived frequency content. In music, this is analogous to changing the tempo of a piece of music without changing the key. In speech, time-scaling is analogous to speeding up or slowing down ones words, but leaving the pitch, tone and inflection the same. Numerous time-scaling techniques have been developed over the past few decades, most of which generally fall into one of three categories: Time domain modification, “phase-vocoder” frequency domain techniques, and analysis/resynthesis using signal models [1]. Each of these types, and their constituent subtypes, carry with them their own set of advantages and disadvantages with regard to computational complexity vs. output quality. In this paper, we propose a new variant of the phase-vocoder technique that uses a recently developed form of the Discrete Wavelet Transform (DWT), the Dual-Tree Complex Wavelet Transform (CD-DWT). This new variant possesses numerous advantageous properties for processing audio signals and potentially offers improved quality and comparable or reduced complexity compared to other current high quality time-scale modification techniques.

2.0 Overview of Time-Scaling Techniques

2.1 Time Domain Techniques: Time-segment processing

In time-segment processing, the basic idea behind the time-stretching technique is to divide the input sound into segments, then if the sound is to be shortened, some segments are discarded, or if the sound is to be lengthened, some segments are repeated. In general, all time-segment processing techniques are based on overlapping and adding adjacent segments extracted from the input signal.

2.1.1 Overlap-Add (OLA)

Overlap-add (OLA) techniques are generally the most computationally inexpensive of all the time-scaling techniques since the basic algorithm requires only simple read/write pointer manipulation and accumulate instructions. For basic time-scale compression, small windowed segments are extracted at time t_i and added to the output at time $t'_i = \alpha t_i$ where α is the time scale factor. The main artifact from the OLA technique comes from the amplitude and phase discontinuity at the boundary of the segments which causes pitch period discontinuities and consequent distortions that are detrimental to signal quality.

2.1.2 Synchronous Overlap-Add (SOLA)

One strategy for reducing the artifacts associated with the OLA technique is to modify the offset for placing each time-segment within a small range around the time-scale factor offset so that the cross-correlation between the overlapping samples is maximized for each pair of overlapping segments[1]. This technique is referred to as Synchronous Overlap-Add (SOLA). The SOLA technique does a much better job at preserving the pitch, magnitude and phase relationships of the time-scaled signal.

2.1.3 Time-Domain Pitch-Synchronous Overlap Add (TD-PSOLA)

Time-Domain Pitch-Synchronous Overlap and Add (TD-PSOLA) is a variation of the SOLA technique in which the signal is first analyzed to identify local pitch across the signal, and local pitch information is used to adjust a variable segment size parameter, and a variable segment offset parameter to preserve the local pitch (fundamental frequency) while achieving a desired time-scale change[1].

Although the SOLA and TD-PSOLA techniques significantly reduce the inherent distortions of OLA methods, noticeable artifacts still remain in the output, such as "buzziness" generated by regular repetition of identical segments in noisy signal segments and smearing of transients when they are stretched in time. The overall character of the undesirable artifacts of time segment techniques are often described as a "choppy" or "granulated" sound. Frequency domain processing methods, described later, though more computationally intensive, are generally considered to produce higher quality results.

2.2 Signal Model Analysis/Synthesis

Signal modeling techniques model sounds as a sum of elementary sinusoidal components called partials. These techniques start by extracting partials, in the form of time dependent magnitude and instantaneous phase data, from a signal via time-frequency analysis, usually a Fourier transform based analysis, such as the short-time Fourier transform (STFT). The decomposition into individual partials is expressed in the additive synthesis equation:

$$s(t) = \sum_{k=1}^{K} a_k(t) \cos \phi_k(t)$$

where $a_k(t)$ is the time dependent magnitude and $\phi_k(t)$ is the instantaneous phase of the *k*-th partial. The instantaneous phase and instantaneous frequency are related by:

$$\phi_k(t) = \int_0^t \omega_k(\tau) d\tau$$

As its name implies, the additive synthesis expression is used to synthesize a signal from the time dependent magnitude and instantaneous phase data. Time-scaling is then achieved by modifying the phase and magnitude data before synthesizing. The modification is a time mapping function, $t \to t' = T(t) = \alpha t$. To time-stretch a signal, we set $|\alpha| > 1$, or to compress in time we set $|\alpha| < 1$. The time-scaled signal is thus:

$$s'(t') = \sum_{k=1}^{K} a_k(t') \cos(\alpha \phi_k(t'))$$

While this time-scaling method has the nice feature of being mathematically straightforward, it only achieves good results for signals consisting of smoothly varying pure tones. It fails for signals containing significant noise components and does a poor job of resynthesizing transients: attacks are smoothed and noise sounds artificial [1]. In addition it is computationally expensive, due to the high overhead for calculation of the cosine functions. The phase-vocoder technique, discussed below, is based on the sum of partials signal model, but it performs the synthesis stage with the inverse STFT, which is more computationally efficient than explicit computation of the additive synthesis equation.

2.3 Phase-Vocoder Time-Scaling Algorithm

The essence of time-scaling is the modification of a signal's temporal evolution while its local spectral characteristics are left unchanged. Phase-vocoder based time-scaling techniques accomplish this via a sequence of analysis, modification and resynthesis performed in the frequency domain [2].

2.3.1 STFT Analysis/Synthesis

The analysis stage of the phase-vocoder is a short-time Fourier transform (STFT). The STFT breaks up a signal into equal length blocks, each starting at the beginning of equally spaced time intervals, and performs the Fourier transform on each block. The Fourier transform is a technique that decomposes a signal into the sum of multiple sine waves evenly spaced in frequency, each having a magnitude and phase of the sinusoidal component of the signal, changing with time. The time interval separating the start of each analyzed data block is called the "hop size," and has a length (in number of samples) that is a fraction of the length of the data block. The result is the *nonheterodyned* (see 2.3.2) STFT representation of the signal, denoted $X(t_a^u, \Omega_k)$:

$$X(t_a^u, \Omega_k) = \sum_{n=-\infty}^{\infty} h(n) x(t_a^u + n) e^{-j\Omega_k n}$$

where x is the original signal, $h(n)$ is the analysis window, $\Omega_k = \frac{2\pi k}{N}$ is the center frequency of the *k*th vocoder "channel," *N* is the size of the discrete Fourier transform, in number of samples, and $t_a^u = R_a u$, where R_a is the analysis hop factor and u is an integer index to specify the analysis block.

The resynthesis stage involves setting synthesis time instants t_s^u, usually uniformly, so that $t_s^u = R_s u$, where R_s is the synthesis hop factor. At each of these synthesis time-instants, a short-time signal $y_u(n)$ is obtained by inverse Fourier transforming the synthesis STFT $Y(t_s^u, \Omega_k)$. Each short-time signal is then multiplied by an optional synthesis window $w(n)$, and the windowed short-time signals are all summed together, yielding the output signal $y(n)$:

$$y(n) = \sum_{u=-\infty}^{\infty} w(n - t_s^u) y_u(n - t_s^u) \text{ with}$$

$$y_u(n) = \frac{1}{N} \sum_{k=0}^{N-1} Y(t_s^u, \Omega_k) e^{j\Omega_k n},$$

In the absence of modifications (i.e. analysis hop size, $R_a = R_s$,synthesis hop size, and $Y(t_s^u, \Omega_k) = X(t_a^u, \Omega_k)$), this output signal is theoretically identical to the original signal *x*. In general, however, a modified $Y(t_s^u, \Omega_k)$ is not the STFT of any actual signal since the Fourier transforms correspond to overlapping short-time signals. The formula above yields a signal whose STFT is close to $Y(t_s^u, \Omega_k)$ with the deviation dependent on the choice of the synthesis window $w(n)$.

2.3.2 Time-Scale Modification

To modify the time scale of a signal, we modify the magnitude and phase data generated in the analysis stage before resynthesizing. We modify the data in two ways. First, we change the relative spacing between start times of output data blocks by a factor equal to the amount of time scaling (i.e. the analysis hop factor R_a is different from the synthesis hop factor R_s). Second, the phase values of the synthesis STFT $Y(t_s^u, \Omega_k)$ are calculated according to a formula given below.

According to the underlying signal model, the input signal is the sum of a number of sinusoids with time-varying amplitudes $A_i(t)$ and instantaneous frequencies $\omega_i(t)$

$$x(t) = \sum_{i=1}^{I(t)} A_i(t) e^{j\phi_i(t)} \text{ with}$$

$$\phi_i(t) = \phi_i(0) + \int_0^t \omega_i(\tau) d\tau$$

where $\phi_i(t)$ and $\omega_i(t)$ are the instantaneous phase and frequency of the i^{th} sinusoid. Based on the above two equations, for a constant modification factor α such that $t_s^u = \alpha t_a^u$, the ideal synthesis phase $\phi_s(t_s^u)$ of the i^{th} time-scaled sinusoid would be

$$\phi_s(t_s^u) = \phi_s(0) + \int_0^{t_s^u} \omega_i(\tau / \alpha) d\tau$$

$$= \phi_s(0) + \alpha \int_0^{t_a^u} \omega_i(\tau) d\tau$$

$$= \phi_s(0) + \alpha \left[\phi_i(t_a^u) - \phi_i(0) \right]$$

where $\phi_s(0)$ is an arbitrary initial synthesis phase.

Phase-vocoder based time scaling modifies the STFT of the sinusoidal input signal components so as to produce the above time-scaled sinusoids. The time-evolution of the sine wave amplitudes is modified simply by setting $\left|Y(t_s^u,\Omega_k)\right| = \left|X(t_a^u,\Omega_k)\right|$ where $t_s^u = R_s u$. However, modification of the sine-wave phases is a bit trickier.

To calculate the phase of $Y(t_s^u,\Omega_k)$, the standard phase-vocoder technique requires phase unwrapping, a process whereby the phase increment between two consecutive frames is used to estimate the instantaneous frequency of a nearby sinusoid in each channel. The instantaneous frequency $\hat{\omega}_k(t_a^u)$ is estimated by first calculating the *heterodyned* phase increment

$$\Delta\Phi_k^u = \angle X(t_a^u,\Omega_k) - \angle X(t_a^{u-1},\Omega_k) - R_a\Omega_k$$

then taking its principal determination (value between $\pm\pi$) denoted $\Delta_p\Phi_k^u$ and deriving the instantaneous frequency $\hat{\omega}_k(t_a^u)$ of the closest sinusoid using

$$\hat{\omega}_k(t_a^u) = \Omega_k + \frac{1}{R_a}\Delta_p\Phi_k^u$$

This procedure is called *phase unwrapping*, because the actual (nonwrapped) value of the phase increment is calculated from its principal (wrapped) determination. The heterodyned phase increment $\Delta_p\Phi_k^u$ is the small phase shift resulting from $\omega_k(t_a^u)$ being close but not necessarily equal to Ω_k. Once the instantaneous frequency at time t_a^u is estimated, the phase of the time-scaled STFT at time t_s^u is set according to the following *phase-propagation* formula

$$\angle Y(t_s^u,\Omega_k) = \angle Y(t_s^{u-1},\Omega_k) + R_s\hat{\omega}_k(t_a^u).$$

With time-scaling modifications completed, we combine the magnitude and phase components to reconstitute the Fourier coefficients of the synthesis blocks. Lastly, we synthesize the output signal by inverse Fourier transforming each synthesis block and overlapping and adding them together to form the time-domain output.

2.3.3 Performance Considerations of the STFT Based Phase-Vocoder

2.3.3.1 Window Size: Frequency vs. Temporal Resolution

The choice of window size determines the frequency and temporal resolution of the STFT. The frequencies measured in each STFT window will be integer multiples of the reciprocal of the analysis window duration, uniformly spaced apart by sampling rate/N Hertz, ranging from sampling rate/window length up to the Nyquist frequency (sampling rate/2). Window size also determines the temporal resolution achievable in the STFT, the smaller the window, the more precisely the onset of signal events can be resolved. Here we run into the time-frequency uncertainty dilemma: increasing the resolution in time requires decreasing window size, which decreases the frequency resolution and vice-versa. The theoretical limit on temporal vs. frequency resolution is expressed in the uncertainty principle for time-frequency:

$$\Delta t \Delta f \geq \frac{1}{4\pi}$$

As a result, the window size choice will always represent a trade off between giving up frequency resolution with smaller windows, which is most detrimental in lower frequencies due to the logarithmic frequency selectivity of human hearing, or giving up temporal resolution with larger windows, which diminishes the ability to capture high frequency transient attacks [10]. Typical window sizes range between 256 to 4096 samples for standard sampling rates (32kHz, 44.1kHz, 48kHz).

2.3.3.2 Effects of Different Window Types

Another consideration for analysis windows is the effect of the windowing function choice. The purpose of applying a windowing function is to localize each time segment to be analyzed in a STFT frame, and to filter out the high frequency signal content due to the discontinuities at the boundaries of the finite window. Another way to state this in frequency domain filtering theory terms is that windowing is used to reduce side lobes due to truncation, i.e. application of a rectangular window [11]. A bandpass interpretation of the STFT [11] reveals that the windowing function also determines how well each analysis frequency band is isolated from neighboring bands (steepness of transition bands), and the effective width of each band (i.e. side lobe attenuation and main lobe width of the equivalent BPF for the STFT analysis band). The ideal windowing function will have maximally wide main lobe and heavily attenuated side lobes. An analysis of the performance of typical windowing functions (Blackman, Hanning and Hamming) with the phase-vocoder algorithm is presented in [9]. To summarize, the Hanning and Hamming windows provide a spectrum with a thinner main lobe ($\frac{4}{NT}$ where N is the size of the window and T is the sampling period) than the Blackman ($\frac{6}{NT}$). The Blackman function provides the best main lobe to secondary lobe magnitude response ratio (57dB) compared to the Hamming function (41dB) or the Hanning function(31dB). Another consideration in evaluating windowing function performance is how it behaves when overlapping windows (discussed below) are used. In [9], the best overall performance was found with the Hanning window.

2.3.3.3 Window Overlapping

Since windowing functions decay to zero (or near zero) at the ends, it is necessary to overlap windows so the windowed signal is not zeroed out at periodic intervals. In particular, windows are overlapped such that their sum, $\sum_n (t - nS)$ (where S is the relative shift offset for each window) is constant for all *t*. Another reason overlapping windows are required is that it increases the accuracy with which individual frequency components (a.k.a partials) are discriminated. This can be seen by considering that instantaneous frequency is calculated by taking the difference of consecutive frames' phase coefficients for each frequency bin. The next step in identifying the frequency is to measure the deviation from the center frequency of the bin by subtracting out the expected phase change of the bin frequency, a positive result signifying a positive deviation in bin frequency and a negative result signifying a negative deviation. In the simple case that any two neighboring frames do not overlap, the deviation in frequency can range between $\pm\pi$, meaning that it is only possible to discriminate a frequency of half the distance between a given bin frequency and its neighboring bins. Frequency deviations beyond that will wrap back into the $\pm\pi$ interval, and the frequency of the partial being tracked by that bin will be incorrectly estimated. When we apply the *phase-propagation* formula after remapping instantaneous frequency coefficients to a scaled time grid, the error in partial frequency estimates will manifest as audible erroneous frequency components (with no time scale modification, the errors would cancel out upon reconstruction). When the windows are overlapped by a given factor however, the expected phase change is decreased by the inverse of the time-scale factor (e.g. 2x overlap corresponds to a decrease by ½ of the expected phase change between consecutive frames), and consequently the effective frequency deviation range we can detect increases by the same factor. As such, with overlapping windows, the true frequency of a partial can be estimated more accurately in the frequency bins that are detecting its signal energy. A typical overlap factor is 4 (75%), which represents redundant sampling of the STFT and consequently increases the computation cost by increasing the number of FFT calculations required to process a signal by the overlap factor.

2.3.3.4 STFT Phase-Vocoder Artifact Causes

There are a number of sources of the audible artifacts in signals time-stretched via the STFT phase-vocoder. Firstly, some artifacts are due to the use of a single fixed size window used for analyzing the entire spectrum; we are forced to compromise temporal resolution for frequency resolution. When we give up temporal resolution, transients are smeared in time after time-stretching, which is perceived as short bursts of narrow-band high frequency noise in the output.

Similarly, low resolution in frequency manifests as blurring together of neighboring frequency components into multiple unstable pitches in the frequency bands occupied by the original components. When the degree of window overlap is insufficient, erroneous frequency components are spawned from existing partials due to frequency estimation errors as described in the preceding section. In addition to these artifacts sources, there is also a set of artifacts caused by loss of vertical phase coherence [2]. Vertical phase coherence refers to the lining up of the phases of signal components at different frequencies at each time instant. By analyzing and reconstructing frequency components within each frequency channel, as the phase vocoder time-scaling algorithm does, only "horizontal" phase coherence (within each channel, along the time dimension) is maintained, but no effort is made to maintain vertical phase coherence, across frequency channels. When *phase-propagation* errors accumulate in individual frequency channels after time-scale modification, dispersion of the original phase relationships among different frequency components results. This loss of phase coherence is a major source of more subtle artifacts in the phase-vocoder. The perceived effect of vertical phase coherence loss is a slight reverberation effect, dubbed "*phasiness*," along with *loss of presence* (the sound source sounds as if it has moved further away) [2]. The reason it is perceived this way is that dispersion in time of originally time aligned phase relationships across frequency is similar to what occurs when sound bounces off different surfaces with different sound absorption characteristics in an acoustic space. Lastly, when noisy signals are time stretch by large factors, there are a noticeable *glissing* artifacts (one or more pitched signals gliding up and down unpredictably). The *glissing* is due to a combination of low frequency resolution and loss of vertical phase coherence. Low frequency resolution leads to discetization of the randomly distributed energy of the noise into widely spaced separate bins, each of which represent the energy as individual frequency components. When phase-propagation errors accumulate and vertical phase coherence is lost for the signal energy in neighboring bins due to noise, the signals in individual bins emerge as discrete frequency components after resynthesis.

2.3.3.5 Phase-Vocoder Improvements

A partial solution to the loss of vertical phase problem has been suggested in [2], called *scaled-phase-locking*. The *scaled-phase-locking* scheme starts by identifying spectral peaks in a STFT analysis frame, and tracking the movement of each peak across consecutive analysis frames. The *phase unwrapping* calculation for a given peak is done by subtracting the phase of the previous frequency bin containing a given peak from the phase of the current frequency bin containing the peak, which may be a different bin. The difference in the unwrapped phase of a peak bin and its nearest k neighboring bins' unwrapped phase is stored for each of the k neighboring bins. After remapping the coefficients of each frame to the time-scaled grid, only the peak frequency channel bins' phases are calculated with the *phase-propagation* formula. The neighboring k bins' synthesis phase coefficients are set so that they have the same phase difference with respect to their nearest peak bin. This technique preserves vertical phase coherence in the neighborhood of each spectral peak. The use of *scaled-phase-locking* has been shown to improve subjective performance of the phase-vocoder, and enables a reduction of the required overlap between analysis frames to 50% for good performance [2].

2.3.3.6 Overview of STFT Phase-Vocoder Shortcomings

There are a number of drawbacks in using the STFT for analyzing audio signals [3]. Firstly, the analysis produces equally spaced frequency bands, which does not correspond to the logarithmic frequency selectivity of human hearing. To get acceptable frequency resolution in low frequencies, with respect to human hearing, long analysis windows must be used, which results in loss of temporal resolution and excessive resolution in high frequencies. Therefore, the STFT is quite inefficient for this purpose. Secondly, when window filtering is applied to the data to reduce errors, the STFT is even more inefficient, as techniques such as overlap-and-add need to be used. Lastly, the compromise of time and frequency resolution that must be accepted due to the uncertainty principle is the same over all frequencies, and cannot be adjusted for different frequency ranges. These shortcomings of the STFT manifest themselves in phase-vocoder time-scaled signals most noticeably as smearing of transients, "*phasiness*," (slight reverberation) along with *loss of presence*. Wavelet transform analysis overcomes the fixed resolution limitation of STFT analysis, and has been investigated as a replacement for the STFT in the phase-vocoder algorithm. In the remainder of this paper, we explore the efficacy of wavelet transform based phase-vocoders.

3.0 Wavelet Transform Based Phase-Vocoder

3.1 Wavelet Transform Analysis for Audio Signals

The wavelet transform was developed as an alternative to the STFT to overcome the problems inherent in its time and frequency resolution properties. Rather than providing uniform resolution in frequency and time domains, the wavelet transform provides high frequency resolution and low time resolution in the low frequency range and high time resolution and low frequency resolution in the high frequency range. In signal processing terms, the wavelet transform can be viewed as a constant Q filterbank with octave spacing between filter center frequencies. The wavelet transform deals with the uniform time-frequency resolution limitation of fixed-size windowing that burdens the STFT by using variable sized windows for different frequency bands [1]. The wavelet transform uses analysis windows that dilate according to the frequency being analyzed, with long time windows where more precise low frequency information is desired, and shorter windows where the high frequency information is desired. A wavelet transform whose frequency scales change by powers of two (dyadically partitioning), such as the fully decimated Discrete Wavelet Transform (DWT), produces an octave band decomposition of a signal. Such a partitioning of the frequency spectrum closely mirrors the logarithmically spaced perception of frequency of the human auditory system and as such makes the wavelet transform an ideal candidate for use in analyzing audio signals.

There are a number of special considerations for audio signals that must be taken into account when selecting the type of wavelet analysis to use [3]. First, we note that when the applied wavelet basis function does not resemble the shape of the analyzed signal, the wavelet coefficients will not extract the main "features" of the signal. The classes of audio signal of primary interest, speech and musical signals, are typically smooth waveforms, which implies that the best wavelet basis function is one that is sufficiently smooth, i.e., high regularity is preferred. Second, the size of the transition_band of low pass and high pass filters is an important factor. Larger transition bands (i.e. low steepness), cause considerable overlapping of low pass and high pass bands. So the output bands of the filter bank are not well separated, and aliasing effects are reenforced when the coefficients are changed (as is required for time-scale modification). Furthermore, linear phase response is crucial for high quality audio filters. When the filters do not have at least an approximate linear phase, different frequencies are delayed by different amounts, which can cause irreversible loss of the original phase relationships of signal components across frequency scales if any modifications are made to coefficients in the wavelet domain.

3.2 Continuous Wavelet Transform (CWT)

The Continuous Wavelet Transform is defined as:

$$W_f(a,b) = \int_{-\infty}^{\infty} f(t)\psi_{a,b}^{*}(t)dt$$

where, * denotes the complex conjugate and $\psi_{a,b}(t)$ is the mother wavelet, scaled by a factor a and dilated by a factor b:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right)$$

In practice, the CWT must be approximated with a discretized version. The requisite discretizations involve a discrete approximation of the transform integral (i.e. a summation) computed on a discrete (but not necessarily dyadic) grid of a scales and b locations. A major disadvantage of the CWT compared to the Discrete Wavelet Transform is the much higher computational cost. In addition, the two most common CWT basis functions for analyzing audio signals in the literature, the Morlet and Mexican Hat wavelets, have neither compact support, vanishing moments, nor orthogonality. As such, invertible filters cannot be calculated, making practical implementation of the inverse CWT problematic (which explains the absence of an inverse CWT function in wavelet software packages such as the Matlab Wavelets Toolbox). For these reasons, the use of the CWT is pursued no further here.

3.3 Discrete Wavelet Transform (DWT), and Its Limitations

The Discrete Wavelet Transform (DWT) maps a continuous function $f(t)$ to a series of coefficients $b_{j,k}$, and is defined as:

$$b_{j,k} = \int_{-\infty}^{\infty} f(t)\psi_{j,k}(t)dt$$

$$\text{where } \psi_{j,k} = 2^{j/2}\psi\left(2^j t - k\right)$$

or, representing the integration as a summation for discrete-time signals,

$$W(j,k) = \sum_j \sum_k x(k) 2^{-j/2} \psi(2^{-j} n - k)$$

where, $\psi(t)$ is the "mother" wavelet function. The decomposition described in this last equation can be implemented by successive highpass and lowpass filtering of the time-domain signal using the following equations:

$$y_{high}[k] = \sum_n x[n] g[2k - n]$$

$$y_{low}[k] = \sum_n x[n] h[2k - n]$$

where $y_{high}[k]$ and $y_{low}[k]$ are the outputs of the highpass (g) and the lowpass (h) filters respectively after subsampling (decimating) by two. The equations are repeatedly applied to the $y_{low}[k]$ output of each stage, which results in decomposition of the signal into octave subbands that can be viewed as a coarse approximation (lowpass part) of the input, and detail information (highpass part). Each subband contains half the samples of the next higher subband, and the total number of DWT coefficients is the same as the number of input samples.

The standard DWT is powerful tool for signal analysis, but it has two major disadvantages for its use in processing audio signals in general, and in the phase-vocoder algorithm in particular. The first disadvantage is the lack of shift invariance of the DWT. This means that small time shifts in the input signal can cause major variations in the distribution of energy between DWT coefficients at different scales [5]. This represents a problem for a DWT based phase-vocoder time-scaling algorithm because the coefficients are remapped to a scaled time grid, which is equivalent to shifting the input signal by varying amounts over time. The distortions resulting from the lack of shift invariance will produce the types of audio artifacts seen in the STFT based phase-vocoder, namely the “phasiness” and reverberation effects (since those artifacts are caused by such erroneous dispersion of signal energy across STFT frequency bins).

The second disadvantage is the absence of phase information due to the fact that the DWT produces only real coefficients for real input signals, which includes audio signals. Without phase information, instantaneous frequency cannot be calculated, which is required in the phase-vocoder algorithm.

Although the prospects at this point for using the wavelet transform in the phase-vocoder algorithm may appear bleak, all hope is not yet lost. A new variant of the DWT that does not suffer from the above mentioned shortcomings is available, namely, the Dual-Tree Complex Wavelet Transform.

3.4 The Dual-Tree Complex Wavelet Transform (DT-CWT)

3.4.1 DT-CWT Properties for Phase-Vocoder Time-Scaling

The Dual-Tree Complex Wavelet Transform (DT-CWT), first and foremost, makes available the phase information required by the phase-vocoder algorithm since it produces complex valued coefficients. Secondly, the DT-CWT offers approximate shift-invariance, which means that aliasing effects due to the decimations in the transform are minimized, and interpolation of coefficients within each sub-band to any desired sampling grid is feasible. This implies good immunity to the smearing of signal energy across frequency scales after time-scaling modifications are applied to transform coefficients. This energy smearing is one of the primary sources of undesirable audio artifacts, as mentioned in section 2.3.4. In addition to these beneficial characteristics, there are forms of the DT-CWT (the Kingsbury odd-even DT-CWT) that offer other desirable characteristics, such as linear-phase filters with steep transition bands (desirable for audio signals, as noted earlier), perfect reconstruction (PR), and efficient order-N computation (specifically, 2N, which is true in general for DT-CWT's) [6]. The order-N computation is possible due to the use of the fast-DWT (Mallat's algorithm), a highly efficient implementation of the DWT.

3.4.2 Kingsbury's Dual-Tree Complex Wavelet Transform (DT-CWT(K))

The DT-CWT maps a continuous, real (or complex) input signal to a series of complex coefficients. It accomplishes this by with two parallel DWT trees' that have a quadrature phase offset relationship to one another. The outputs of either DWT tree thus represents real and imaginary components of complex coefficients of a wavelet decomposition of a signal.

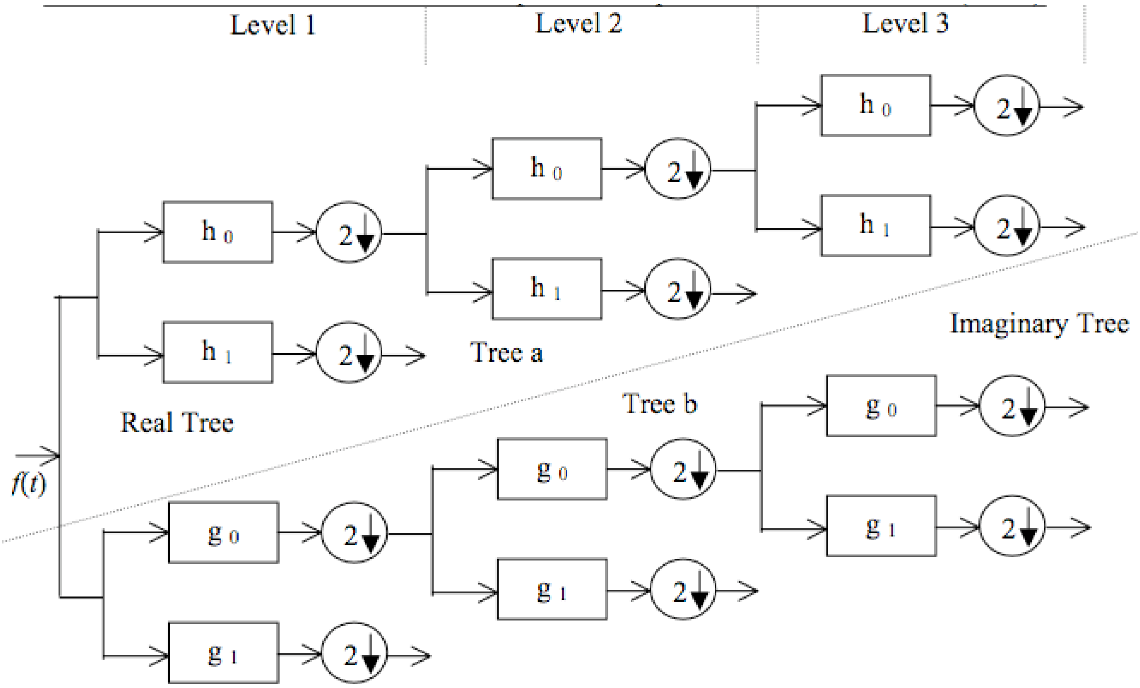


Figure 1. Analysis filterbank structure of the DT-CWT

3.4.2.1 DT-CWT(K) Filter Design for Shift Invariance and PR

Kingsbury observed that approximate shift invariance is possible with standard DWT by doubling the sampling rate at each level of the tree [6]. For this to work, the samples must be evenly spaced and the down-samplers must be eliminated after the level-1 filters. This is equivalent to having two parallel fully decimated trees, real (tree-a) and imaginary (tree-b) (as in figure 1), if the delays of first level filters of tree-b are one sample offset by their corresponding filter in tree-a. This offset ensures the pickup of opposite samples in both trees. To get uniform intervals between the samples of both trees after level-1, filters in one tree must provide delays that are half a sample different from those in opposite tree.

One way in which the DT-CWT(K) can be designed to have the required delays is to use odd and even length filters in the corresponding levels of the parallel filter trees. The design of DT-CWT(K) is based on viewing the scaling coefficients from tree-b as interpolating mid-way between the corresponding coefficients from tree-a. To have linear phase filters with required delay offset between filters beyond level-1 of both trees, it is necessary to have odd-length filters in one tree and even-length filters in the other tree. Note, the filters at level-1 in both trees are odd length, and the corresponding lowpass and highpass filter pairs are apart by one sample.

The PR condition for the DT-CWT(K) model using standard multi-rate identities is given in [7] as:

$$\tilde{F}(Z) = \tilde{F}_a(Z) + \tilde{F}_b(Z) = F(Z)$$

$$= \frac{1}{M}\sum_{k=0}^{M-1} X(W^k Z)\left[A(W^k Z)C(Z) + B(W^k Z)D(Z)\right]$$

where, $M = 2^m$ is the total up/down sampling factor determined by the number of levels (m) of the dual-tree, the factor $W = e^{\frac{j2\pi}{M}}$, letters A, B represent transfer functions of the analysis dual-tree and C, D represent the corresponding transfer functions for the synthesis dual-tree.

The filters in the synthesis tree may be biorthogonal for PR or near-orthogonal for energy preservation (in transform domain) of the filters of analysis tree. The odd-even filter design is based on minimum mean squared error in the approximation [6]. We note that in practice, filters with compact support will not have zero gain in their stop bands and the aliasing terms in the PR equation will not be zero. Furthermore, odd-length filters cannot have precisely the same frequency responses as the even-length ones. So a typical DT-CWT(K) will only be ‘approximately’ shift-invariant.

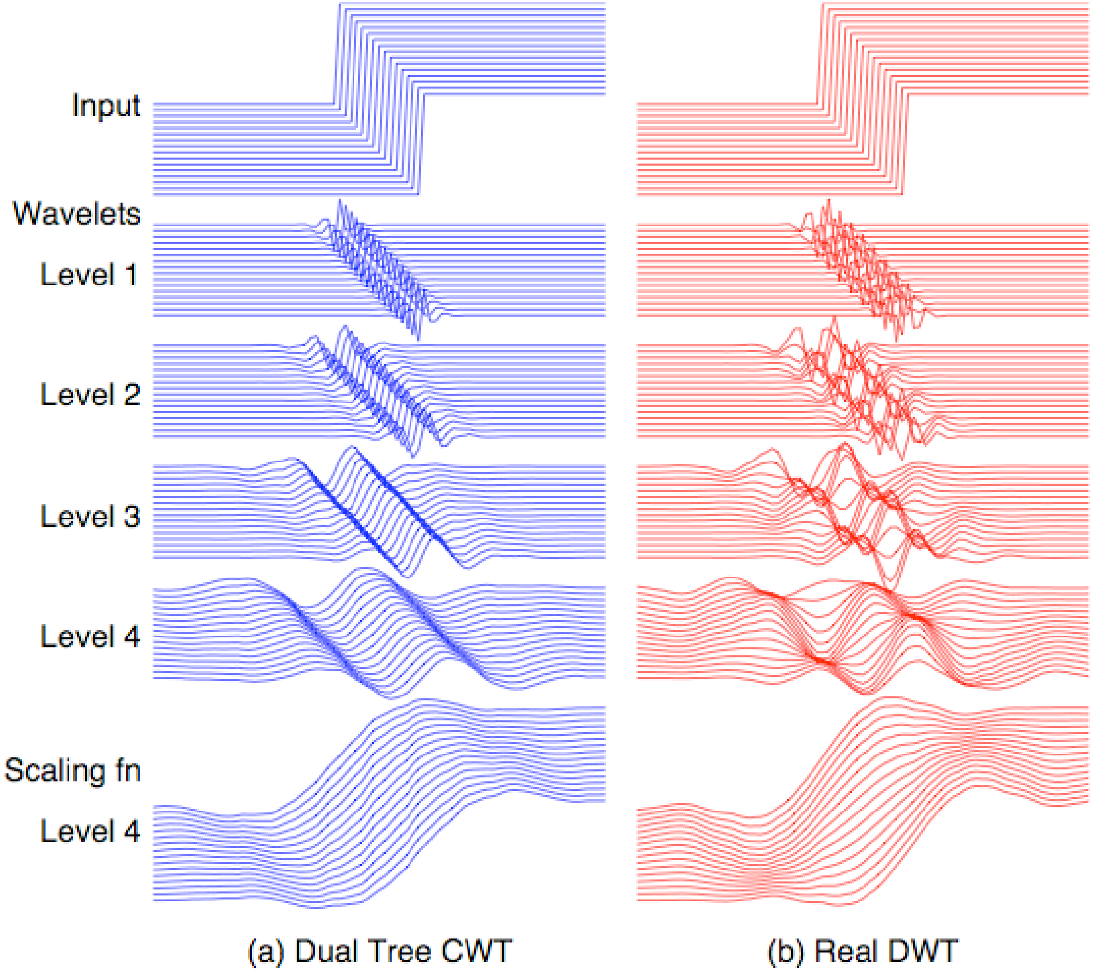


Figure 2. Illustration of shift-invariance of the DT-CWT:
Wavelet and scaling function components at scale 1 to 4 of 16 shifted step responses for (a) DT-CWT and (b) real DWT

3.5 Using the DT-CWT in the Phase-VocoderAlgorithm

We now repeat the phase-vocoder time-scaling derivation, modified to fit the dyadic time-frequency grid of the DT-CWT. For convenience, we express the complex DT-CWT coefficients in a form that resembles STFT coefficients so we can see more easily the similarities between the derivations. We express the DT-CWT coefficients for the input signal $x(t)$ as,

$$DT\text{–}CWT\{x(t)\} = X(t^{j,k}, \xi_j)$$

where $t^{j,k} = (2^j T_s)k$ (T_s is the sampling period for the sampled input signal, $x(n) = x(nT_s)$, k is the DT-CWT coefficient (integer) translation index, and ξ_j is the pseudo-frequency, in radians, of scale j. The pseudo-frequency is the center frequency of the sub-band filter corresponding to the jth scale, which can be estimated as the frequency midway between the –3dB points of the filter.

The coefficient magnitudes are time-scaled by mapping the output coefficient magnitudes to the input coefficient magnitudes using:

$$\left|Y(t_s^{j,k}, \xi_j)\right| = \left|X(t_a^{j,k}, \xi_j)\right| \qquad (3.1)$$

where $t_s^{j,k} = \alpha t_a^{j,k}$ (α is the time-scaling factor). Next we do *phase unwrapping*, i.e. calculate the phase increment between two consecutive analysis windows to estimate the instantaneous frequency of a nearby sinusoid (referring to the underlying signal model) in each pseudo-frequency channel (scale). The phase increment $\Delta\Phi_{j,k}$ between analysis window k and k-1 for each scale j is:

$$\Delta\Phi_{j,k} = \angle X(t_a^{j,k}, \xi_j) - \angle X(t_a^{j,k-1}, \xi_j) - \xi_j \qquad (3.2)$$

from which we take the principal determination (value between $\pm\pi$) $\Delta_p\Phi_{j,k}$ and derive the instantaneous frequency $\hat{\omega}_j(t_a^{j,k})$ of the closest sinusoid using

$$\hat{\omega}_j(t_a^{j,k}) = \xi_j + \Delta_p\Phi_{j,k} \qquad (3.3)$$

The phase of the time-scaled DT-CWT at time $t_s^{j,k} = \alpha t_a^{j,k}$ is found using the *phase propagation* formula (from section 2.3.2)

$$\angle Y(t_s^{j,k}, \xi_j) = \angle Y(t_s^{j,k-1}, \xi_j) + \alpha\hat{\omega}_k(t_a^{j,k}) \qquad (3.4)$$

In practical terms, this algorithm can be implemented in the following steps:

1. Starting with the DT-CWT of input signal $x(t)$, convert the complex coefficients to magnitude and phase form: $\left|X(t_a^{j,k},\xi_j)\right|$, $\angle X(t_a^{j,k},\xi_j)$.
2. Calculate the instantaneous frequencies across each scale using (3.2) and (3.3).
3. For each scale, map magnitudes and instantaneous frequencies from the scaled time axis for the input to the unscaled time axis for the output. Use (3.1) to map the input to the output magnitudes, and use:

   $$\hat{\omega}_j^{(Y)}(t_s^{j,k},\xi_j) = \hat{\omega}_j^{(X)}(t_a^{j,k},\xi_j) \qquad (3.5)$$

   to map the input instantaneous frequencies $\hat{\omega}^{(X)}$ to output instantaneous frequencies $\hat{\omega}^{(Y)}$. (Note, no actual calculation is required for this step, it is only necessary to define the mapping for the next step which will carry out the mapping implicitly)
4. Using the mapping defined in step 3 as an underlying function, we interpolate magnitudes and instantaneous frequency values across each scale at points coinciding with the DT-CWT time grid. This can be done using the `interp1` function in Matlab (using piecewise cubic spline interpolation):

   ```
   w_ouptut = interp1(t_input,w_input,t_output,'spline');
   ```

   where w_input = $\hat{\omega}_j^{(Y)}(t_s^{j,k},\xi_j)$ the remapped instantaneous frequencies for the output, and t_input = the remapped time points : $t_s^{j,k} = \alpha t_a^{j,k}$,

   ($k = 0,1,\ldots N-1$) for

input signal of length N) and t_output = the time points of the output time grid: $t_a^{j,k}$, $(k = 0,1,\ldots\lfloor \alpha^{-1}(N-1)\rfloor)$.

We likewise interpolate magnitude values across each scale.

```
Ymagn_output = interp1(t_input,Ymagn_input,t_output,'spline');
```

5. Convert the instantaneous frequency values across each scale to phase values by applying the phase propagation formula (3.4) to the interpolated instantaneous frequencies produced in step 4.

$$\angle Y(t_a^{j,k},\xi_j) = \angle Y(t_a^{j,k-1},\xi_j) + \hat{\omega}_j^{(Y)}(t_a^{j,k})$$

6. Convert the magnitude and phases to complex coefficients.
7. Apply the inverse DT-CWT to the complex coefficients (at the same scales used for the input DT-CWT) to generate the time-scaled output signal $y(t)$.

4.0 DT-CWT Phase-Vocoder Implementation

An implementation of the DT-CWT phase-vocoder algorithm was created in Matlab using the DT-CWT Pack (version 4.3) provided by Dr. Nick Kingsbury, to perform the forward and inverse DT-CWT's. Matlab code and example scripts for demonstrating the algorithm can be downloaded from http://www.ece.utexas.edu/~jlivings.

4.1 Input Signal Considerations for DWT Analysis

The dyadic partitioning inherent in the DWT implies that signal length should be a power of two so the signal can be fully partitioned, allowing the maximum possible number of wavelet scales to be used in the decomposition of the signal into frequency bands. Zero padding can be used to extend the length of the input to the nearest power of two. The number of wavelet scales is dependent on the signal size. The highest frequency scale corresponds to the sampling frequency/2, and subsequent frequency scales are centered at one half the next highest scale. As such the maximum number of scales is log2(N), where N is the length of the signal, and the lowest frequency band analyzed is centered at $\frac{sampling\_rate}{2^{\log_2 N}}$. The analysis window length should thus be chosen so that the lowest frequency band is at least as low as the lowest perceivable frequency, which is in the neighborhood of 10Hz.

4.2 DT-CWT Calculation

The parameters for the DT-CWT function are the filter functions used for the first decomposition level and the filter functions used for all subsequent decomposition levels. The criteria for choosing the filters are the degree of shift invariance and the amount of aliasing energy. A useful measure of the aliasing energy is the ratio of the unwanted aliasing energy to the desired non-alias energy as formulated in [6].

$$R_a = \frac{\sum_{k=1}^{M-1} E\{A(W^k z)C(z) + B(W^k z)D(z)\}}{E\{A(z)C(z) + B(z)D(z)\}}$$

where letters A, B represent transfer functions of the analysis dual-tree and C, D represent the corresponding transfer functions for the synthesis dual-tree and $W = e^{\frac{j2\pi}{M}}$. $E\{U(z)\}$ calculates the energy, $\sum_r |u_r|^2$ of the impulse response of the z-transfer function $U(z) = \sum_r u_r^{\,2} z^{-r}$. Kingsbury used this measure to compare a number of filters designed for use as basis functions in the DT-CWT in [6]. Of the available filters for use in the DT-CWT function in the Matlab toolkit, the filter set with the best combination of shift-invariance and lowest aliasing to non-alias energy ratio was the biorthagonal, near-symmetric 13, 19 tap filters for the level 1 decomposition and quarter-sample shift 18,18 tap filters (‘qshift_d’) for levels >1. All filters in the set have linear phase, which as noted in section 3.1 is an advantageous property for audio signal filtering.

4.3 Analysis, Time-Scale Modification and Resynthesis Stages

The phase-vocoder algorithm steps: estimation of instantaneous frequencies across scales, remapping of magnitude and instantaneous frequency coefficients to the scaled time grid , conversion of instantaneous frequencies back to phase coefficients, and inverse DT-CWT is as described in section 3.5. The center frequency used for each wavelet scale was midpoint of the octave band represented by the scale, and the initial phase for the derivative of phase calculation was zero.

4.4 Handling Boundary Effects Between Successive Transform Windows

The truncation of the input signal at the beginning and end needs to be handled in such a way as to minimize the distorting effects of the aliasing frequencies caused by the discontinuities at the signal boundaries. There are a few possible approaches for alleviating boundary effects: *zero-padding*, *periodic extension*, and *symmetric extension* [10]. *Zero-padding* makes the assumption that zeros exist beyond the ends of the signal when doing the DWT filtering operations at the signal edges. *Periodic extension* wraps around to the other end of the signal when filtering runs past the signal edges. *Symmetric extension* mirrors the signal about the end points. An in-depth analysis of the properties of each of these signal extension strategies is presented by Strang and Nguyen [7], and they concluded that *symmetric extension* performed best from mean squared error and peak signal-to-noise ratio perspectives. Symmetric extension is the strategy used for handling boundary effects in Kingsbury's DT-CWT Matlab toolkit.

5.0 Evaluation Of Algorithm Implementation

To test the performance of the implementation, three different types of signals, speech, music, and a pure sine tone, were time-stretched by different time scale factors. The desired time-stretching was achieved for all of the test signals, however significant artifacts were also present in the time-stretched signals. Signals processed with a time scaling factor of one (no time-stretching) were reconstructed without any audible change to the signal. The time-stretched signals had erroneous extra frequency components generated along side the existing components. For speech the results sounds clear and intelligible, but with a prominent extra signal component sounding approximately one octave below the original pitch. For the music signal, the vocal part was affected the same way the speech signal was affected, but the polyphonic instrumental parts have additional frequency aliases at irregular intervals (other than just octave intervals). The spectrogram of a 500 Hz signal time-stretched by a factor of two (figure 3) shows that the original signal has been stretched to twice its original length without changing its frequency, and shows the additional signal aliases at different frequencies.

5.1 Discussion of Possible Sources of Artifacts

The filters used in the DT-CWT, as with all DWTs, are designed to cancel alias terms after resynthesis for perfect reconstruction of the signal. Perfect reconstruction however naturally assumes that the wavelet coefficients will not be modified in the transform domain, and modifications will likely lead to aliasing. The aliasing comes from signal energy belonging to one subband leaking into another subband due to the non-brickwall nature of the lowpass and highpass filters used. This is certainly one of the sources of the aliased signals present in the time-stretched output. The near shift-invariance and low aliasing energy properties of the DT-CWT however should greatly alleviate such aliasing after time stretching modifications are performed on transform coefficients, as explained in the earlier sections.

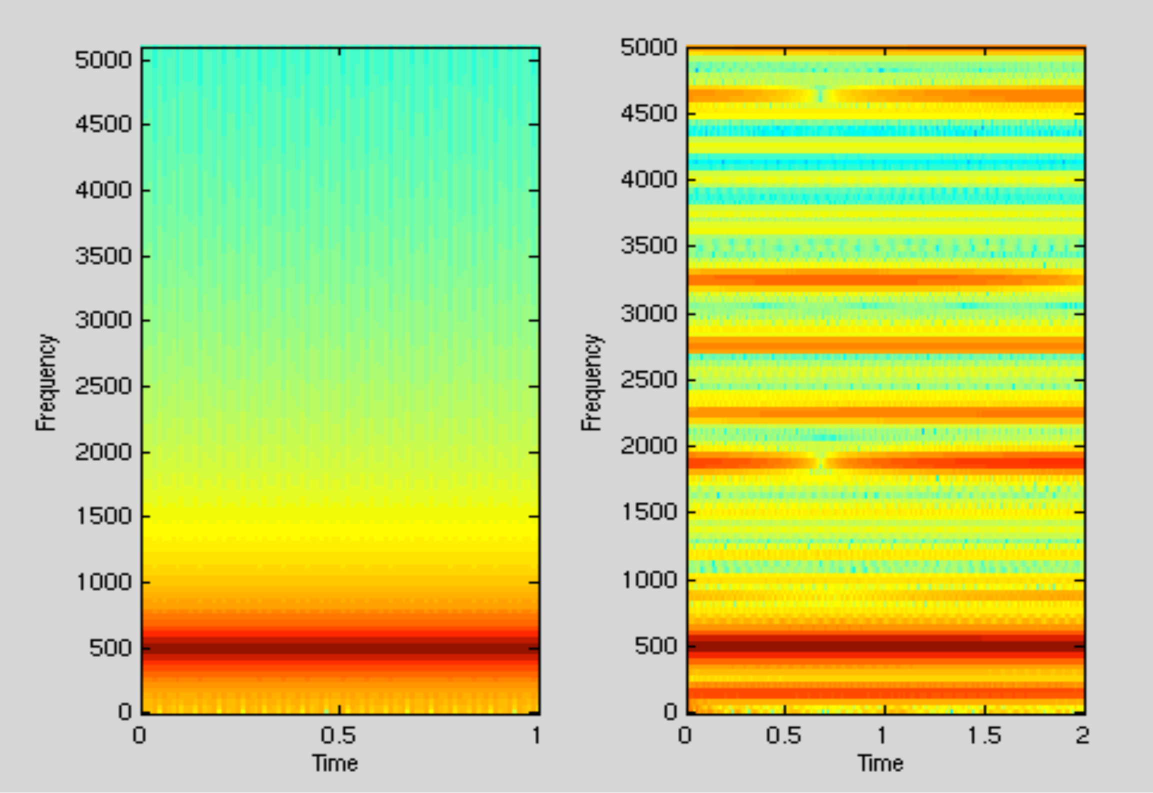


Figure 3. Spectrogram of 1 second 500Hz sinewave (left). Spectrogram of 500Hz sine wave time-stretched by a factor of 2 by the DT-CWT phase-vocoder (right).

Comparison with the output of the STFT phase-vocoder provides a clue to the main sources of the erroneous extra frequency components. If we look at the output of a similarly time-stretched sine wave signal from the STFT phase-vocoder with an excessively small window size, we see a very similar set of erroneous frequency components (figure 4).

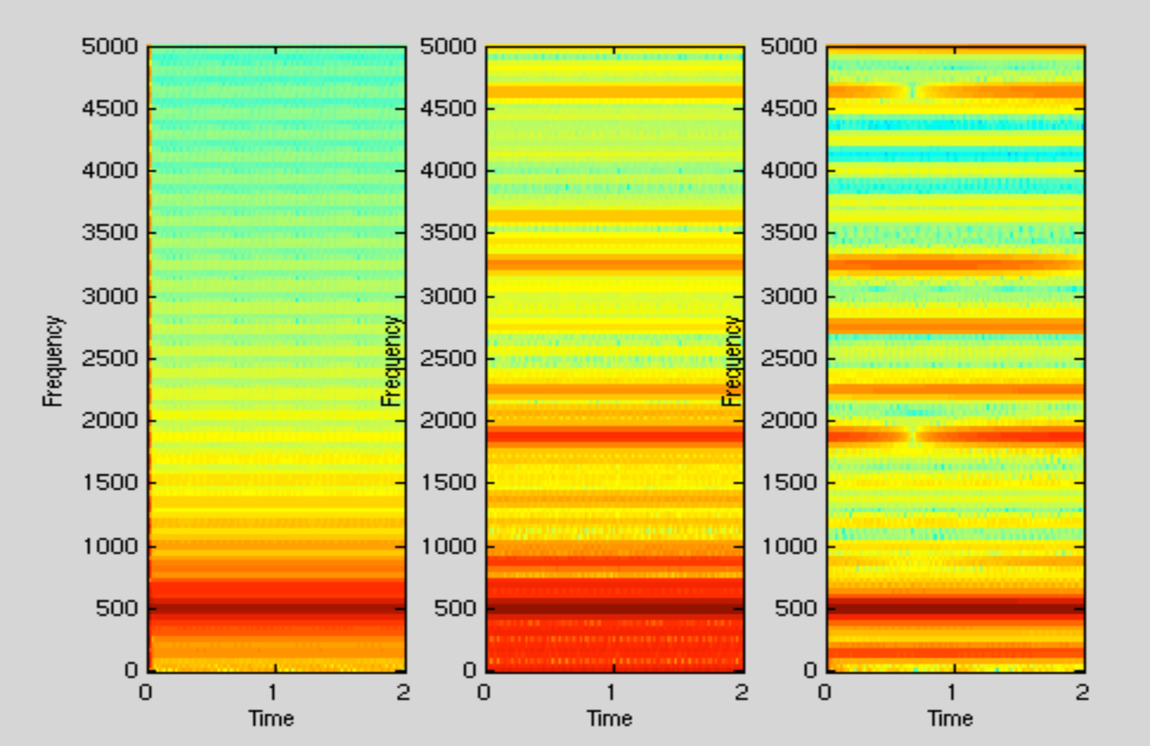


Figure 4. Spectrograms of a 500Hz sine wave (44.1 kHz sampling rate) time-stretched by a factor of two by the STFT phase-vocoder with 1024 sample window, 4x overlap factor (left), STFT phase-vocoder with 128 sample window, 4x overlap factor (middle), and DT-CWT phase-vocoder (right).

As can be seen in figure 4, a similar set of extra frequency components appears in the STFT phase-vocoder output with a window size of 128. If the window size is increased to 1024, these extra frequency components do not appear. The reason for the extra frequency components is inadequate frequency resolution due to the small window size. The excessively wide frequency bands of the 128 sample analysis window allows a significant amount of signal energy of the 500Hz signal to leak into analysis bins other than the bin that contains the actual signal. When the phase unwrapping calculation is done there is energy from the 500Hz signal in analysis bins that are too far away in frequency to accurately estimate the instantaneous frequency, and the signal is erroneously estimated to be within the subband of these analysis bins (see Section 2.3.3.3). The similarity in the pattern of these errors to those found in the DT-CWT time stretched signal strongly implies that the extra signal components in the latter are also due to errors in estimation of instantaneous frequencies due to inadequate frequency resolution. The octave bandwidth of the fully decimated DT-CWT is apparently too wide to isolate and accurately estimate the instantaneous frequency of individual partials in audio signals; the shift-invariance and low aliasing energy properties of DT-CWT cannot overcome the bandwidth limitations of the fully decimated, dyadic (divide by two) frequency partitioning.

5.2 Discussion of Possible Remedies of Artifacts

To overcome the limited frequency resolution of the dyadic DT-CWT, we could instead use a wavelet packet transform (WPT) version of the DT-CWT. The WPT differs from the DWT in that the WPT may decompose one or both the LPF and HPF branches of each node of the filter-bank decomposition tree rather than just the LPF branch (see figure 1). By splitting the HPF branches as well as the LPF branches, the signal can be split into analysis subbands with bandwidths that are less than an octave by factors of 0.5. A logical choice for the bandwidth is 1/3rd of an octave, which corresponds to the critical bandwidths that represent frequency selectivity of human hearing. Such a subband decomposition (as suggested in [10]) seeks to split each octave into three bands that are equally spaced on a logarithmic frequency scale. The 1/3rd octave logarithmic partitioning can be approximated by splitting an octave band into two halves, then splitting the lower half octave into quarter octaves in the decomposition tree of the WPT. This splits the relative bandwidths in the octave into partitions that are 25%, 25% and 50% of an octave (from lower to higher frequency). The true 1/3 octave logarithmic partitioning splits an octave into 26%, 33%, and 41% relative bandwidth partitions. The 25%, 25%, 50% partitioning, referred to as the *pseudo third-octave basis* (PTOB) in [10] lies above and approximately parallel to the critical band curve (figure 5) and should suffice as an approximation to true 3rd octave spacing for time-scale modification. The narrower bandwidth of the PTOB decomposition would greatly reduce the phase unwrapping errors of the dyadic DT-CWT, and thus provide a great reduction in the erroneous frequency component artifacts.

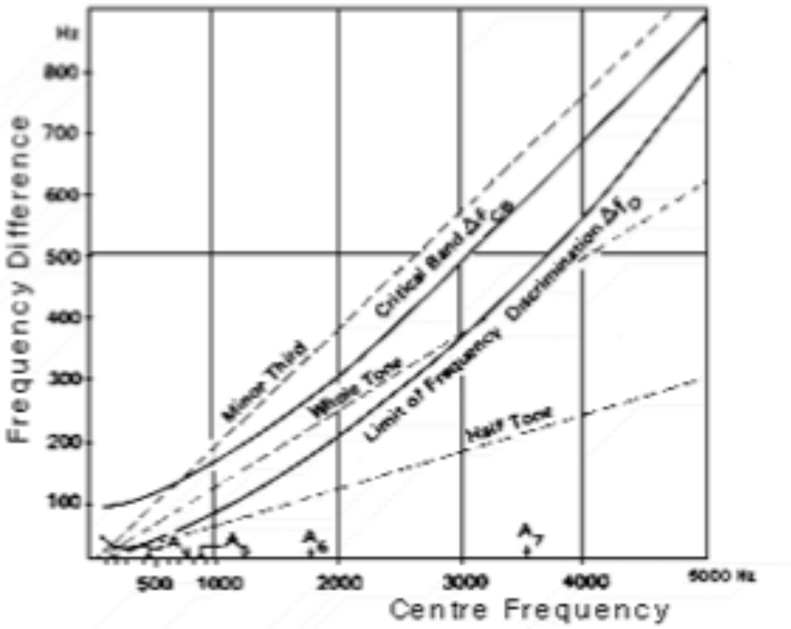
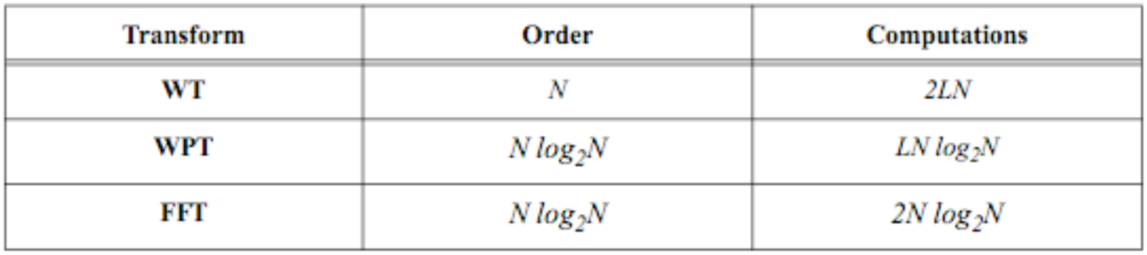


Figure 5. Critical Bandwidth and Musical Intervals versus Frequency, from [15].
The Minor Third (quarter octave) curve corresponds to 25%, 25%, 50% octave partitioning.

5.3 Complexity Comparisons of Phase-Vocoder Implementations

To compare the complexity of the different versions of the phase-vocoders, we will consider the relative complexity of the transforms used, ignoring the phase unwrapping (instantaneous frequency estimation) and phase-propagation stages which are common to all the implementations. The fast wavelet transform, FWT, used in the fully decimated DT-CWT has $O(N)$ complexity, specifically 2LN computations are required, where L is the length of the filters used and N is the length of the window. The STFT phase-vocoder has fundamental $O(N \log_2 N)$ complexity, based on the underlying FFT calculation. There is additional overhead of N multiplications due to the application of the windowing function as well as an additional overall increase by a factor (typically 2 or 4) due to the overlap factor used. A WPT based DT-CWT has the same $O(N \log_2 N)$ complexity as the STFT, with $LN \log_2 N$ actual computations required. The complexity of each of the underlying transforms is summarized in table 1.



Transform	Order	Computations
WT	N	$2LN$
WPT	$N \log_2 N$	$LN \log_2 N$
FFT	$N \log_2 N$	$2N \log_2 N$

Table 1. Complexity of transforms used in each of the phase-vocoder implementations. N refers to the transform window size, L refers to filter length.

From table 1 we see that the WPT based phase-vocoder will have the same general level of complexity as the STFT based phase-vocoder.

6.0 Conclusion

The use of the wavelet transform in place of the STFT in the phase-vocoder algorithm for time-scaling audio signals has been investigated in the past, motivated by the fact that the wavelet transform offers variable time-frequency resolution that can efficiently and precisely capture audio signal information in a manner very well-matched to human auditory perception characteristics. Despite this, little has emerged in the audio processing literature likely due to inherent limitations of traditional forms of the wavelet transform, such as lack of phase information from the DWT, and high computational cost and lack of available inverse transform implementations for the CWT. In this paper, a new wavelet transform based phase-vocoder algorithm was presented that uses a new form of the DWT, the Dual-Tree Complex Wavelet Transform, DT-CWT, that overcomes many of the problems of the older DWT forms.

A preliminary implementation of the algorithm in Matlab resulted in output that was time-stretched as desired, but with the addition of erroneous frequency components due to instantaneous frequency estimation errors cause by the insufficiently narrow octave band frequency resolution of the fully decimated DT-CWT. Use of the wavelet packet transform, (WPT) with approximately 1/3rd octave logarithmically spaced subbands instead of the octave band, fully decimated DT-CWT was proposed as a solution to remedy the artifacts resulting from inadequate frequency resolution.

7.0 Future Work

The DT-CWT based phase-vocoder algorithm presented is based on a fully decimated DWT, which decomposes the audio spectrum into octave bands. As was determined from experiments with the algorithm implementation, the octave bandwidth of the fully decimated DT-CWT was not narrow enough to avoid frequency estimation errors leading to erroneous frequency component artifacts in time-stretched outputs. To provide sufficiently narrow bandwidths for accurate frequency estimation and to better match the 1/3rd octave critical bands of the human auditory system, a WPT implementation of the DT-CWT is required. Unfortunately the DT-CWT Pack toolkit for Matlab only offers a fully decimated implementation of the DT-CWT, and it is outside the original scope and not feasible within the time constraints of this project to develop an implementation of a WPT version of the DT-CWT. As such, the use of WPT will have to be left for future work. In addition to the use of the WPT, other improvements to the DT-CWT phase-vocoder may be possible based on the phase-locking techniques to preserve vertical phase coherence discussed in section 2.3.3.5, adapted to work with the DT-CWT.

BIBLIOGRAPHY

[1] Sanjaume, Jordi Bonada. *Audio Time-Scale Modification in the Context of Professional Audio Post-production*. Informàtica i Comunicació digital, Universitat Pompeu Fabra, Barcelona. Barcelona, Spain, 2002.

[2] Laroche, J.; Dolson, M. *Improved phase-vocoder time-scale modification of audio*. IEEE Transactions on Speech and Audio Processing, Volume: 7 Issue: 3, May 1999, Page(s): 323 –332.

[3] Bömers, Florian. *Wavelets In Real-Time Audio Signal Processing: Analysis and Simple Implementations*. Department of Computer Science, University of Mannheim. Mannheim, Germany, 2000.

[4] Shukla, Panchamkumar D. *Complex Wavelet Transforms and Their Applications*. Signal Processing Division, Department of Electrical Engineering, University of Strathclyde. Strathclyde, Scotland UK, 2003.

[5] Neumann, Julia; Steidl, Gabriele. *Dual-Tree Complex Wavelet Transform in the Frequency Domain and an Application to Signal Classification*. Department of Mathematics and Computer Science, University of Mannheim. Mannheim, Germany, 2003.

[6] Kingsbury, N. G. *Complex Wavelets for Shift Invariant Analysis and Filtering of Signals*. Journal of Applied and Computational Harmonic Analysis, vol 10, no 3, May 2001, pp. 234-253.

[7] Strang, G; Nguyen, T. *Wavelets and Filter Banks*. Wellesley-Cambridge Press, 1996.

[8] Rosa-Zurera, M. ; Ruiz-Ryes, N. ; Vera-Candeas, P. ; D; López-Ferreras, F. *Use of the Symmetrical Extension for Improving a Time-Varying Wavelet-Packet-Based Audio Coder*. Digital Signal Processing (Elsevier Science), vol. 13, pp. 457-469, July 2003.

[9] Arfib, Daniel. ; Bernardini, N. ; De Gozen, A. *Traditional Implementations of a Phase-Vocoder: The Tricks of the Trade*.Proceedings of the COST G-6 Confrerence on Digital Audio Effects (DAFX-00) Verona, Italy, December 7-9 2000.

[10] Gerhards, R.H. *Sound Analysis, Modification, and resynthesis with Wavelet Packets*. Masters Thesis, Simon Fraser Univerity, School of Engineering Science, November 2002.

[11] Oppenheim, A.V; Schafer, R. W.; Buck, J.R. *Discrete-Time Signal Processing 2nd ed*. Prentice-Hall, Inc, 1998.

[12] Bernsee, S, M. *Pitch Shifting Using the Fourier Transform*. www.dspdimension.com, 1999.

[13] Kingsbury, N. G. *Dual-Tree Complex Wavelets*. www.eng.cam.ac.uk/~ngk. Signal Processing Group, Dept. of Engineering, University of Cambridge. September 2004.

[14] Baranuik, R. G., Kingsbury, N. G., Selesnick, I.W. *The Dual-Tree Complex Wavelet Transform: A Coherent Framework for Multiscale Signal and Image Processing*. IEEE Signal Processing Magazine, November 2005, pp. 123-151.

[15] Roederer, J. *Introduction to the Physics and Psychophysics of Music*. Springer Press, 1975.

VITA

Jeffrey Livingston was born in Tulsa Oklahoma on August 8, 1971 to Harry and Sandra Livingston. He received a Bachelor of Arts in Music, with Jazz Piano Performance emphasis, from the University of New Orleans in December 1993, and a Bachelor of Science in Electrical Engineering from the University of Texas at Austin in August 2001. Between bachelor's degrees, Jeff worked as a professional musician for a number of musical groups based in Tulsa Oklahoma until moving to Austin in 1996. In September 2001 he began work at Cirrus Logic Inc. in Austin Texas as an audio DSP firmware engineer. In 2003 he was admitted to the graduate school of the University of Texas at Austin to pursue a Master's degree in Electrical Engineering. In September 2004, Jeff left Cirrus to work full-time on his master's degree and resumed working at Cirrus again in November 2005 while continuing graduate studies part-time at UT.

Permanent address: 1022 Quail Park Drive
Austin, TX. 78758