The Report Committee for Aaron Charles Treptow
Certifies that this is the approved version of the following report:

Pitch Shifting Techniques for High-Frequency Passive Sonar Audio

APPROVED BY
SUPERVISING COMMITTEE:

Neal Hall, Supervisor

Gregory Allen, Co-Supervisor

Pitch Shifting Techniques for High-Frequency Passive Sonar Audio

by

Aaron Charles Treptow, BM

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 2016

Acknowledgements

I would like to thank the following people for their generous assistance in producing this report. Their contributions, guidance, expertise, and words of encouragement were invaluable to me:

- Dr. Gregory Allen
- Dr. Brian Evans
- Dr. Neal Hall
- Daniel Huff
- Bernard Maldonaldo
- Steven Morrissette

Thank you for your time, energy, and support.

Abstract

Pitch Shifting Techniques for High-Frequency Passive Sonar Audio

Aaron Charles Treptow, MSE

The University of Texas at Austin, 2016

Supervisors: Neal Hall and Gregory Allen

Listening to passive sonar signals is a vital tool for sonar operators to classify underwater sound sources. While many passive sonar systems operate in the human auditory range (20 Hz to 20 kHz) there are a considerable number of high-frequency systems that extend beyond this range. This report examines pitch shifting algorithms for compressing ultrasonic, bandlimited passive sonar signals down into the auditory spectrum. By utilizing pitch shifting techniques the signal's harmonic structure and length in time are retained. The frequency spectrum is lowered into the auditory range so that the sonar operator may then listen and characterize targets. Three pitch shifting algorithms are examined: Waveform Similarity Overlap-Add (WSOLA), Phase Vocoder, and Constant-Q Transform (CQT). Both synthetic and real sonar data is experimentally applied to each method and results are presented. Comparisons of performance are provided with an emphasis on feasibility for real-time sonar system implementation.

Table of Contents

List of Tables vii

List of Figures ix

Chapter 1: Introduction 1

Chapter 2: Passive Sonar Background 3

  Underwater Acoustics 3

  Passive Sonar 6

Chapter 3: Pitch Shifting Algorithms Investigated 11

  Time and Pitch Scale Modification (TSM & PSM) 12

  Overlap Add Algorithm 14

  Waveform Similarity OLA Implementation (WSOLA) 16

    WSOLA Artifacts and Improvements 20

  Phase-Vocoder 21

    Fast-Fourier Transform (FFT) Phase Vocoder 22

    Phase Vocoder Artifacts and Improvements 25

  Constant-Q Transform (CQT) Algorithm 27

    CQT Pitch Scaling 31

    CQT Artifacts and Improvements 33

Chapter 4: Experimental Methodology 34

  Experimental Model 34

  Synthetic and Sonar Input Signals 39

  Algorithm Implementation 44

Chapter 5: Results 47

  Synthetic Data Results 47

  Sonar Data Results 54

  Comparison of Algorithms 59

Chapter 6: Conclusion..61

References..62

List of Tables

Table 1: Comparison of PSM Algorithms 59

List of Figures

Figure 2.1: Sound Speed Profile in the Ocean [5] 4

Figure 2.2: Model of Acoustic Propogation Examples in the Ocean [8] 5

Figure 2.3: Average Deep-water Ambient Noise Spectra [4] 6

Figure 2.4: Cylindrical Sonar Transducer Array [4] 8

Figure 2.5: Delay-and-Sum Beamformer 9

Figure 2.6: Digital-Down Conversion 10

Figure 3.1: TSM Analysis-Synthesis Procedure 12

Figure 3.2: PSM through TSM and Resampling 14

Figure 3.3: OLA Procedure with $\alpha < 1$ 15

Figure 3.4: OLA Phase Jumping [11] 16

Figure 3.5: WSOLA Alogrithm Procedure [11] 17

Figure 3.6: WSOLA Periodicity Preservation [16] 19

Figure 3.7: Transient Doubling Artifact of WSOLA Signals [11]. 20

Figure 3.8: Phase Vocoder Scaling Examples [12]. 24

Figure 3.9: CQT Bins on Time-Frequency Plane [26]. 30

Figure 3.10: Comparison of CQT Time-Frequency Sampling Schemes [26].......31

Figure 4.1: Frequency Spectrum of Experimental Sonar Signal 35

Figure 4.2: Flowchart of the Experimental Pitch Shift Process 36

Figure 4.3: Frequency Spectrum Pitch-Shift Model. 37

Figure 4.4: TSM Analysis-Synthesis Procedure. 38

Figure 4.5: CQT Analysis-Synthesis Procedure. 39

Figure 4.6: In-band Synthetic Noise Spectrogram 40

Figure 4.7: Five Second Chirp Signal Spectrogram 40

Figure 4.8: Synthetic Sonar Signal Spectrogram 41

Figure 4.9: Synthetic Sonar Signal Time Plot 42

Figure 4.10: FIR-Filter for Synthetic Data 42

Figure 4.11: Target A Waterfall Plot...43

Figure 4.12: Target B Waterfall Plot...43

Figure 4.13: Target C Waterfall Plot...44

Figure 4.14: Flowchart of Experimental Process 45

Figure 5.1: Spectrograms of 5 s Linear Chirps 48

Figure 5.2: Spectrograms of 100 ms Delayed Linear Chirps..................................50

Figure 5.3: Spectrograms of 20 ms Impulse Chirp Train. 51

Figure 5.4: Time-Domain Comparisons of Single 20 ms Chirp. 52

Figure 5.5: Shifted In-Band Noise Comparison. 53

Figure 5.6: Comparison of Shifted Synthetic Sonar Signal 54

Figure 5.7: Comparison of Outboard Motor Recording (Target A) 56

Figure 5.8: Comparison of Ping Reflection (Target B) 57

Figure 5.9: Comparison of Passing Jet Ski (Target C) 57

Chapter 1: Introduction

Since its inception over 100 years ago, passive sonar has been an invaluable tool used to silently detect and characterize both manmade and biological underwater acoustic signals [1]. Sonar transducers electrically convert acoustic energy and, through signal processing, passive sonar systems can provide sonar operators both visual and sonic representations of the processed input signals. The underwater acoustic environment allows for acoustic propagation of almost six decades of useful frequencies (a few Hz to several hundred kilohertz), and detection is dependent on each sonar system's unique design. In the case of passive sonar audio, the signal processing to provide a sonar operator real-time sound of the reconstructed input signal is trivial when detection is within the range of human hearing (20 Hz to 20 kHz) [2]. While many passive sonar systems operate within this range, there is also interest in utilizing ultrasonic (> 20 kHz) passive sonar implementations in the field. To provide sonar operators audio that somehow represents ultrasonic input signals, additional audio processing is required. Pitch shifting is of interest in order to adjust the frequency content, while retaining the harmonic relationships of the out-of-band acoustic input signal.

Pitch shifting is a heavily researched and implemented technique in the fields of speech and music processing, however, as of the writing of this report, there does not currently exist any previously publicly published material as applied to sonar signal processing. That's not to say that methods to translate ultrasonic frequencies, including sonar signals, into the auditory range do not already exist. In fact, high frequency (HF) passive sonar systems during World War II were designed, built, and deployed to map ultrasonic signals for sonar operators to listen and characterize targets through frequency modulation [3, pg. 247]. Ultrasonic input signals were down-converted through cosine multiplication, known as heterodyne mixing. However, heterodyne mixing does not preserve the spectral characteristics or phase relationships of the incoming acoustic signal. Pitch-shifting dilates (or compresses) the frequency content and provides additional techniques, unique to each algorithm, to retain the phase relationships. By reconstructing the translated sonar signal, the operator may be able to better characterize the target on playback.

The following report researches pitch-shifting band-limited HF passive sonar signals down into the auditory spectrum. Three algorithms (WSOLA, Phase Vocoder, and Constant-Q Transform) are investigated, experimentally applied to both synthetic and sonar data, and then results are compared. First, a brief background chapter on underwater acoustics and passive sonar signal processing is provided to give context into the passive sonar input signals. Then, the three-aforementioned pitch-shifting algorithms are investigated, including discussion of potential algorithmic artifacts and enhancements. Next, the experimental methodology is shown and justified, followed by experimental results and comparisons of the algorithms performance.

Chapter 2: Passive Sonar Background

Before investigating pitch shifting algorithms and the subsequent experiment, a background discussion on passive sonar is presented to better understand the underwater acoustic environment and sonar signal processing. The unique properties of underwater acoustic propagation and the input signals generated through beamforming and complex down conversion signal processing add context to the ultrasonic pitch shifting experiment. The following chapter provides background passive sonar information, including an introductory understanding of the underwater environment and acoustics, and the sonar signal processing utilized to produce the input signals utilized in the experiment.

Underwater Acoustics

Starting in the early nineteenth century, then earnestly following World War I, underwater acoustics is a complex and active field of research. Compared to the relatively homogenous medium of air, the underwater environment provides a more diverse set of parameters to address to better understand acoustic propagation. A combination of surface boundaries, salinity, noise, and temperature and pressure gradients all significantly contribute to the underwater acoustic field, including sound speed, attenuation, and boundary reflections and scattering. This section is not intended to be exhaustive, but rather inform the reader of the fundamental characteristics of the underwater acoustic environment as compared to air. Further suggested reading can be sourced from [4-6].

Fundamental to all compressional (longitudinal) plane wave acoustic propagation, pressure is defined by the relation between the velocity of the fluid particles u, the density of the medium ρ, and propagation of the compressional wave *c:*

$$p = \rho c u \qquad (2\text{-}1)$$

where ρc is the *specific acoustic resistance* of the medium. In the case of sea water, $\rho c = 1.5 * 10^5 \frac{g}{s*cm^2}$, whereas for air, $\rho c = 42 \frac{g}{s*cm^2}$, and the propagation of sound is seen as the relation of the medium's compressibility and density:

$$c = \frac{p}{u} * \frac{1}{\rho} \qquad (2\text{-}2)$$

The density of sea water is function of temperature, static pressure (based on depth), and salinity. Empirical equations, including one presented by [7] are commonly used to estimate the speed of sound at a given density at sea. Figure 2.1, below, shows the standard sound speed profile as a function of varying density. On average, the velocity of sound in water (both fresh and salt) is 1,450 m/s, close to 4 times greater than that in air (343 m/s), making it far more conductive to sound.

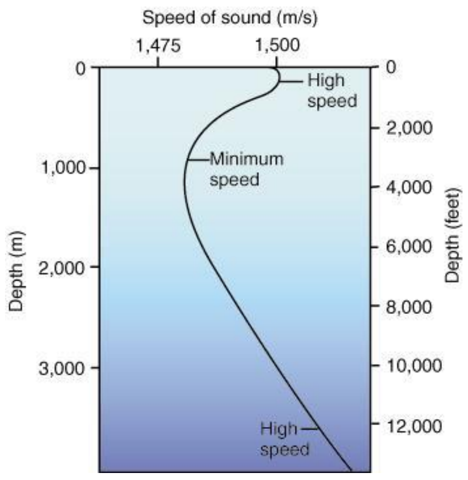

Figure 2.1: Sound Speed Profile in the Ocean [5].

Loss mechanisms due to spreading loss, volume absorption, and both reflection and scattering losses attenuate acoustic waves underwater. Standard geometrical (spherical and cylindrical) spreading is the largest contributor to attenuation. Volume absorption, through fluid viscosity and molecular chemical relaxation, attenuates as a function of frequency and is the greatest factor in the attenuation of high frequencies. In addition, ocean surface and floor reflections add loss to incident sound waves either by transfer of acoustic energy, or by rough interface scattering.

The sound speed profile gradient at varying depth, coupled with surface and ocean bottom boundaries, add significant propagation factors in the underwater acoustic environment. Essentially, sound waves continue to bend in the direction of slower sound speeds during propagation and where a sound source originates has ramifications on how far it may travel before it is attenuated below the noise floor. Acoustic sources submerged just below the surface can skip along the boundary (surface duct) and never deviate towards the bottom. They can be trapped within an underwater waveguide (e.g. the SOFAR channel [4]) and travel large distances (frequency-dependent), reflect from multiple surface and bottom boundaries in shallow water, or can bend from deep water towards the surface.

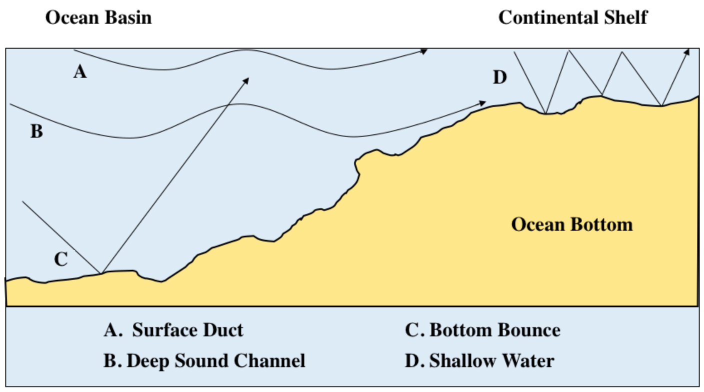

Figure 2.2: Model of Acoustic Propagation Examples in the Ocean [8].

Given its high conductivity to sound and multiple propagation paths, the ocean is inherently a noisy environment. Background noise due to biologics, weather, and man-made sources all contribute to the overall underwater ambient noise spectrum (Figure 2.3). In lower frequencies, noise is dominated by man-made shipping industry traffic. In mid to high frequencies, sea surface wave roughness due to weather, including rain and wind, are the greatest contributors. Noise is also found within the sonar system, itself, including thermal and self-noise. Thermal noise, due to the molecular movement of electrons in a sonar system, is dominant at frequencies above 100 kHz. Self-noise is a combination of acoustic, mechanical, and electronic noise created either by, a sonar system's interaction with its environment, or self-generated noise by the vessel on which the sonar is mounted [9]. Both thermal and self-noise sources, frequency-ranges, and amplitudes are dependent on each sonar system.

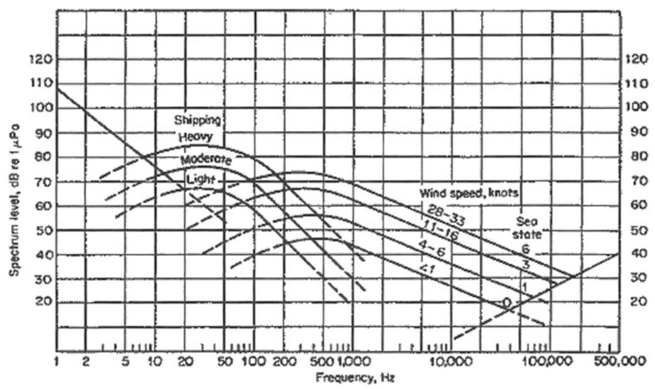

Figure 2.3: Average Deep-water Ambient Noise Spectra [4].

PASSIVE SONAR

Passive sonar historically originates from a military need to silently detect submarines during World War II. In active sonar, the sonar system initiates an acoustic signal and then listens to its return. Passive sonar simply listens to external sources. The following section provides an overview of a passive sonar system utilizing the well-known sonar equation used to estimate a system's ability to detect a target. In particular, the signal processing array gain term in the equation, through use of beamforming, is discussed. In addition, the digital-down conversion process, by complex basebanding the sonar signal to reduce computational complexity is introduced.

The passive sonar equation is used to give an estimated prediction on the ability to detect a signal embedded in noise.

$$SE = SL - NL + AG - TL - DT \qquad \text{(2-3)}$$

where, SE = signal excess at the input to the detector

SL = source level on its acoustic axis (sound pressure level, re 1 μPa at 1 m)

NL = noise level at the receiver (re 1 $\frac{\mu Pa^2}{Hz}$)

AG = array gain based on directivity of the sonar system

TL = transmission loss from the source to the receiver

DT = detection threshold required by the sonar system to perform detection.

All terms are in units of dB and referenced to 1 μPa (micropascal). Having a signal excess greater than 0 dB denotes that the target signal can be detected. Whereas source level (SL), noise level (NL, apart from self and thermal noise), and transmission loss (TL) are all dependent on either the target or environmental factors previously discussed, both array gain (AG) and detection threshold (DT) are solely dependent on the sonar characteristics, including its signal processing.

Sonar systems typically consist of an array of transducers, often configured in either a planar or cylindrical shape (Figure 2.4). Spatial filtering of the array's output signal, through the formation of directional beams, is known as beamforming. This provides both increased directionality and array gain by enhancing the amplitude of a coherent wave front relative to the background noise and directional interference. An important beamforming algorithm in practice is the *delay-and-sum beamformer*.

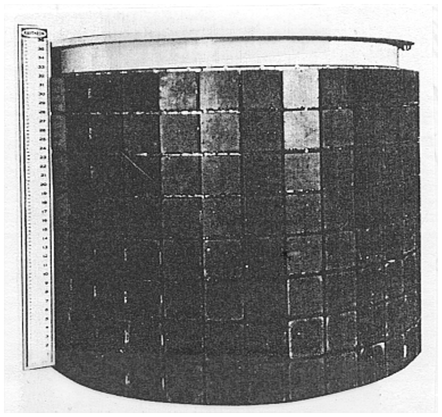

Figure 2.4: Cylindrical Sonar Transducer Array [4].

Given an array of N sensors, let $x_n(t)$ be the impinging sound wave on the array and $b(t)$ is the beam time series output of the beamformer (Figure 2.5). Time shifts (τ_n) are applied to each channel in the array and are multiplied by a *shading coefficient* (w_n) that shapes the beam's response. The delay-and-sum beamformer output is defined as:

$$b(t) = \sum_{n=0}^{N-1} w_n x_n(t - \tau_n) \quad (2\text{-}4)$$

where each channel is processed then summed together for the final beam time series output. The beam's direction, called the Maximum Response Angle (MRA), can be chosen arbitrarily. By introducing a delay on each channel, the array can be steered in a specified direction. Given an array geometry and desired MRA, the delay for each sensor is determined by projecting the sensors onto a plane perpendicular to the MRA. This distance is divided by the speed of sound to calculate the necessary delay for each sensor using the desired MRA [10].

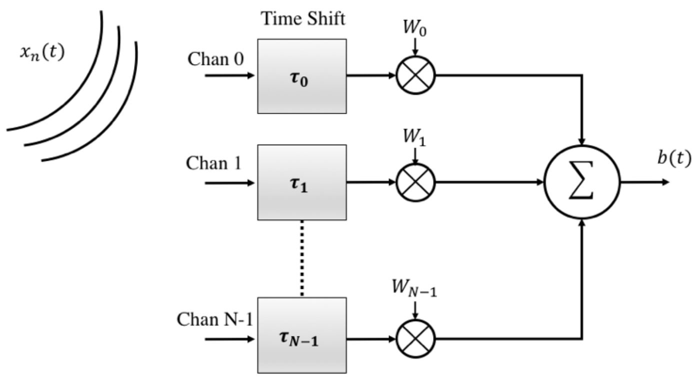

Figure 2.5: Delay-and-Sum Beamformer.

In addition to beamforming, bandlimited sonar signals are often down converted to a lower, more computationally efficient sampling rate by a complex basebanding digital-down conversion process (Figure 2.6). The real-valued band-limited signal is multiplied, in parallel, by both a cosine and sine term. The mixer frequency of both cosine and sine is chosen so that the difference of the positive cosine-multiplied (in-phase) signal and the summation of the negative sine-multiplied (quadrature) signal are centered at 0 Hz (baseband). The basebanded in-phase and quadrature signals share the same amplitude spectrum with an orthogonal 90-degree phase difference. After lowpass filtering and downsampling, the entire original input bandwidth is retained in a lower sampled, complex basebanded signal.

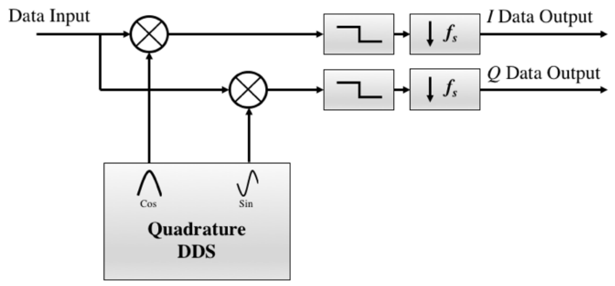

Figure 2.6: Digital-Down Conversion.

The discrete-time electroacoustic output from the down converted, beamformed sonar signal is then further processed, and analyzed, for target detection, graphical visualization, or, as in the case of this report, audio processing and playback.

Chapter 3: Pitch-Shifting Algorithms Investigated

The intent of pitch shifting is to preserve the time-scale and harmonic structure of the incoming audio signal while dilating, or compressing, its frequency content. While pitch shifting has been a heavily researched topic in the fields of speech and music over the past 50 years, there is no readily accessible documentation as applied to compressible ultrasonic sound waves, like those encountered in underwater sonar. In the case of HF passive sonar, the beamformed sonar signal may require being shifted into the auditory spectrum, which could entail compressing the frequency content multiple octaves and significantly past the original intent of the algorithms investigated in this report.

To understand the algorithms utlized in this experiment, first an analysis-synthesis model of time and pitch scale modification (TSM and PSM) is presented, along with the basic overlap-add method (OLA). The TSM analysis-synthesis model, and subsequent PSM and OLA processes, form the basis of the pitch shifting algorithms presented in this chapter. Three algorithms are investigated: Waveform Similarity Overlap Add (WSOLA), Phase Vocoder, and an invertible Constant-Q Transform (CQT). These algorithms were chosen for two reasons. First, they are a diverse collection of the pitch shifting methods in use today as each are implemented in different domains. Secondly, all three have well referenced, readily available open source libraries in C, Python, and *Matlab*, respectively[1]. For each pitch-shifting method, the underlying theory is presented, as well as known limitations and proposed enhancements. To reduce confusion, the notation between algorithms is kept as consistent as possible. Driedger's recently published *A Review of Time-Scale Modification of Music Signals (2016)* [11] proved to be an invaluable source,

[1] See Chapter 4 for specific libraries.

both for laying the notational groundwork of this chapter, and his concise presentation of time-based TSM theory.

TIME-SCALE AND PITCH-SCALE MODIFICATION (TSM & PSM)

Time-scale modification of an audio signal changes its time-scale without affecting its frequency content. The resulting effect is perceived on playback as speeding up, or slowing down, without affecting the pitch of the original signal. This can be viewed as an *analysis-synthesis* model (Figure 3.1), a technique common to many digital signal processing algorithms. Here, an input audio signal is decomposed into equal overlapping, successive frames for analysis. The idea is that the localized frequency content of the signal is captured in each frame.[2] After time-scale modification, the frames are then synthesized together and, in theory, the signal's pitch has not been altered while the time scale has either been reduced or elongated.

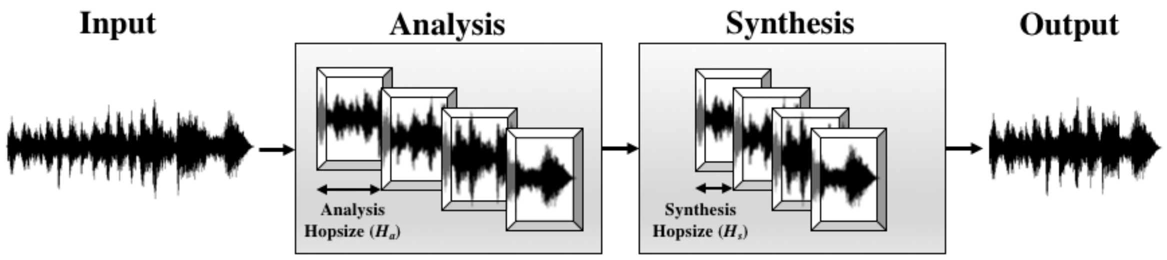

Figure 3.1: TSM Analysis-Synthesis Procedure.

Following the notation presented in [11], the TSM input is a discrete-time audio signal $x\colon \mathbb{Z} \to \mathbb{R}$ and is evenly sampled at a sampling rate, f_s. In the TSM analysis stage, x

[2] Frame lengths vary based on input signal, but 20 milliseconds is common in speech and music. [12]

is divided into short frames, x_m, where m: $\in \mathbb{Z}$. Each frame is of length N samples, and evenly spaced by *analysis hopsize*, H_a:

$$x_m(n) = \begin{cases} x(n + mH_a), & if\ n \in \left[-\frac{N}{2} : \frac{N}{2} - 1\right], \\ 0 & otherwise. \end{cases} \quad (3\text{-}1)$$

In the TSM synthesis stage, the frames are positioned on the time axis based on a specific *synthesis hopsize*, H_s. When $H_s \neq H_a$, there is a modification of the input signal's time scale by a *stretching factor* $\alpha = \frac{H_s}{H_a}$. In practice, H_s is commonly held constant[3] with specific synthesis overlaps to avoid amplitude modulation at the output [12] and analysis hopsize $H_a = \frac{H_s}{\alpha}$. If one were to start summing the overlapped, time stretched synthesis frames without further modification, the resulting output signal would be subject to undesired artifacts at the frame boundaries. These artifacts include phase discontinuities and amplitude fluctuations. In fact, it's the varying methods to handle these frame boundary modifications that, in large part, define the TSM algorithms presented in this report. Therefore, prior to synthesis, the analysis frames are modified accordingly (algorithm dependent) and form synthesis frames, y_m. The frames are overlapped and summed to construct the TSM output signal:

$$y(n) = \sum_{m \in \mathbb{Z}} y_m(n - mH_s). \quad (3\text{-}2)$$

Pitch-scale modification (PSM) allows pitch-shifting without affecting the audio playback rate. In practice, this is commonly achieved by a combination of TSM and resampling. The first stage performs a stretch factor, α, TSM then the resulting signal is resampled at $\frac{f_s}{\alpha}$. Now, the output signal has the same duration as the input with its frequency

[3] Typically, $H_s = \frac{N}{4}$ [12]

content expanded, or compressed, by α. It is important to note that reversing the order of TSM and resampling yields the same PSM result. However, in the case of frequency compression it is better to perform TSM first, then resample, since the time-scaling stage yields a signal of smaller duration ($\alpha < 1$) for computational efficiency [13]. In Figure 3.2, the PSM output is achieved after resampling the TSM output from Figure 3.1.

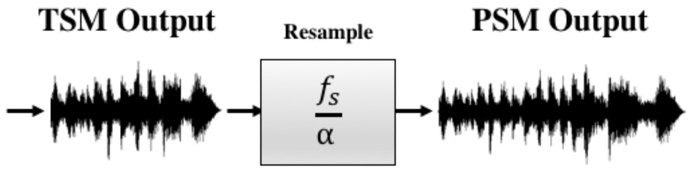

Figure 3.2: PSM through TSM and Resampling.

OVERLAP ADD ALGORITHM (OLA)

A natural progression from the TSM model is the time-based overlap-add method (OLA). OLA is a straightforward solution to addressing the potential synthesis frame edge phase discontinuities and amplitude fluctuations from the summation of overlapped synthesis frames. To enforce smooth transitions between frames, a window function, *w(n)*, is applied to the analysis frames prior to synthesis reconstruction. The most common choice for *w(n)* is the Hann window function:

$$w(n) = \begin{cases} \frac{1}{2}\left(1 - \cos\left(\frac{2\pi\left(n+\frac{N}{2}\right)}{N-1}\right)\right), & if\ n \in [-\frac{N}{2} : \frac{N}{2} - 1] \\ 0, & otherwise \end{cases} \qquad (3\text{-}3)$$

Equation 3-1 is used to compute each m^{th} analysis frame, x_m. The window, $w(n)$, is applied to x_m and a scaling operation is performed using the sum of overlapping window functions (0 to *k windows)* in the denominator. This scaling operation provides amplitude correction of each synthesis frame, y_m:

$$y_m(n) = \frac{w(n)x_m(n)}{\sum_{k\in\mathbb{Z}} w(n-kH_s)} \qquad (3\text{-}4)$$

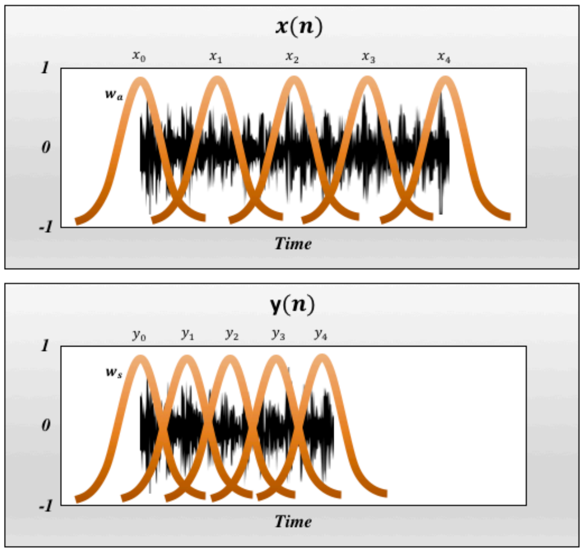

Figure 3.3: OLA Procedure with $\alpha < 1$. (Note that window lengths are held constant per stage).

It is important to note that the size of $w_a(n)$ should be longer than one pitch period of the lowest fundamental frequency contained in $x(n)$ to retain harmonic content in the windowed analysis frame. However, given a stretch factor of $\alpha \neq 1$, the OLA procedure suffers from phase jump artifacts during the frame reconstruction in the synthesis stage. In general, OLA is not capable of retaining local periodic structures that are present in the input signal [11]. This occurs at the frame boundaries during re-synthesis when the phase of the fundamental frequencies of two successive, overlapped frames are not the same. Therefore, the OLA method is not acceptable for input signals containing harmonic components since it exhibits artifacts like warbling, which is a type of periodic frequency modulation observed in processed polyphonic signals [14, pg. 99].

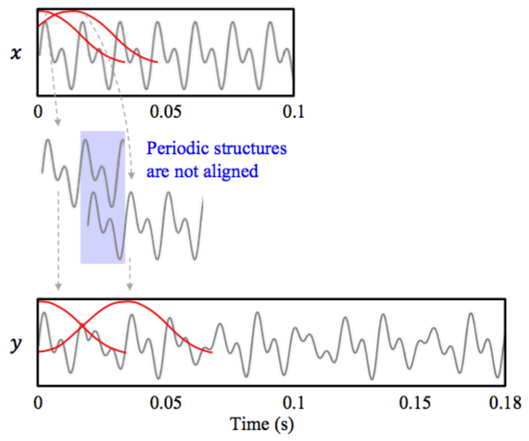

Figure 3.4: OLA Phase Jumping [11].

Waveform Similarity Overlap Add Algorithm (WSOLA)

Waveform similarity overlap-add (WSOLA) was first proposed in 1993 [15] as a high quality TSM for speech processing. Based on TSM overlap-add (OLA), WSOLA is a time-based time-scaling method that allows for tolerance at the analysis frame edges and performs optimization techniques to adjust where successive frames overlap. The purpose in adjusting the successive overlapped frame locations is to maintain waveform continuity and periodicity in the signal (i.e. to avoid warbling). In this section, the WSOLA algorithm is presented, highlighting its improvements to the overlap-add method. In addition, this section discusses WSOLA's notable TSM artifacts and proposed solutions.

The underlying issue with the OLA algorithm is that it is not sensitive to the input signal. It copies windowed analysis frames from a fixed position of the input signal to fixed positions in the output. Analysis frames have no effect on the outcome of the algorithm. In WSOLA, tolerance regions ($\pm\Delta_{max}$) are allowed for each successive analysis frame. The addition of the tolerance region allows the algorithm to apply optimization techniques, such as auto-correlation, with the current and previous frame to determine the best location for the analysis frame overlaps. Given the best location within the tolerance region, the frame is overlapped and applied to the previous frame, and the iterative process continues to the next frame for similar treatment. Given the m^{th} iteration in the process, let us call the previously adjusted frame the *adjusted analysis frame* x'_m where:

$$x'_m(n) = \begin{cases} x(n + mH_a + \Delta_m), & if\ n \in \left[-\frac{N}{2} : \frac{N}{2} - 1\right], \\ 0, & otherwise \end{cases} \qquad (3\text{-}5)$$

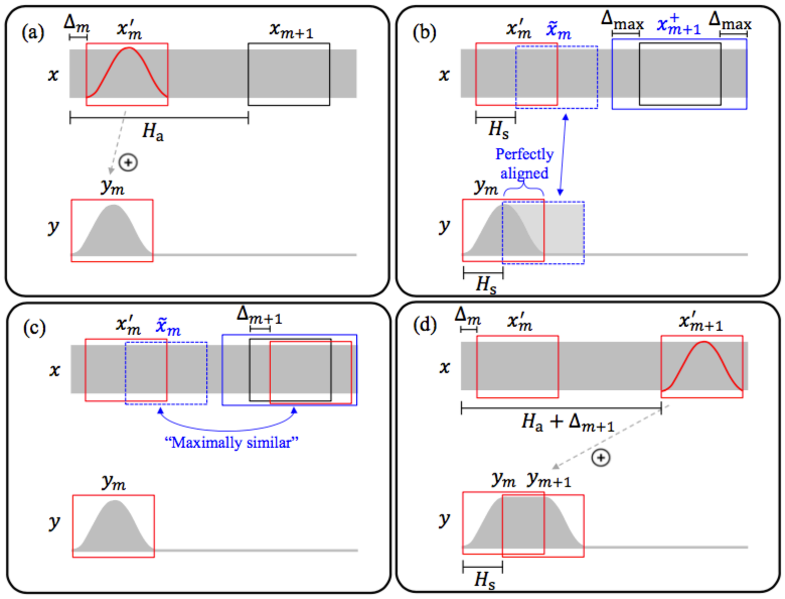

Figure 3.5: WSOLA Algorithm Procedure [11].

The adjusted analysis frame, x'_m, is windowed and copied to the output signal, y, as in the OLA method. The next step is to adjust the position of the next analysis frame, x_{m+1}, which can be interpreted as a constrained optimization problem. A shift index, $\Delta_{\mathrm{m+1}} \in [-\Delta_{\mathrm{max}}:\Delta_{\mathrm{max}}]$, is chosen such that periodic structures of the adjusted analysis frame, $\mathrm{x}'_{\mathrm{m+1}}$, are optimally aligned with structures of the previously copied synthesis frame, y_{m}. The two frames are then windowed, overlapped, and summed together at the synthesis hopsize H_{s}. Note, in the case when the stretching factor is 1, the obvious next frame selection would be the natural progression of the input signal. However, given both the constraint of $H_s \neq H_a$, and $\Delta_{m+1} \in [-\Delta_{max}:\Delta_{max}]$, x'_{m+1} must be in an extended frame region, x^+_{m+1}, given as:

$$x^+_{m+1}(n) = \begin{cases} x(n + (m+1)H_a), \ if\ n \in \left[-\frac{N}{2} - \Delta_{max}:\frac{N}{2} - 1 + \Delta_{max}\right], \\ 0, \qquad\qquad otherwise. \end{cases} \tag{3-6}$$

The next adjusted frame, x'_{m+1}, must come from selecting the waveform within its region most like the naturally occurring frame following x'_m, notated as $\hat{x}_m$:

$$\hat{x}_m(n) = \begin{cases} x(n + mH_a + \Delta_m + H_s), \ if\ n \in \left[-\frac{N}{2}:\frac{N}{2} - 1\right], \\ 0, \qquad\qquad otherwise \end{cases} \tag{3-7}$$

As proposed by [15], one method to determine similarity between x'_{m+1} and $\hat{x}_m$, is to perform *cross-correlation* between the two waveforms:

$$c(p, q, \Delta) = \sum_{n \in \mathbb{Z}} q(n) p(n + \Delta) \tag{3-8}$$

where $\hat{x}_m = q$ and $x'_{m+1} = p$. The optimal shift index Δ_{m+1} is computed by obtaining the maximum cross-correlation location between the two signals:

$$\Delta_{m+1} = \frac{argmax}{\Delta \in [-\Delta_{max} : \Delta_{max}]} \, c(q, p, \Delta) \qquad (3\text{-}9)$$

With the calculated shift index Δ_{m+1}, we then compute the new synthesis frame, y_{m+1}, similar to Equation 3.3 for OLA:

$$y_{m+1}(n) = \frac{w(n) x_m(n + (m+1) H_a + \Delta_{m+1})}{\sum_{k \in \mathbb{Z}} w(n - kH_s)} \qquad (3\text{-}10)$$

We then use Equation 3.2 to create the output signal. The entire TSM process of the WSOLA algorithm can be seen in Figure 3.5. For PSM, we follow the same procedure of resampling the WSOLA TSM output back to the original time-length. As seen in Figure 3.6, the WSOLA algorithm maintains relatively simple periodic patterns of input signals. However, significant transient and complex, polyphonic pitch artifacts are still common and discussed in the next section.

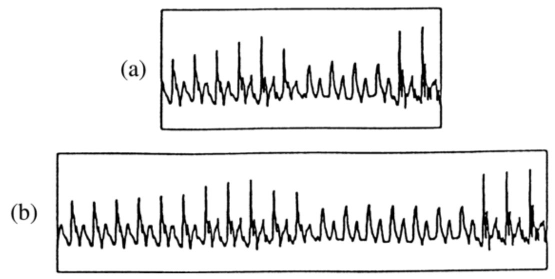

Figure 3.6: WSOLA Periodicity Preservation. Part a) is the original signal and b) is the corresponding WSOLA output waveform when slowed down 60% speed [16].

WSOLA Limitations and Enhancements

As stated in [17], the WSOLA method suffers from two critical artifacts: *transient doubling, or skipping*, and distortion to the temporal envelope of complex, polyphonic input signals, such as music. As seen in Figure 3.8, if a single transient is captured within the overlapping region between two successive analysis frames (x'_m and x'_{m+1}) the result is a doubling of the transient on overlapped synthesis frames at output. Consequently, this also means that if the analysis hopsize is large then transients may be skipped entirely. Transient skipping is significant in context to our experiment since the stretching factor will be relatively small to shift the HF sonar signals into the auditory range.

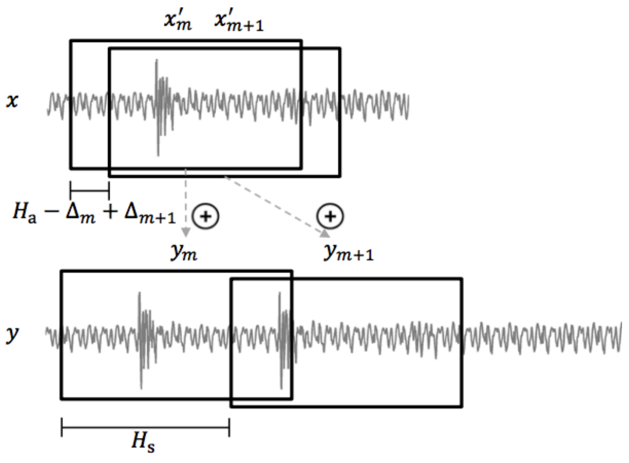

Figure 3.7: Transient Doubling Artifact of WSOLA Signals [11].

One technique to avoid transient skipping or doubling is presented by [17]. By using a supplemental transient detection algorithm, the process temporarily alters H_a to be equal to H_s during the presence of a transient. The analysis window is transferred to the synthesis stage unaltered, thus preserving the detected transient. In areas between detected transients H_a is dynamically altered in order to preserve the original, intended global stretching factor.

The other prevalent WSOLA artifact is warbling of complex, polyphonic input signals, such as orchestral music. The WSOLA algorithm, by design, is intended to preserve the fundamental frequency per analysis frame. For input signals, and analysis frames, with complex harmonic content, the algorithm can only optimize the strongest detected periodic signal and fails to preserve the entire harmonic relationship. As stated in [11, pg. 10], to assure that WSOLA can adapt to the most dominant pattern in the waveform of the input signal, one frame must be able to capture at least a full period of the pattern. In addition, the tolerance parameter, Δ_{max}, must be large enough to allow for an appropriate adjustment. Therefore, it should be set to at least half a period's length.

PHASE VOCODER

One of the most prominent pitch shifting algorithms in use today is the phase vocoder. Originating in 1966 [18], the phase vocoder was the first TSM algorithm to represent and alter signals based their short-time amplitude and phase spectra in the frequency domain. Based off the channel vocoder [19], the phase vocoder is named by the algorithm's adjustment of time-varying phase, or instantaneous frequency, of the modified input signal. First presented in a digital filter bank implementation [18], a practical, digitally implemented phase vocoder was realized in 1976 [20]. Utilizing the Fast Fourier Transform (FFT) algorithm, short-time Fourier transform (STFT) TSM windows are overlapped with modifications to the signal's phase at each frame. The following section presents the original phase vocoder algorithm [20] through use of the STFT. First, the STFT is defined in context to the previously presented TSM analysis-synthesis model with an additional phase correction stage before re-synthesis. Next, the phase vocoder algorithm's limitations are discussed as well as enhancements to the original algorithm to reduce computational complexity and improve its performance.

Fast Fourier Transform (FFT) Phase Vocoder

While there is educational value in investigating the filter bank implementation[4] of the phase vocoder, its computational costs prohibit practical use. The STFT FFT phase vocoder implementation, however, has a long-standing history of practical, real-time implementation [12]. The FFT phase vocoder can be viewed as a complementary to the filter bank method, where each successive STFT of the input signal is like a group of parallel, linearly-spaced filter banks. The number of filter banks is equal to the number of frequency bins in the STFT. Successive input signal frames are windowed, transformed into the frequency-domain, and analyzed into overlapping frames separated by an analysis hopsize, H_a. Like in the time-domain case, TSM is realized by varying H_a with respect to the synthesis hopsize, H_s, with scaling factor, $\alpha = \frac{H_s}{H_a}$. While keeping consistent with previous TSM and OLA notation, the following phase vocoder analysis and definitions are based on Laroche's *Improved Phase Vocoder Time-Scale Modification of Audio* [14].

Analysis time-instants, t_a^m, for successive integer values, m, are set uniformly for the input signal, x(n), where $t_a^m = mH_a$. At each time-instance, t_a^m is windowed by $w(n)$ and a Fourier transform is calculated resulting in a STFT representation of the input signal:

$$X(t_a^m, \Omega_k) = \sum_{n=-\infty}^{\infty} w(n)x(t_a^m + n)e^{-j\Omega_k n} \qquad (3\text{-}11)$$

[4] Readers are encouraged to read [21] for a filter bank explanation.

$\Omega_k = \frac{2\pi k}{N}$ is the center frequency of the kth vocoder channel (bin) in radians per sample, and N is the size of the FFT. $X(t_a^m, \Omega_k)$ is both a function of time (via variable m) and frequency (via Ω_k) and equation 3.11 is evaluated from $0 \leq k < N$.

Without modification, the output signal is realized by setting uniform synthesis time instants, t_s^m, where $t_s^m = H_s m$. Each short-time signal, y_m(n), is found by taking the Inverse-Fourier transform of the synthesis STFT, $Y(t_s^m, \Omega_k)$. The short-time signals are windowed then summed together to form $y(n)$. Since a windowing function is applied to reduce spectral leakage during the analysis stage, the synthesis frames must be overlapped if a useable synthesis is required. Generally, 75% ($\frac{3N}{4}$) overlap with a Hanning window is recommended for TSM to avoid modulation at synthesis [12].

$$y(n) = \sum_{m=-\infty}^{\infty} w(n - t_s^m) y_m(n - t_s^m) \qquad (3\text{-}12)$$

$$y_m(n) = \frac{1}{N} \sum_{k=0}^{N-1} Y(t_s^m, \Omega_k) e^{j\Omega_k n} \qquad (3\text{-}13)$$

Time-scale modification is done in two steps. First, the time-scaling is a result of the stretching factor α. Second, the phase values between successive time-scaled synthesis STFTs must be tracked and adjusted between each time instance to avoid discontinuities. Figure 3.9 shows three examples of how α and the TSM is altered based on H_a with H_s held constant at $\frac{N}{4}$ samples (75% Overlap).

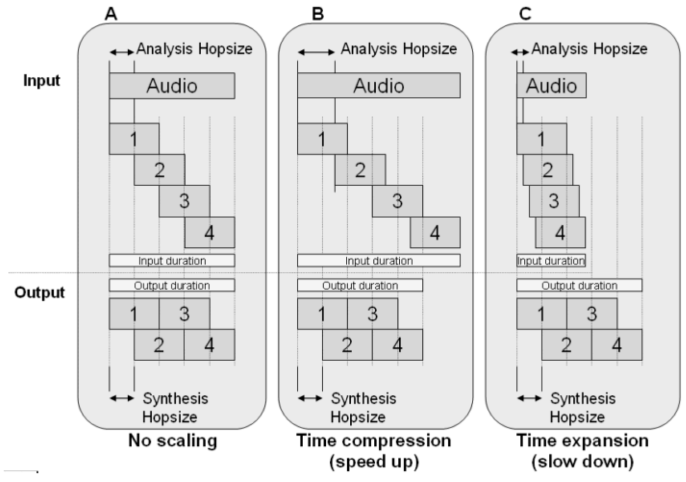

Figure 3.8: Phase Vocoder Scaling Examples [12].

To maintain phase continuity between successive output frames the synthesis frames must overlap synchronously through a phase update process. To calculate suitable synthesis phases, the instantaneous frequency, $\widehat{w}(t_a^m)$, of each bin is first calculated. Using $\widehat{w}(t_a^m)$, it is possible to predict the expected phase of any component for any given H_s. The heterodyned phase increment is calculated by subtracting successive phase calculations as seen in equation 3.14, below.

$$\Delta\Phi_k^m = \angle X(t_a^m, \Omega_k) - \angle X(t_a^{m-1}, \Omega_k) - H_a\Omega_k \in [-\pi{:}\pi] \quad (3\text{-}14)$$

where $\angle X(t_a^m, \Omega_k)$ and $\angle X(t_a^{m-1}, \Omega_k)$ represent the phase of the current and previous analysis frames, respectively, and $H_a\Omega_k$ is the predicted phase difference for one hop period for the center frequency of current STFT bin. The instantaneous frequency, $\widehat{w}_k(t_a^m)$, in radians per sample:

$$\widehat{w}_k(t_a^m) = \Omega_k + \frac{1}{H_a}\Delta\Phi_k^m \qquad (3\text{-}15)$$

is then multiplied by the synthesis hopsize to calculate the phase spectrum for the synthesis frame at the time scaled output and added to the phase from the previous synthesis frame. This is known as phase unwrapping, or *horizontal phase coherence*.

$$\angle Y(t_s^m, \Omega_k) = \angle Y(t_s^{m-1}, \Omega_k) + H_s \widehat{w}_k(t_a^m) \quad (3\text{-}16)$$

The last step in the phase vocoder algorithm is to overlap and add the updated TSM phase corrected STFTs and apply the synthesis window to obtain the output using equations 3-12 and 3-13.

Phase Vocoder Artifacts and Enhancements

When evaluating output signals from the standard phase vocoder there are two notable artifacts commonly encountered: *Phasiness* and *transient smearing* [14]. Even though the time-scaled output, *y(n)*, is horizontally phase coherent between successive synthesis frames, the sonic quality of the standard phase vocoder algorithm is often described as sounding 'phasey', or 'reverberant.' Transient smearing, perceived as a loss in percussiveness in the signal, or with 'less bite,' is also commonly noticed during playback.

The horizontal phase coherence can be thought of as phase coherence *within* each k^{th}-STFT frequency bin over time. While every channel's phase is calculated, and updated, exclusive to itself, between frames there is no attempt to correct the phase relationships across STFT bins (neighboring frequencies) within a given frame. This is known as *vertical phase coherence*. By focusing solely on the horizontal phase coherence, the vertical phase coherence is dramatically altered leading to the reverberant artifacts at the output.

In [14], a technique called *identity phase locking* is introduced to mitigate this effect by identifying sinusoidal frequency bins in the STFT analysis frames through a magnitude peak picking process. The detected peak bins are considered the main sinusoidal contributions to the signal, while neighboring bins are designated non-sinusoidal bins. For each detected peak, $\widehat{w}_k(t_a^m)$ and the horizontal phase coherence is calculated, as before. Then, non-sinusoidal bins are updated by maintaining the phase difference that exists between itself and its closest peak bin. If Ω_{kP} is the center frequency of the closest peak bin to the non-sinusoidal bin frequency Ω_k, the non-sinusoidal bin's phase is locked from the analysis to synthesis stage based on its phase difference from Ω_{kP}:

$$\angle Y(t_s^m, \Omega_k) = \angle Y(t_s^m, \Omega_{kP}) - \angle X(t_a^m, \Omega_{kP}) + \angle X(t_a^m, \Omega_k) \qquad \text{(3-17)}$$

Identity phase locking introduces two significant computational advantages to the original phase vocoder. First, it relaxes previous phase-unwrapping constraints and allows for larger analysis hopsizes to be implemented. Second, only peak bins require trigonometric calculations between stages and then non-sinusoidal bins phase updates can be updated by one complex multiply. Given the angle required to rotate a peak channel (θ), one can simply use the phasor $Z = e^{j\theta}$ to calculate all neighboring, non-sinusoidal channels.

$$\theta = \angle Y(t_s^m, \Omega_{kP}) - \angle X(t_a^m, \Omega_{kP}) \qquad \text{(3-18)}$$

$$Y(t_s^m, \Omega_k) = ZX(t_a^m, \Omega_k) \qquad \text{(3-19)}$$

The identity phase locking adaptation to the original algorithm significantly improves the computational efficiency[5] of the TSM, the consistency of the synthesized output, and greatly improves TSM phase artifacts. However, it is noted in [12] that results become subjectively worse as α is decreased.

Transient smearing is a direct result of how the phase vocoder will compress, or expand, the duration of a transient event in the same manner as it would with any other audio signal. As horizontal phase coherence is updated between successive frames, the impulsive, broadband transient components are smeared during the additive synthesis stage. In addition, even with the inclusion of vertical phase coherence, the broadband components of the transients are not properly retained during synthesis. As suggested earlier for WSOLA transient preservation, a solution is to implement an additional transient detection process that updates the shifting factor $\alpha = 1$ during transient frames. Then it dynamically alters the global shifting factor to compensate for the detected, unmodified transient frames.

CONSTANT-Q TRANSFORM ALGORITHM (CQT)

The Constant-Q transform (CQT), introduced by [22] in 1991, was originally devised as an alternative approach to the STFT for spectral analysis of signals. Fourier transform methods, including the STFT, produce a linearly-spaced frequency resolution, independent of frequency. The CQT provides geometrically-spaced frequency resolution, dependent on the center frequencies (Q-factors) of the windows used in each bin. This results in larger low frequency analysis windows for increased frequency resolution, while also improvement on high frequency time resolution. Since its introduction 20 years ago,

[5] Typically, 50% less computation.

use of the CQT in signal processing applications has been limited due to its computational demands and non-invertiblity. However, recently presented methods for efficient, invertible CQT implementations, using non-stationary Gabor transforms (NSGT), have shown a fully reconstructable CQT [23]. The following section first introduces the invertible CQT by use of the NSGT. Then discusses how the CQT[6] is applied to pitch shifting and concludes by addressing CQT's known limitations and sources for future improvements towards a real-time PSM implementation.

As given by [24], a discrete-time based signal, *x(n)*, has a CQT representation *X(k,n):*

$$X(k,n) = \sum_{m=0}^{N} x(m)\, a_k^*(m-n) \quad (3\text{-}20)$$

where *k* and *n* represent frequency and time indices, respectively, *N* is the length of *x(n)*, and *atoms* $a_k^*(t)$ are the complex conjugated, modulated localization Gabor window functions [25, pg. 9]:

$$a_k = g_k(m) e^{\frac{j2\pi m f_k}{f_s}},\ m \in \mathbb{Z} \qquad (3\text{-}21)$$

with a zero-centered window function, $g_k(m)$, and a bin center frequency, f_k, that are geometrically spaced such that

$$f_k = f_0 2^{\frac{k}{b}}, k = 0,1,\dots,K-1 \qquad (3\text{-}22)$$

[6] For ease of notation, the CQT acronym will be used, but infers a NSGT is applied. (i.e. CQ-NSGT).

where b represents the number of bins per octave, f_0 is the lowest frequency analyzed, and K is the total number of bins. As proposed in [25], assuming $g_k(m)$ is symmetric about m=0, $a_k^*(m) = a_k(-m)$ equation 3.20 can be re-written as:

$$X(k,n) = [x * a_k](n) = F_N^{-1}[(F_N x)(F_N a_k)](n) \quad \text{(3-23)}$$

where * represents convolution and F_N is the N-point DFT operator. Letting $\hat{x} = F_N x$ as the N-point DFT sequence of $x(n)$, and $\hat{a}_k = F_N a_k$, the expression can further be reduced to show that the CQT coefficients can be computed by multiplication in the DFT domain (i.e. fast convolution):

$$\begin{aligned} X(k,n) = c_k(n) &= [F_N^{-1}(\hat{x}\hat{a}_k)](n) \qquad \text{(3-24)} \\ &= [F_N^{-1}(\hat{c}_k)](n) \end{aligned}$$

To efficiently evaluate $X(k,n)$, Schorkhuber [26] proposes frequency-domain sub-sampling each output, c_k(n), by only considering $n \in 0, H_k, 2H_k, \ldots \frac{N-1}{H_k}$, where H_k is the analysis hopsize for each frequency bin, k. Effectively, this can be seen as sub-sampling each output, $c_k(n)$, of the K-channel filter bank. To properly account for the CQT phase, all non-zero spectral components in the range between $-\frac{f_s}{2}:\frac{f_s}{2}$ are be mapped to the frequency range $-\frac{f_s^k}{2}:\frac{f_s^k}{2}$ by the mapping function:

$$M(f, f_s^k) = f - \left\lfloor \frac{f}{f_s^k} \right\rfloor f_s^k, \qquad \text{(3-25)}$$

where f_s^k is sub-sampling rate $f_s^k = \frac{f_s}{H_k}$ and $\lfloor * \rfloor$ denotes rounding towards negative infinity. Figure 3.10, below, shows the sampled CQT bins in the time-frequency domain.

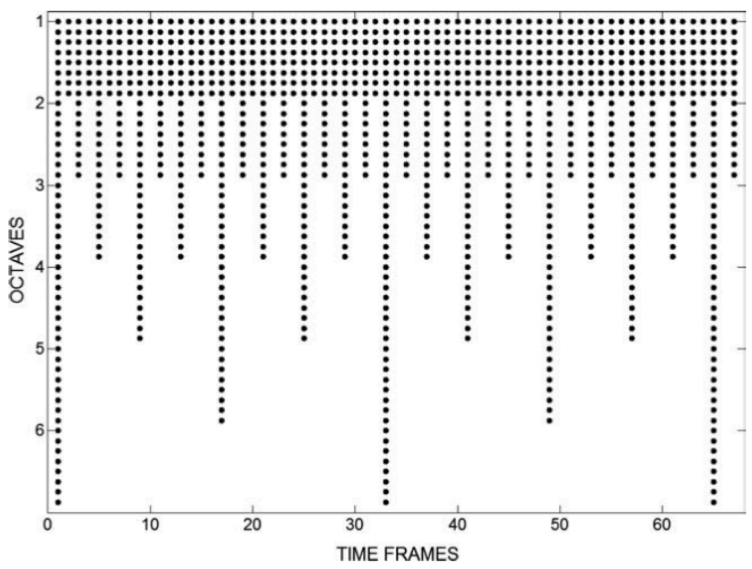

Figure 3.9: CQT Bins on Time-Frequency Plane [26].

It has been proven the transform yields perfect reconstruction of *x(n)* through use of *dual frames* $\hat{a}_k$ for synthesis atoms [25, pg. 7]. This is referred to as the *painless* case and requires the input signal to be transformed in its entirety before analysis-synthesis. In practice, the CQT analysis and synthesis stages are computed through use of the FFT, and IFFT, but given the geometric frequency spacing of the transform a synthesis OLA application of $\hat{a}_k$ is necessary for reconstruction.

CQT Pitch-scaling

In [26], a CQT frequency-domain pitch shifting algorithm is presented by application of a shifting factor, α, to the CQT bins. For the CQT resolution, b (equation 3.23), the shift in CQT bins is given by $r = b * \log(\alpha)$, where r is independent of the frequency spectral peak. For example, shifting a bandlimited 2.5 kHz to 5 kHz signal that has CQT bin resolution $b = 6$ up one octave (5 kHz to 10 kHz) would require $r = 6$ bin shifts ($\alpha = 2$). To achieve proper CQT pitch shifting, time alignment is an important factor in implementation. CQTs that have a common subsampling factor for all frequency bins are realized by subsampling all bins at the highest frequency channel. As seen in Figure 3.10a, the minimally redundant fully-invertible CQT has atom hopsizes, H_k, that are dependent on varying window lengths, N_k, per frequency channel and cannot be shifted along the frequency axis without changing their position in time. The overly subsampled (fully rasterized) CQT (Figure 3.11b) bins are temporally aligned allowing frequency components to retain their relationships when a shift is applied.

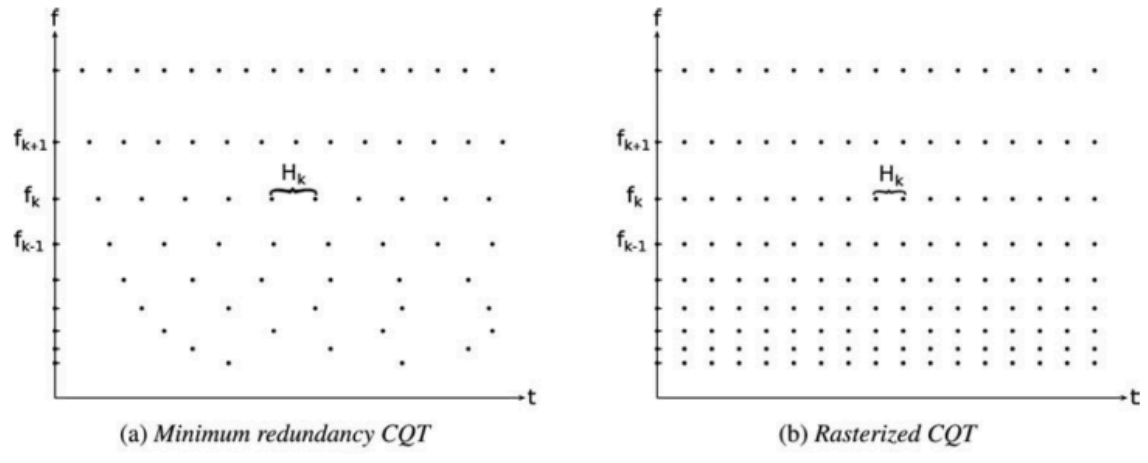

Figure 3.10: Comparison of CQT Time-Frequency Sampling Schemes [26].

To maintain phase coherence, the CQT algorithm performs a vertical and horizontal phase coherence update stage like the improved FFT phase vocoder. However, given the geometric frequency-bin spacing, an approximation is presented in [26, pg. 569] to calculate the instantaneous frequency for the horizontal phase update. In relating notation between the two algorithms: Let $f_1 = \widehat{w}_k(t_a^m)$ and $f_2 = \widehat{w}_k(t_s^m)$, the instantaneous frequency before and after modification, respectively. Let $\hat{f}_1 = \Omega_{kPa}$ and $\hat{f}_2 = \Omega_{kPs}$, the peak-detected center frequency bins before and after modification, respectively. Let $\Delta\phi_1 = \angle X(t_a^m, \Omega_{kP})$ and $\Delta\phi_2 = \angle Y(t_a^m, \Omega_{kP})$, the calculated unwrapped phase before and after modification, respectively. In this case let $\Phi_{CQT} = \theta$, the angle required to rotate the peak detected bin. To correctly perform the horizontal update between successive frames the current frequency shift $\Delta f = f_2 - f_1$ is needed to compute Φ_{CQT}.

$$\Phi_{CQT} = \Delta\phi_2 - \Delta\phi_1 = \frac{2\pi R}{f_s}\Delta f \qquad \text{(3-26)}$$

However, given the geometric frequency-bin spacing of the CQT,

$$\Delta f = f_2 - f_1 = f_1\left(2^{\frac{r}{B}} - 1\right) \neq \hat{f}_2 - \hat{f}_1 \qquad \text{(3-27)}$$

and therefore, an approximation using $\hat{f}_1 \approx f_1$ is substituted as shown:

$$\Phi_{CQT} \approx \frac{2\pi R}{f_s}\widehat{f}_1(2^{\frac{r}{B}} - 1) \qquad \text{(3-28)}$$

Listening tests have shown this approximation minimally alters the signal through slight frequency and amplitude modulations, but is recommended for its computational efficiency.

CQT Limitations and Enhancements

Given its relatively short time span as a PSM algorithm and promising geometric frequency bin resolution, there are still notable limitations for the CQT. The most significant one is that the current algorithm cannot support real-time operation. In addition, the CQT PSM also exhibits vertical phase-based transient smearing similar phase vocoder behavior. These limitations and proposed future solutions are discussed below.

To perform a perfectly reconstructed *x(n)* using dual NSGT atom frames, $\hat{a}_k$, the CQT implementation inherently requires an FFT of the entire input signal prior to processing. As such, the CQT algorithm utilized in the experiment cannot support time-frame based processing for PSM. However, in 2012 a time-frame based CQT called *sliCQT* was proposed [27] where blocks of the input signal can be transformed, modified, and overlapped in the same manner as TSM. The analysis windows must be carefully chosen to reduce spectral leakage, and each slice requires zero-padding to maintain temporal alignment prior to processing.

Also, like the phase vocoder, the CQT PSM suffers from transient smearing due to loss of the vertical phase coherence across bins due to impulsive transient signals. The increased time resolution due to the CQT bin spacing helps mitigate transient smearing as compared to the phase vocoder's linear bin spacing. A further solution to the problem is to utilize a separate transient detection algorithm as mentioned previously.

Chapter 4: Experimental Methodology

In order to properly evaluate the investigated pitch shifting methods for HF passive sonar, a common experimental model is required. In addition, a set of defined input signals and implementation methodology for each algorithm is presented. This chapter first outlines the pitch shifted, HF passive sonar signal chain utilized in the experiment, including the modeled HF sonar input signal, pitch shifting procedure, and shared pitch shifting parameters for each algorithm. Then, the synthetically generated and real sonar input signals are introduced and rationalized, followed by the use and modifications to the pitch shifting algorithm resources utilized in the experiment.

EXPERIMENTAL MODEL

Before conducting the ultrasonic sonar pitch shifting experiment, a common model is required to define the shared parameters applied to each PSM algorithm. While each one is implemented in various domains and phase correction techniques, the parameters of the input signal, the underlying analysis-synthesis pitch shifting process (Chapter 3), and intended output signal remains the same for each case. In this section, a discrete-time sonar input signal is defined, based off of known HF sonar systems [28-30]. The experimental TSM pitch shifting process, fundamental to both the WSOLA and phase vocoder methods, is presented, followed by the analysis-synthesis CQT pitch shifting implementation. Finally, the pitch shifting parameters and experimental output signal is discussed, including insight into why specific experimental parameters are chosen.

As presented in Chapter 2, the output of the sonar beamformer is a complex-basebanded and downsampled discrete-time signal. To convert to audio, we first upsample, filter, and unbaseband back to a real signal (i.e. not complex). Based on the frequency ranges of referenced sonar systems [28-30] and in context to the experiment, our input signal model (Figure 4.1) will have a center frequency (f_c) of 20 kHz with a 30 kHz bandwidth ($B, 20 \pm 15\ kHz$) and a sampling rate of 125 kHz (f_s). The choice in a HF sonar signal centered at 20 kHz is suitable for our experiment for two reasons: (1) It contains a broad frequency range (~ 2.75 octaves) to map into the auditory spectrum, and (2) is computationally less demanding to shift towards the center range of human hearing compared to theoretical high-frequency sonars with $f_c > 20$ kHz.

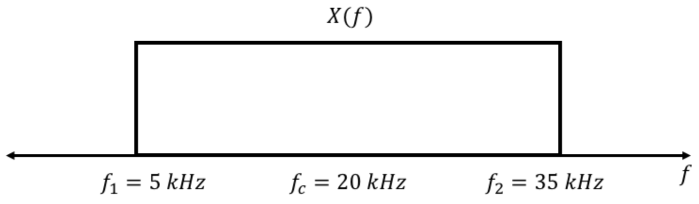

Figure 4.1: Frequency Spectrum of Experimental Sonar Signal.

While all three algorithms perform pitch shifting through various domains and modification techniques, the fundamental TSM process applied to the experimental sonar signal is common between the WSOLA and phase vocoder implementations (Figure 4.2). That is, the input signal will undergo a time sample frame-based analysis-synthesis process with output, *v(n)*, the time-scale modified signal of *x(n)*. The TSM signal is then interpolated back into its original time-scale, but retains the intended modified pitch (*y'(n)*). In the case of CQT, *x(n)* is transformed in its entirety, the CQT frequency bins are shifted to the desired pitch, and the inverse transform is applied to produce *y'(n)*. Interpolation is not necessary since the inverse transformed signal retains the same length as *x(n)*. Finally, each experimental method is resampled from f_s *= 125 kHz to 48 kHz* through an upsample, FIR filter, then downsample process for proper audio playback.

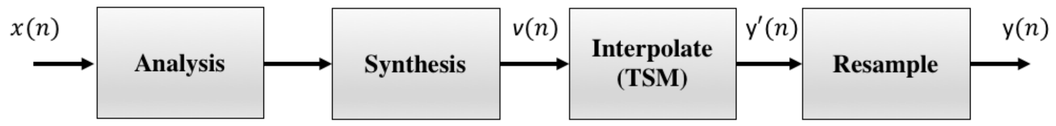

Figure 4.2: Flowchart of the Experimental Pitch Shift Process.

A pitch shift from $f_c^x = 20\ kHz$ to $f_c^y = 2\ kHz$, stretching factor $\alpha = \frac{f_c^y}{f_c^x} = \frac{2\ kHz}{20\ kHz} = 0.1$, is applied to each experiment (Figure 4.3). The decision to set $\alpha = 0.1$ and center the output at 2 kHz is based on multiple factors. First, the use of a single-decade downward shift simplifies the math, both conceptually and computationally, in the pitch shifting process. Secondly, from an auditory perception standpoint, centering at 2 kHz is advantageous since human hearing is most sensitive from 2 to 5 kHz [31], while tonal presence and separation is achievable in the two octaves below f_c (i.e. 500 Hz to 2 kHz). Given the logarithmic perception of pitch, it is conceivable the compressed harmonic relationships in the shifted output signal could benefit from a lower f_c. However, given the substantial shift required in the experiment, 2 kHz is a modest design compromise.

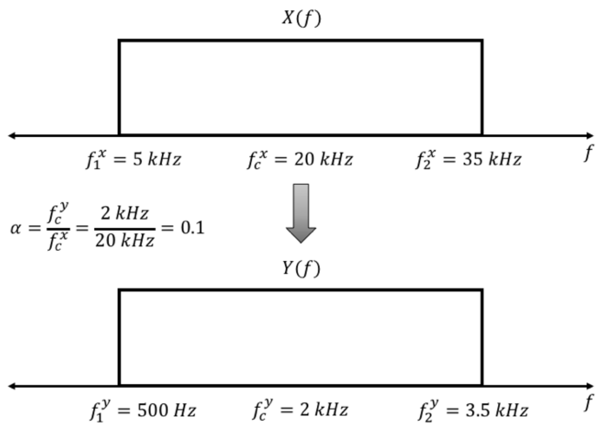

Figure 4.3: Frequency Spectrum Pitch-Shift Model.

Given our input signal model and defined pitch shifting parameters, Figure 4.4 below outlines the experimental TSM analysis-synthesis process. As presented in Chapter 3, TSM is a frame-based process dependent on the desired stretching factor α and hopfactors H_a and H_s, where $\alpha = \frac{H_s}{H_a}$. A frame time-length of 20 milliseconds (ms), or $N = 2500$ samples ($f_s * 20\ ms$), is chosen since it encompasses the lowest fundamental 5 kHz frequency content of the signal (100 cycles/20 ms = 5 kHz). As stated in [14], H_s is held constant for a 75% overlap, or $H_s = \frac{N}{4} = 625$ samples. Given $\alpha = 0.1$, we solve $H_a = \frac{H_s}{\alpha} = 6{,}250$ samples. For example, frames x_1 and x_2 are obtained based on N and H_a, windowed to reduce spectral leakage, then analyzed for modification. After modification, x_1 and x_2 result in y_1 and y_2, respectively, which are then windowed at synthesis for proper amplitude adjustment. The 75% overlap factor, H_s, is applied and y_1 and y_2 are summed together to form *v(n)*. The TSM signal is interpolated up to $\frac{f_s}{\alpha}$ (1.125 MHz), and the resulting output, *y(n)*, is the time-aligned, pitch shifted version of *x(n)*.

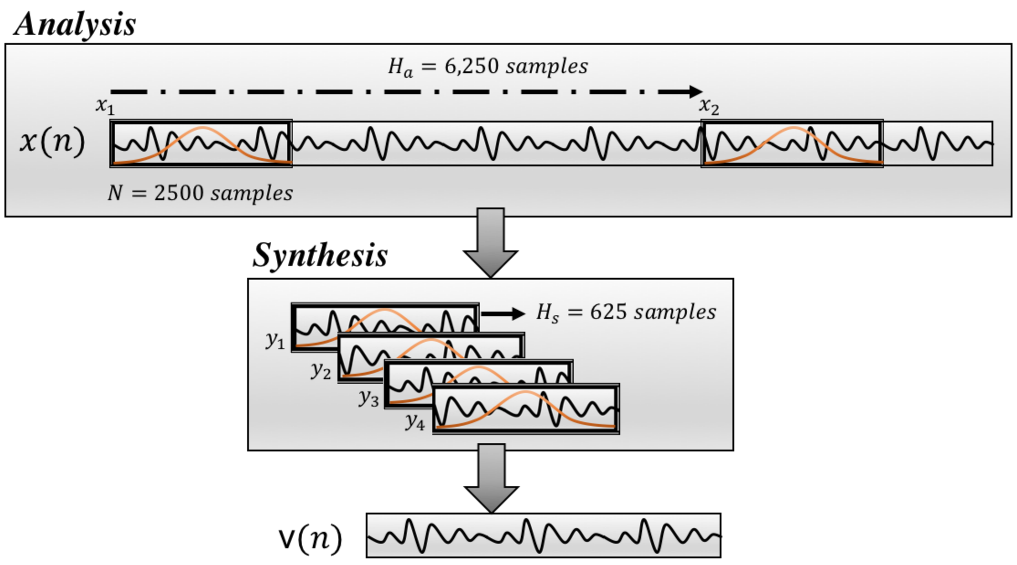

Figure 4.4: TSM Analysis-Synthesis Procedure.[7]

The use of the non-stationary Gabor frames in the CQT method enforces that *x(n)* is retained in its entirety during the analysis-synthesis process (Chapter 3). Temporal alignment of the transformed signal is retained by sampling *x(n)* at the highest required frequency f_2, ensuring the CQT coefficients (frequency bins) represent the entire signal (Figure 4.5). The frequency bins are shifted down to the desired pitch, vertical and horizontal phase adjustments are performed, then the inverse CQT is applied to generate the pitch modified output signal *y(n)*.

[7] The time-domain process is shown, but the frequency-domain process is the same using STFT frames.

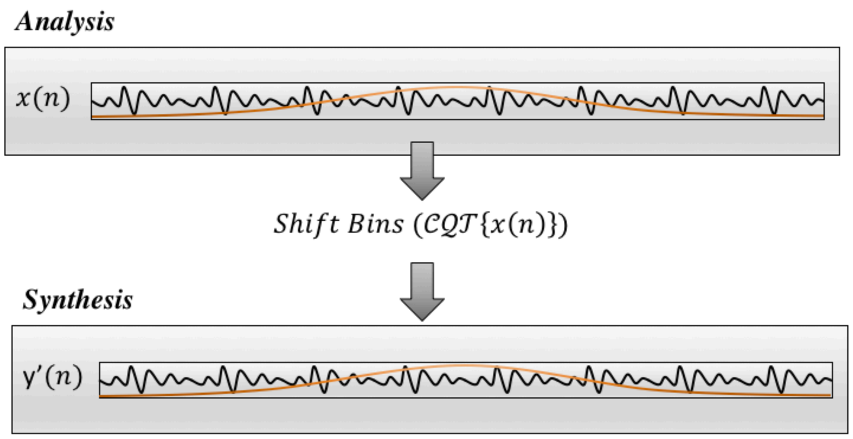

Figure 4.5: CQT Analysis-Synthesis Procedure.

Synthetic and Sonar Input Signals

The conducted experiment utilizes a combination of both synthetic and real sonar input signals. Synthetic data is first generated and processed to gain an understanding into the behavior and artifacts of each pitch shifting method. In this section, the types of synthetic data, the rationale behind their use in the experiment, and the process to create them are shown. Then, sonar data recorded in the field and tuned to the model is introduced, along with a discussion on the beamformed input signal and the targets included in the experimental data.

Three categories of synthetic data are generated for the experiment: In-band noise, linear chirps, and a mock sonar signal consisting of chirps, noise, and pure sinusoidal tones. As presented in Chapter 2, self, ambient and thermal noise are all present in the experiment's band of interest. (See Figure 2.3) The inclusion of white noise (Figure 4.6), and how each algorithm interprets signal in relation to it, is important in modeling the real sonar signal. In the case of linear chirps, both manmade (i.e. sonar pings) and biological signals produce impulse-likeive-like, wideband chirps within our model's frequency range. Given the large analysis hopsize required in the experiment's TSM, it is of interest to see how each algorithm responds to both slowly-varying (5 seconds, Figure 4.7) and short, impulse-likeive-like (20 and 100 milliseconds) chirp signals.

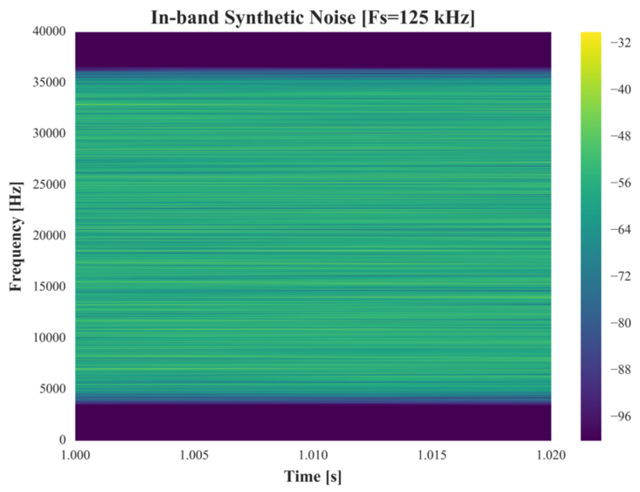

Figure 4.6: In-Band Synthetic Noise Spectrogram.

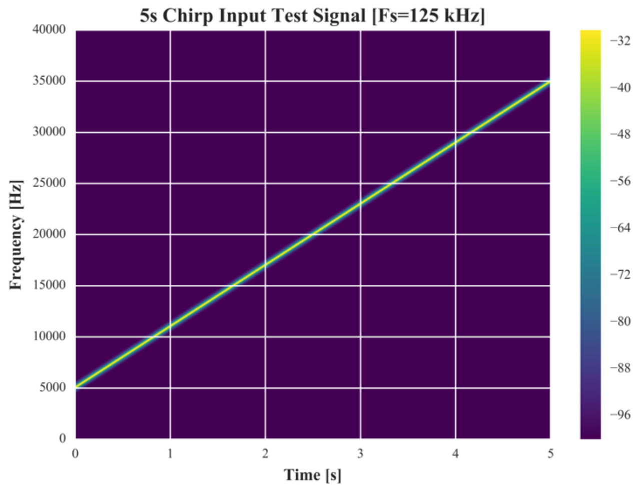

Figure 4.7: Five Second Chirp Signal Spectrogram.

To model a synthetic sonar signal, a combination of white noise, 20 and 100 millisecond chirps, and sinusoidal tones were summed together as seen in Figure 4.8, below. From 0 to 5 seconds, 20 millisecond chirps are set at integer time values (i.e. 0,1,2…5 seconds) similarly followed by 100 millisecond chirps from 6 to 9 seconds. Two tones are used at 10 and 20 kHz, with the first 10 kHz tone playing from 1.5 to 3.5 seconds, then again from 4.5 to 6.5 seconds. The 20 kHz tone is set from 5.5 to 7.5 seconds and the noise level is set to be ~9 dB below the chirp and tone signal level.

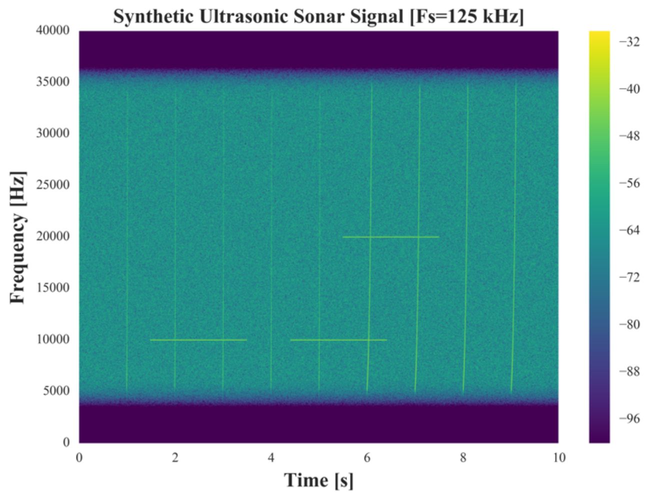

Figure 4.8: Synthetic Sonar Signal Spectrogram.

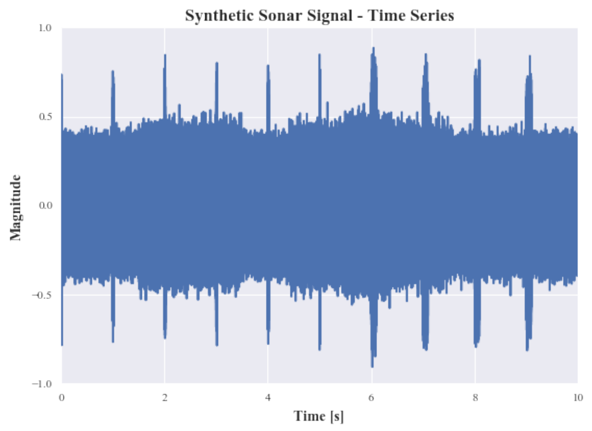

Figure 4.9: Synthetic Sonar Signal Time Plot.

All synthetic data was created using Audacity [32] at a 125 kHz sampling rate and saved as output in 32-bit floating-point format. In the case of the white noise and synthetic sonar signal, an additional 128-tap Hamming FIR filter (Figure 4.10) was implemented in Python to remove the out of band signal prior to applying the pitch shifting algorithms.

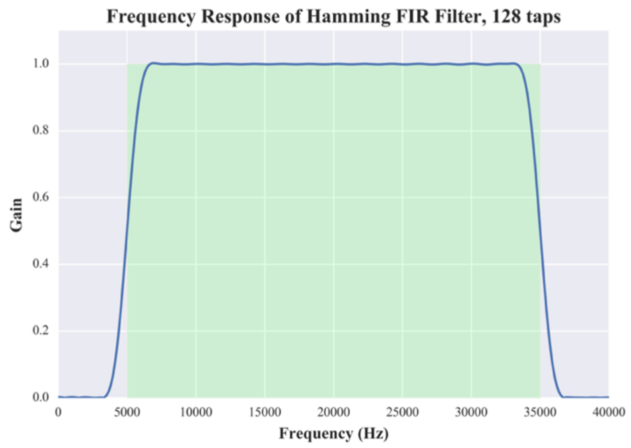

Figure 4.10: FIR-Filter for Synthetic Data.

In addition to synthetic data, field measurements of passive sonar data were collected using a sonar system tuned to meet the specified experimental model. Figures

4.11 – 4.13 below are time-series waterfall plots of the passive sonar data showing three identified targets through a peak detection imaging algorithm. Target A is a jon boat with an outboard motor, target B is a constant ping reflection of a solid underwater boundary, and target is C is a single-person jet ski. Beams with the most significant peak-identified signal (graphically shown in red) are chosen and the unbaseband, resampled beam input signal is in 32-bit floating point format.

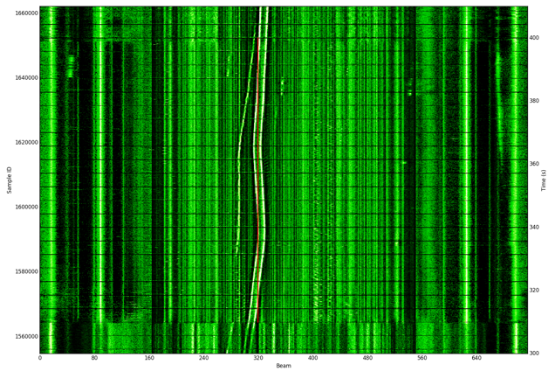

Figure 4.11: Target A Waterfall Plot.

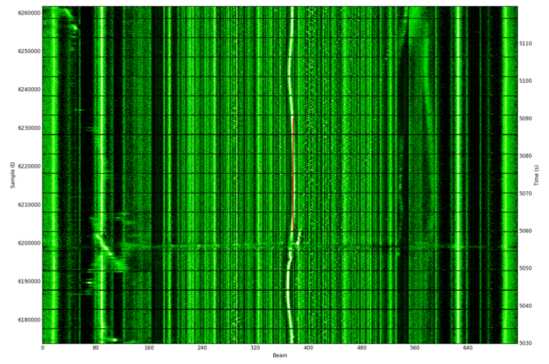

Figure 4.12: Target B Waterfall Plot.

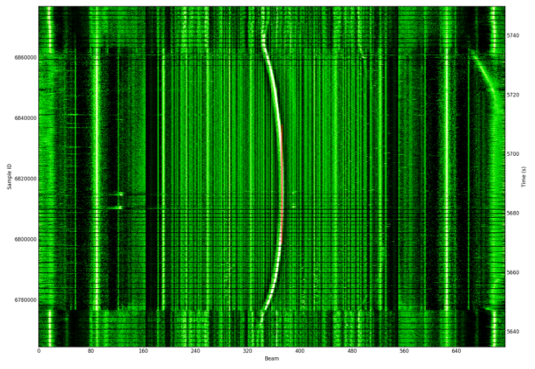

Figure 4.13: Target C Waterfall Plot.

ALGORITHM IMPLEMENTATION

To implement the pitch shifting algorithms, three separate resources are utilized in the experiment. The main criteria for their use in the experiment are that they are: open source and easily accessible, well documented and referenced, and adaptable to meet the experiment model. The synthetic and real sonar signals are processed by each algorithm, then the resulting output is analyzed in Python and resampled for audio playback (Figure 4.14). The following section presents the C-based *Soundtouch* library [33] for WSOLA, followed by the Python *Librosa* library [34] for the phase vocoder, and finally the *Matlab toolbox for CQT* [35] pitch shifting.

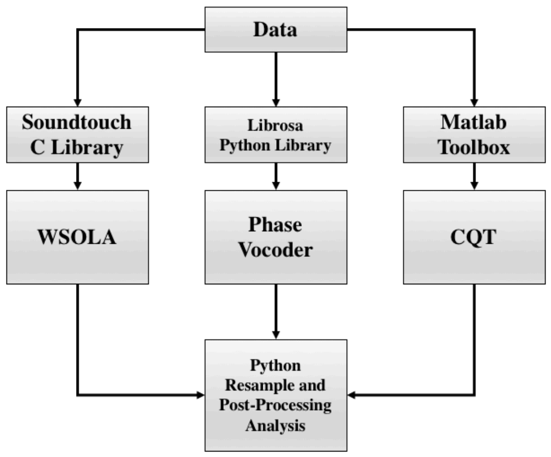

Figure 4.14: Flowchart of Experimental Process.

The C-based *Soundtouch* library utilizes the time-based WSOLA algorithm (Chapter 3) and linear interpolation with anti-alias filtering to perform pitch shifting. Parameters are modified in the source code (TDStretch.h) to change the frame window length (*N*), WSOLA tolerance ($\pm\Delta$), and synthesis hopfactor H_s. For the experiment, *N* and H_s are set to the model's specifications (2500 samples and 75% overlap), while $\pm\Delta$ is kept at the library's default setting. A terminal command-line utility allows users to modify an input signal with a pitch-scale modification factor and saves a modified output file. It is important to note that *Soundtouch* does not employ additional transient detection in its implementation. Therefore, potential transient doubling and/or skipping artifacts may be exhibited in the experimental results.

The Python-based *Librosa* library is an audio analysis package that includes a phase vocoder *identity phase locking* (see Chapter 3) implementation and non-linear (Kaiser window) interpolation to produce the pitch shifted output signal, *y'(n)*. STFT (frequency bins) are shifted by $\alpha = 0.1$ and modified with vertical and horizontal phase updates before windowed synthesis. The STFT bin resolution is set to 48/octave, *N* is set to 2500 samples, and the synthesis overlap setting is set to the 75% (625 samples) experimental model, by default. *Librosa* does not utilize transient detection in its implementation, therefore, phase vocoder transient smearing artifacts may be exhibited in the experimental results.

The *Matlab* CQT toolbox is provided by [35] as a resource to research the audio signal applications of the recently realized invertible CQT transform. As noted before, the current toolbox does not utilize time-based N frame input signals for a real-time implementation (i.e. the input signal is processed in its entirety). CQT bins are shifted by $\alpha = 0.1$ and bin resolution is set to 48/octave, consistent with the phase vocoder. The toolbox does not have an additional transient detection algorithm so frequency-based transient smearing artifacts are possible.

Chapter 5: Experimental Results

In this chapter the experimental passive sonar PSM results are shown by following the methodology discussed in Chapter 4. First, processed synthetic data is presented and discussed, followed by the real sonar signals. Finally, the three pitch shifting algorithms are compared based on their performance and ability for real-time implementation. Synthetic results include the theoretical output, $y_{th}(n)$, as a basis for comparison, and the pitch shifting algorithms are evaluated based on their output relative to each other and $y_{th}(n)$. Previous discussion on potential artifacts add insight into each algorithm's response to the synthetic and sonar input signals. A comparison of the algorithms is shown using metrics including transient and harmonic preservation, input signal distortion, and computational complexity. All results are post-processed[8] and graphically presented using Python. A supplemental website (https://atreptow.gitlab.io/psm_passive_sonar/) is provided for readers to listen to all the processed audio data presented in this chapter.

SYNTHETIC DATA RESULTS

The synthetic results are presented in side by side comparison, including $y_{th}(n)$, for each linear chirp input signal (5 seconds, delayed 100 milliseconds, and 20 millisecond train). Next, a spectral comparison on how each algorithm responds to random in-band noise is discussed. Finally, a comparison using the synthetic sonar signal is shown.

The five second linear chirp results are shown in Figure 5.1. Overall, each method can retain the structure of the slowly time-varying chirp relatively well, however both the phase vocoder and CQT algorithms exhibit vertical phase 'noise', as seen as vertical

[8] Resampling the PSM signal $y'(n)$ to $y(n)$ (i.e. 125 kHz to 48 kHz) through poly-phase filter bank.

magnitude smearing. WSOLA performs the best given the overlapped 10 millisecond frame windows are long enough to carry the fundamental frequency in the signal and the algorithm analysis frames can successfully retain waveform similarity. The CQT response exhibits phase offset complications on the lower end of the spectrum (seen as spreading in the chirp tail), vertical phase smearing as the frequency increases, and slight spectral leakage at both the beginning and end of the time-series. The phase vocoder contains vertical phase smearing throughout all frequencies, observed as the unnatural *phasiness* artifact as discussed in Chapter 3, and is the least desirable of the three algorithms in this case.

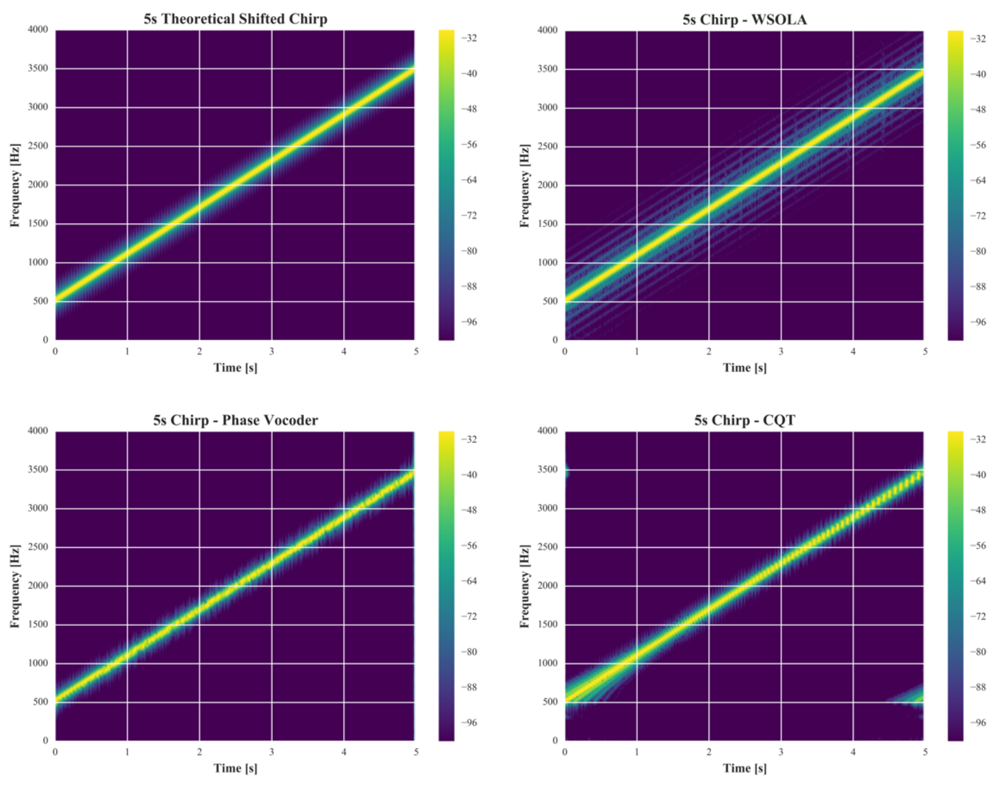

Figure 5.1: Spectrograms of 5 s Linear Chirps.

The results of the delayed 100 millisecond chirp (Figure 5.2) begin to show each method's transient response, including transient smearing and aliasing. While WSOLA follows the overall chirp trend and retains the input signal's length, there are noticeable gaps and additional spectral information throughout. At 100 milliseconds, the input signal is too fast for the analysis tolerance autocorrelation to produce waveform similarity at the frame boundaries in the same manner as before. Without waveform similarity at each of the analysis overlap regions the resulting output exhibits significant aliasing. The phase vocoder stretches the signal by ~100%, where the overall chirp tracking is the least favorable of the three, and exhibits significant transient smearing. Like the WSOLA, the phase vocoder's hopsize is too large to keep up with the fast-time varying chirp. During the phase update process, the peak bin shifts between frames do not contain similar sinusoids, and both horizontal and vertical phase is not maintained. The CQT response follows the 100 milliseconds chirp the most accurately out of the three. There are vertical phase artifacts throughout, and similar spectral leakage artifacts as exhibited in the 5 second chirp, but the overall signal mostly retained.

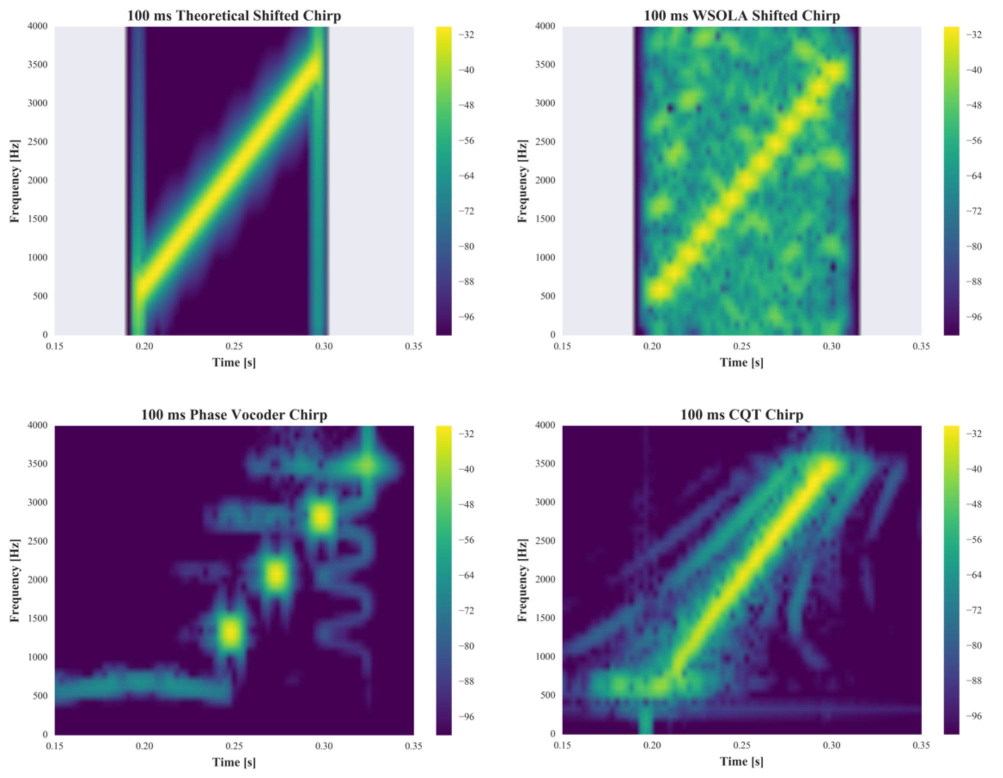

Figure 5.2: Spectrograms of 100 ms Delayed Linear Chirps.

The 20 millisecond chirp train results (Figure 5.3) follow closely with the previous delayed 100 millisecond chirp however, transient smearing and artifacts are intensified for each algorithm. With an even shorter input signal, the WSOLA algorithm cannot keep up with the changing frequency per analysis hopsize. The resulting signal lumps small, equally spaced portions of the frequency content and again, exhibits spectral leakage. The phase vocoder again produces transient smearing, but in this case, the output signal alters pitch (+1 kHz per chirp) then repeats after reaching f_2. The CQT contains the most in-band frequency content of the input signal, but has more aliased, out of band noise associated with each chirp. In addition, it contains a noticeable lower frequency tonal artifact and has imperfect reconstruction of the linear chirp within the band.

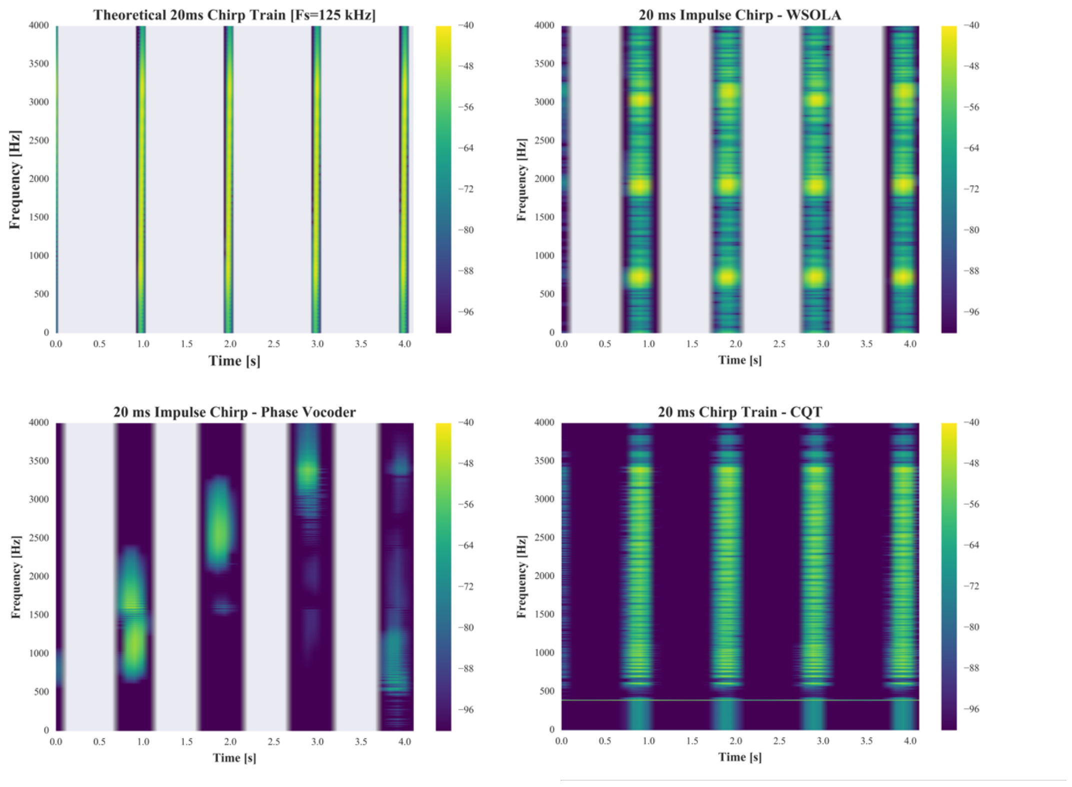

Figure 5.3: Spectrograms of 20 ms Impulse Chirp Train.

Figure 5.4 shows the time-series of a single 20 millisecond chirp response for each algorithm in comparison to the theoretical output. As seen in the figure, the WSOLA algorithm fails to follow the input since the successive analysis windows do not have similar frequencies. The paired frame edges have discontinuities and result in non-sinusoidal waveforms that add undesired frequency content throughout. The phase vocoder FFT implementation holds the starting sinusoidal frequency of the signal and again, the adjacent frames are spaced too far apart for proper frequency updates. The result is a single, decaying sinusoid determined by the dominant frequency bin per analysis window. The

CQT waveform retains the length of the 20 ms chirp, however, there is significant aliasing and additive noise present in the output.

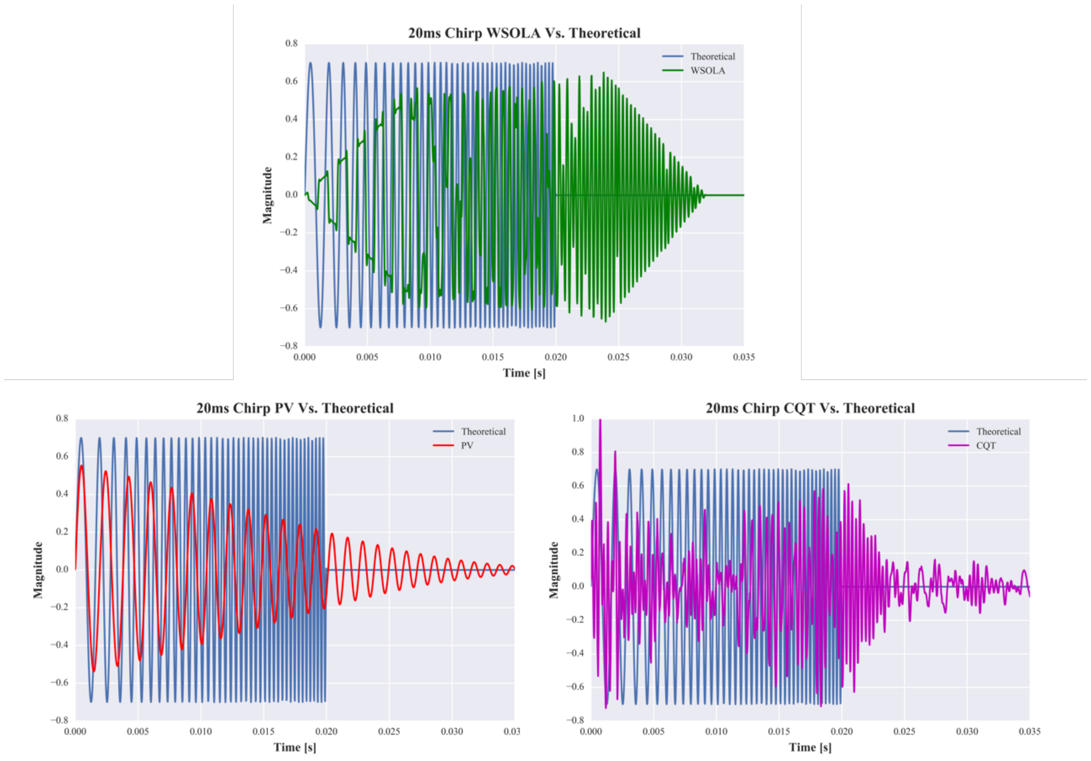

Figure 5.4: Time-domain Comparisons of Single 20 ms Chirp.

Figure 5.5 is a frequency spectrum comparison of how each algorithm handles pure white noise as an input. Here, the WSOLA algorithm is problematic given that there is never similarity in the incoming signal. The result is not only an increase in the overall in-band noise level, but additional out-of-band noise is added since the algorithm can never correlate between analysis frames, thus adding discontinuities and aliasing at all frequencies. Both the phase vocoder and CQT exhibit a decrease in the overall noise output, however, the CQT contains a noticeable low frequency out-of-band peak.

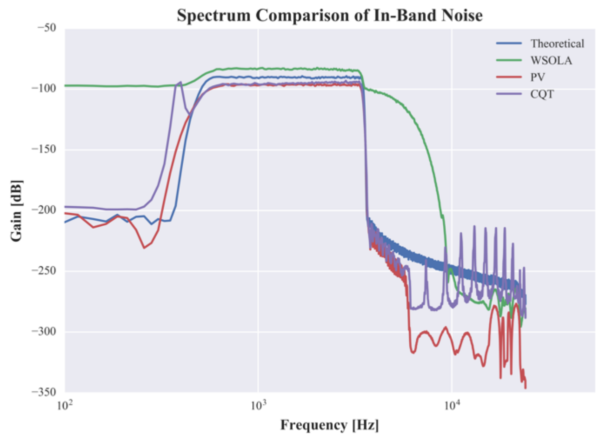

Figure 5.5: Shifted In-Band Noise Comparison.

Finally, the synthetic sonar signal (Figure 5.6) shows how each algorithm handles the combination of chirps, pure tones, and white noise. Given the WSOLA algorithm introduces out-of-band noise, a bandpass FIR filter is added after pitch shifting. The WSOLA method adds additional noise into the in-band spectrum, however, the pure tone signal content is retained the best out of the three methods. The 20 ms chirps are reduced to tonal bursts (seen at ~500 and ~1200 Hz), while the subsequent 100 ms chirps are closer to their theoretical output, as seen before in Figure 5.2. The phase vocoder produces 20 ms chirp frequency hopping artifacts and 100 ms chirp tonal bursts, as noted before. It also retains the pure tone signal content of the input signal. The in-band noise level is less than that of the WSOLA method and closest of the three to the expected output. Finally, while the CQT reconstructs the 100 ms chirps the best, the 20 ms chirps are smeared and reduced to the lower one-third of the original frequency content. The other two-thirds are either not captured, or lost to noise. Most noticeably, the CQT is not able to resolve the pure tone inputs when additive noise is present in the experimental synthetic sonar signal.

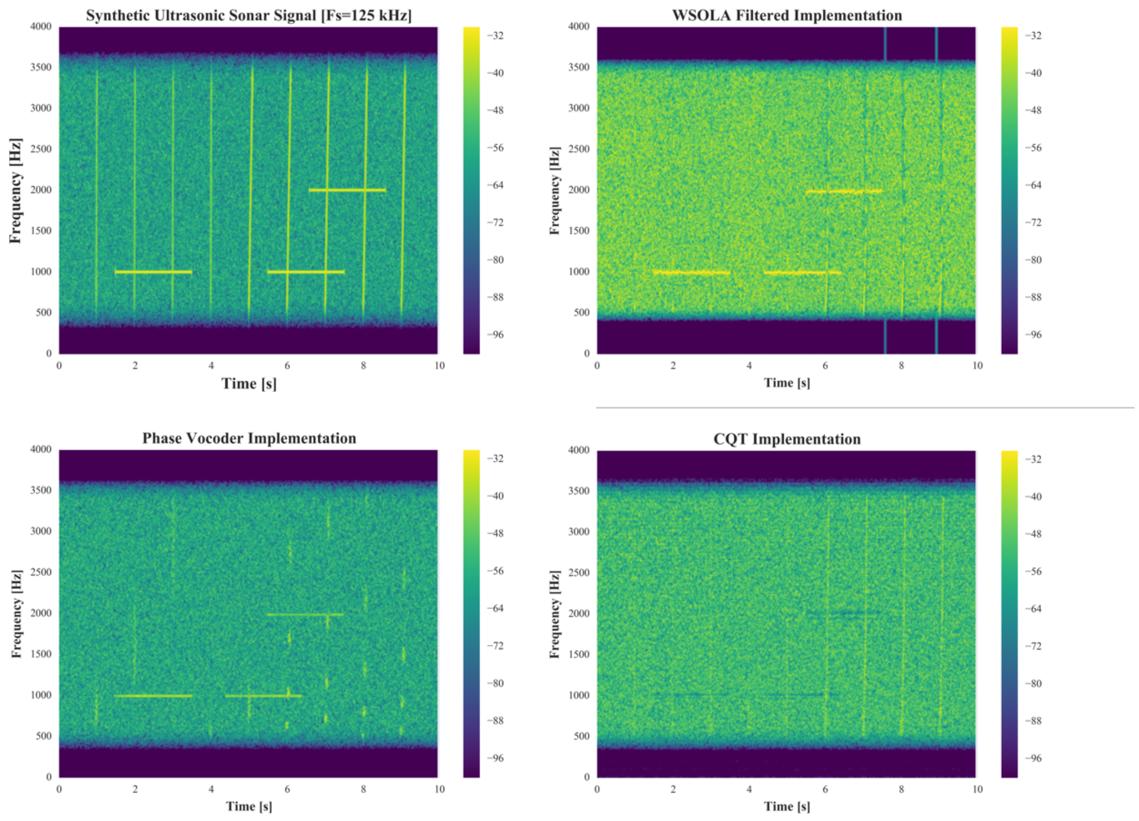

Figure 5.6: Comparison of Shifted Synthetic Sonar Signal.

Sonar Data Results

With a better understanding of how each pitch shifting algorithm responds to the synthetic input signals, we now investigate and compare them using recorded sonar signals from the field. Three sonar signals with known targets are applied: an outboard motor on a jon boat (Target A), the transmission and reflections of an active sonar ping (Target B), and a fast-moving jet ski (Target C). Figures include a side-by-side spectrogram comparison of each algorithm including the original, pre-shifted sonar signal. Please note that active sonar pings are present in all the data and self-noise of the sonar system used in the experiment is prevalent at both 18 kHz and within 27 to 30 kHz (1.8 kHz and 2.7 - 3 kHz, after pitch shifting).

Target A is of an outboard motor generating acoustic energy underwater. It's in-band content can be seen in the sonar spectrogram as short broadband bursts, most notably from 11 to 23 seconds (Figure 5.7). The WSOLA algorithm's intolerance to noise overcomes target signal, leaving only the sonar's self-noise and occasional ping. The resulting audio signal is not pleasant on the ears. Comparatively, the phase vocoder shows an improvement in a higher signal-to-noise ratio and detected ping events. While the short target bursts can be perceived on playback, they suffer from transient spreading at output. This can be visualized as a smearing of the target signal compared to the input. The CQT signal exhibits the best SNR and reconstruction of the active pings. Again, the target's signal suffers from transient spreading artifacts, but it can be heard and is more sonically defined compared to the phase vocoder.

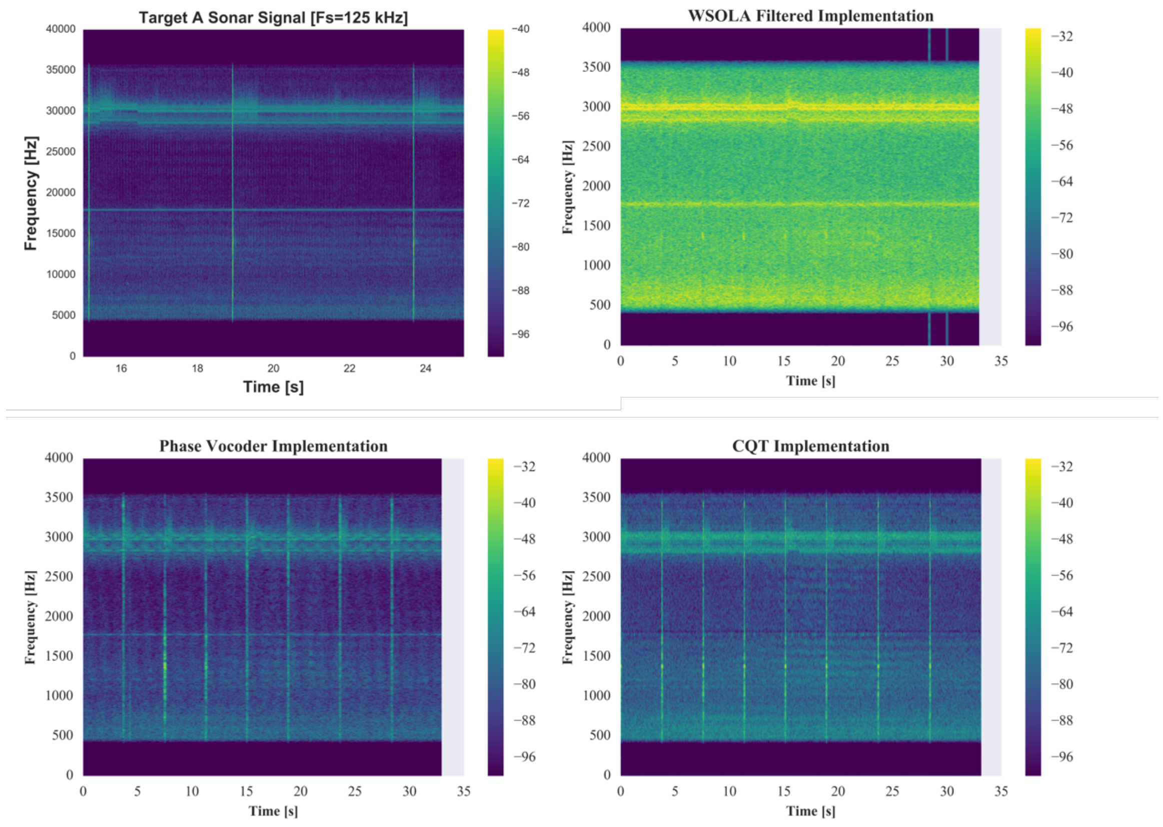

Figure 5.7: Comparison of Outboard Motor Recording (Target A).

Target B is a reflection of a sonar ping off an underwater boundary. The reflections can be seen following each ping in the original sonar signal in Figure 5.8. Again, the WSOLA's output is inundated with noise due to the lack of periodicity in the input signal. The initial pings, and their reflections, are neither visually or aurally perceptible in the results. The phase vocoder algorithm resolves the initial ping with partial reconstruction of the original signal, like its results with synthetic short-time chirp signals. As seen in the figure, and listening upon playback, the phase vocoder occasionally reproduces the boundary reflection. However, the added attenuation reduces the ability for the algorithm to successfully track the target consistently. While the CQT algorithm is still not able to retain the harmonic structure of the initial ping, its overall reproduced bandwidth and presence is greater than both the phase vocoder and WSOLA algorithm. It consistently tracks the presence of the reflection to produce a temporal signal of the event. In addition, it does the best job at handling the in-band noise in the signal.

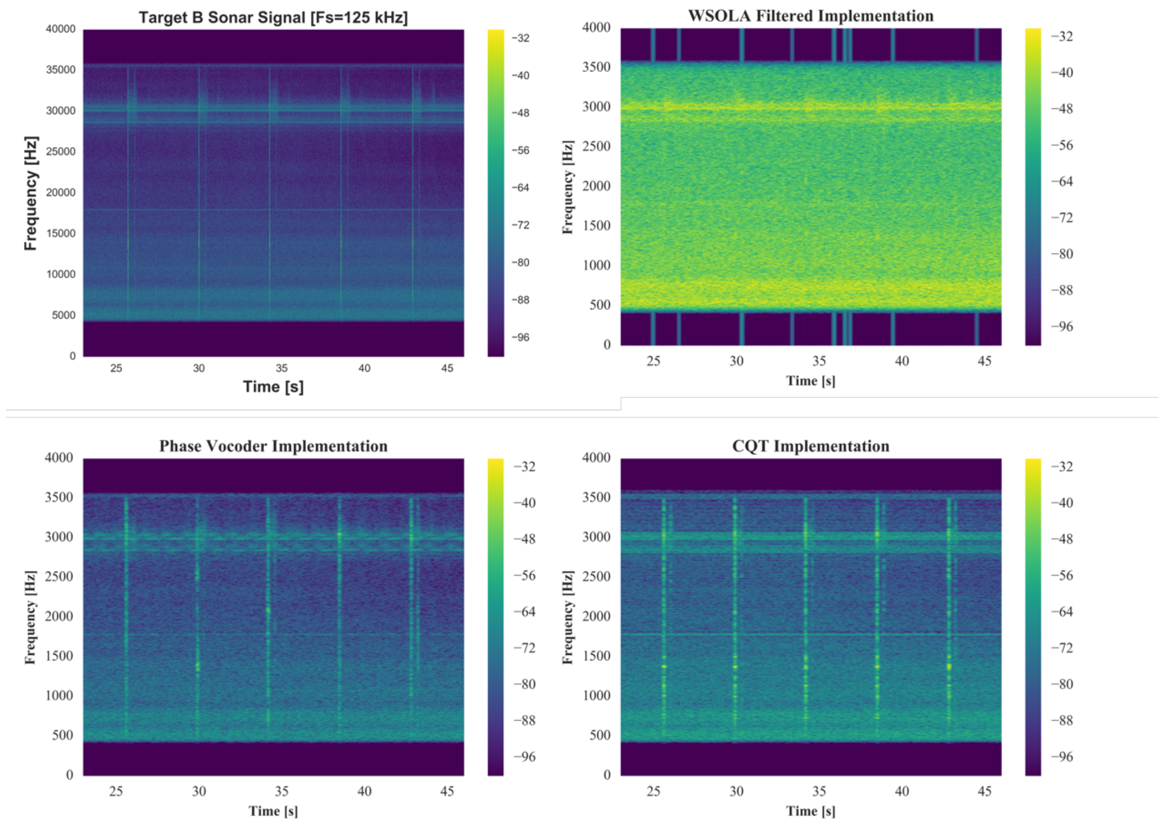

Figure 5.8: Comparison of Ping Reflection (Target B).

The last target investigated is a fast-moving jet ski. Given the relative speed of the target to the sonar, and the small aperture of the sonar beam, the recorded event is ~5 seconds in length. The original sonar signal in Figure 5.9 shows a parallel group of tones as the jet ski approaches, followed by a prominent frequency sweep from 12 kHz to 6 kHz as the target passes by the sonar. Finally, a broadband resonant frequency event occurs as the jet ski leaves the beam's range. Given the increased tonal presence in the signal, the

WSOLA algorithm can track the initial incoming target and downward frequency sweep as it passes by. However, the presence of noise is still apparent throughout and is detrimental in the performance on playback. Both the phase vocoder and CQT algorithms capture the initial tonal and broadband pass of the target well. While the CQT does a better job in noise reduction, the phase vocoder is more consistent in retaining the tonal events and sounds more natural upon playback.

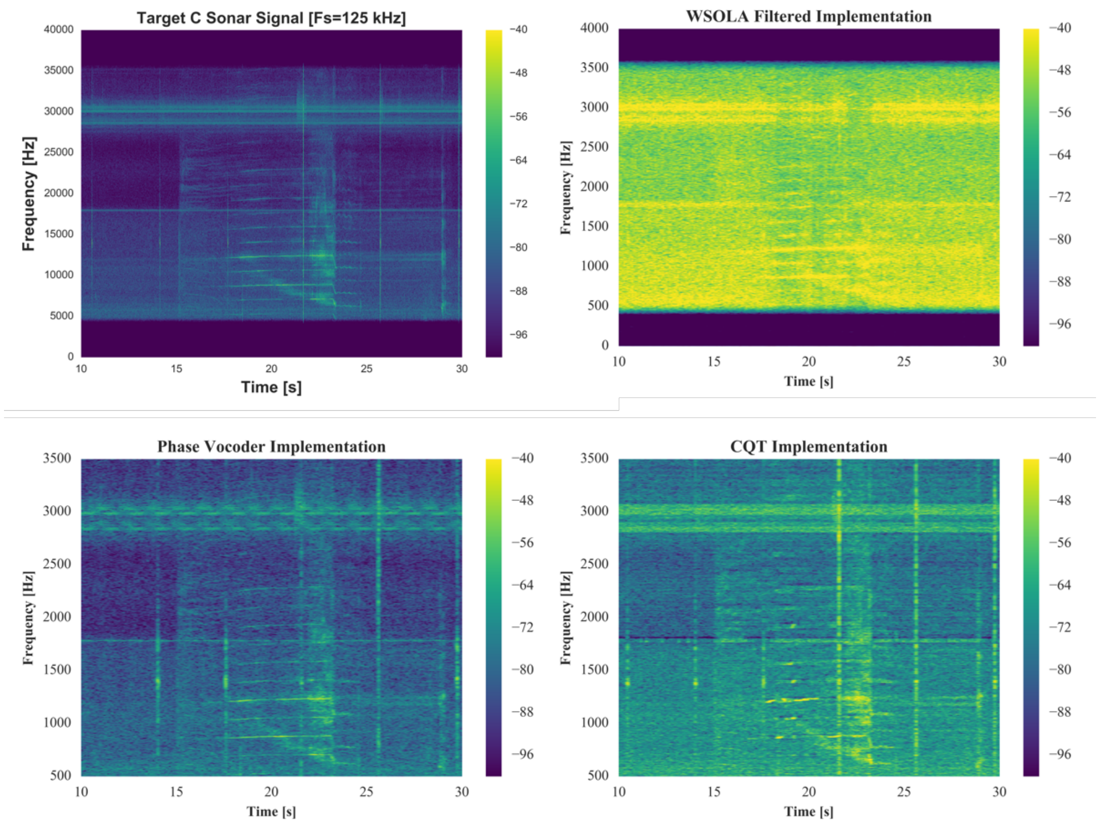

Figure 5.9: Comparison of Passing Jet Ski (Target C).

Comparison of Algorithms

Utilizing the results of the experiment, a measure of computational complexity, and previous discussion on real-time implementation, a comparison of the algorithms is shown below. Metrics include: transient behavior, tonal preservation, noise handling, computational complexity, and real-time implementation. For computational complexity, an estimation is performed by averaging the floating-point operations per second (FLOPS) for the algorithmic implementations used in the experiment: WSOLA (225,000 FLOPS), Phase Vocoder (695,000 FLOPS), and CQT (6 million FLOPS). Algorithms are rated on +/- score based on their results.

	WSOLA	**Phase Vocoder**	**CQT**
Transient Behavior	-	-	+
Tonal Preservation	+	+	-
Noise Handling	-	+	+
Computational Complexity	+	+	-
Real-time Implementation	**Yes (+)**	**Yes (+)**	**No (-)**

Table 1: Comparison of PSM Algorithms.

For a real-time HF passive sonar implementation, the WSOLA algorithm is promising given its computational complexity, however, since the experimental sonar targets were relatively close to the noise floor, WSOLA's requirement for constant periodic analysis lead to overall increased noise that overtakes any semblance of the target signal and results in an undesired audio output. However, given WSOLA has the lowest computational complexity of the three, and has a proven real-time implementation, further investigation into adjusting the tolerance window may improve its signal to noise ratio. WSOLA may also prove valuable in situations where target signals are significantly higher than the noise floor. The phase vocoder algorithm performed well enough to be considered for future use in sonar. While its resulting transient behavior was not desirable, transient suppression maybe helpful when detection of longer time-varying signals is more important. In addition to WSOLA, the phase vocoder has current real-time implementations available for use. Finally, the CQT algorithm consistently outperformed both the WSOLA and phase vocoder for transient detection and noise suppression in the experimental results, however, a real-time implementation does not currently exist. While the computational complexity needs to be reduced, the experimental results give merit to research if a real-time method is feasible. As previously mentioned, methods of time-frame based CQT implementations have recently been presented.

Chapter 6: Conclusion

In this report, an investigation and experiment on pitch shifting HF passive sonar into the auditory range is explored. Compared to previous methods of ultrasonic frequency modulation, pitch shifting preserves the harmonic relationship of the sonar input signal and may prove vital in helping sonar operators classify ultrasonic passive sonar targets in the field. Each of the three algorithms (WSOLA, Phase Vocoder, and CQT) investigated utilize various domains and pitch shifting modification techniques. While the original intent of the algorithms are for speech and music processing, their use in HF passive sonar audio processing may be valuable in future applications.

Future research and experimentation on this subject include further modifications to the algorithm's parameters, utilizing different sonar data, exploring the viability of a real-time CQT implementation, and testing a channel vocoder algorithm. Modifying the TSM parameters, including: the analysis window length, the synthesis overlap size, and the WSOLA tolerance window may result in improved pitch shifting performance. In addition, higher signal-to-noise sonar data may yield different results than presented in this report. Further investigation into a realizable real-time CQT algorithm is also a potential avenue of exploration. Finally, another consideration for future work is to investigate a channel vocoder implementation. While the channel vocoder does not maintain the signal's phase relationship, it may prove to be an effective algorithm in translating HF passive sonar into the auditory range.

References

[1] W.S. Burdic, *Underwater Acoustic System Analysis*. vol. 2, 1991.

[2] Buerau of Navy Personal, *Fleet Type Submarine Sonar Operators Manual*. (1944). NavPers 16167 [Online]. Available: http://maritime.org/doc/fleetsub/sonar/index.htm

[3] Buerau of Navy Personal, *Naval Sonar*. (1953). NavPers 10884 [Online]. Available: http://www.maritime.org/doc/sonar/

[4] R. J. Urick, *Principles of underwater sound*. 1983.

[5] F. B. Jensen, W. A. Kuperman, M. B. Porter, H. Schmidt, and J. F. Bartram, *Computational Ocean Acoustics*, vol. 97, no. 5. 1995.

[6] L. M. Brekhovskikh and Y. P. Lysanov, *Fundamentals of Ocean Acoustics*, vol. 8, no. 6. 1991.

[7] K. V. Mackenzie, “Nine-term equation for sound speed in the oceans,” *J. Acoust. Soc. Am.*, vol. 70, no. 3, p. 807, 1981.

[8] D. Havelock, S. Kuwano, and M Vorlander, *Handbook of Signal Processing in Acoustics*, vol. 2, 2008.

[9] X. Lurton, *An Introduction to Underwater Acoustics: Principles and Applications,* NY, NY: Springer, 2010, ch. 4, pp. 115-117.

[10] G. Allen, *What Is Beamforming?* (2004). [Online]. Available: http://gallen.bitbucket.org/Beamforming/index.html

[11] J. Driedger and M. Müller, “A Review of Time-Scale Modification of Music Signals,” *Appl. Sci.*, vol. 6, no. 2, p. 57, 2016.

[12] D. Barry, “Time and pitch scale modification: A real-time framework and tutorial,” *Conf. Pap.*, pp. 1–8, 2008.

[13] J. Laroche and M. Dolson, “New phase-vocoder techniques for pitch-shifting, harmonizing and\nother exotic effects,” *Proc. 1999 IEEE Work. Appl. Signal Process. to Audio Acoust. WASPAA’99 (Cat. No.99TH8452)*, pp. 91–94, 1999.

[14] J. Laroche and M. Dolson, “Improved phase vocoder time-scale modification of audio,” *IEEE Trans. Speech Audio Process.*, vol. 7, no. 3, pp. 323–332, 1999.

[15] W. Verhelst and M. Roelands, “An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech,” *IEEE Int. Conf. Acoust. Speech, Signal Process.*, vol. 2, no. 1, pp. 2–5, 1993.

[16] W. Verhelst, “Overlap-add methods for time-scaling of speech,” *Speech Commun.*, vol. 30, no. 4, pp. 207–221, 2000.

[17] S. Grofit and Y. Lavner, “Time-scale modification of audio signals using enhanced WSOLA with management of transients,” *IEEE Trans. Audio, Speech Lang. Process.*, vol. 16, no. 1, pp. 106–115, 2008.

[18] J. L. Flanagan and R. M. Golden, “Phase Vocoder,” *Bell Syst. Tech. J.*, vol. 45, no. 9, pp. 1493–1509, 1966.

[19] H. Dudley, “Remaking Speech,” *J. Acoust. Soc. Am.*, vol. 11, no. 2, pp. 169–177, 1939.

[20] M. Portnoff, “Implementation of the Digital Phase Vocoder Using the Fast Fourier Transform,” *Ieee Tassp*, vol. 24, no. 3, pp. 243–248, 1976.

[21] M. Dolson, “The phase vocoder: A tutorial,” *Comput. Music J.*, vol. 10, no. 4, pp. 14–27, 1986.

[22] J. C. Brown, “Calculation of a constant Q spectral transform,” *J. Acoust. Soc. Am.*, vol. 89, no. January 1991, p. 425, 1991.

[23] G. Velasco, N. Holinghaus, M. Dorfler, and T. Grill, “Constructing an invertible constant-Q transform with non-stationary Gabor frames,” *14th Int. Conf. Digit. Audio Eff. (DAFX 2011)*, pp. 93–99, 2011.

[24] C. Schörkhuber, A. Klapuri, N. Holighaus, and D. Monika, “A Matlab Toolbox for Efficient Perfect Reconstruction Time-Frequency Transforms with Log-Frequency Resolution,” *Proc. 53rd AES Conf. Semant. Audio*, pp. 1–8, 2014.

[25] P. Balazs, M. Dörfler, F. Jaillet, N. Holighaus, and G. Velasco, “Theory, implementation and applications of nonstationary Gabor frames,” *J. Comput. Appl. Math.*, vol. 236, no. 6, pp. 1481–1496, 2011.

[26] C. Schörkhuber, A. Klapuri, and A. Sontacchi, “Audio pitch shifting using the constant-Q transform,” *AES J. Audio Eng. Soc.*, vol. 61, no. 7–8, pp. 562–572, 2013.

[27] N. Holighaus, M. Dörfler, G. A. Velasco, and T. Grill, “A framework for invertible, real-time constant-Q transforms,” pp. 1–22, 2012.

[28] A. D’Amico and R. Pittenger, “A brief history of active sonar,” *Aquat. Mamm.*, vol. 35, no. 4, pp. 426–434, 2009.

[29] H. Cox, “Navy Applications of High-Frequency Acoustics,” *AIP Conf. Proc.*, vol. 728, pp. 449–455, 2004.

[30] Y. Pailhas, K. Brown, D. Lane, N. Valeyrie, and C. Capus, “Developing new sensing capabilities for archaeological operations using wideband sonar systems,” *Ocean. 2016 - Shanghai*, 2016.

[31] D. W. Robinson, “Threshold of Hearing and Equal-Loudness Relations for Pure Tones, and the Loudness Function,” *J. Acoust. Soc. Am.*, vol. 29, no. 12, p. 1284, 1957.

[32] "Audacity ®". (2016) *Audacityteam.org*. (Version 2.1.2) [Software]. *Available from http://www.audacityteam.org.*

[33] Parviainen, O. (2015). *Soundtouch Audio Processing Library.* (Version 1.9.2) [Software]. *Available from http://www.surina.net/soundtouch/.*

[34] McFee, B. (2016). *Librosa*. (Version 0.4) [Software]. *Available from http://librosa.github.io/.*

[35] Schörkhuber, C. (2015). *A Matlab Toolbox for Efficient Perfect Reconstruction Time-Frequency Transforms with Log-Frequency Resolution.* [Software]. *Available from http://www.cs.tut.fi/sgn/arg/CQT/.*