Thesis for degree of Master of Science in Music Technology

A Method of Morphing Spectral Envelopes of the Singing Voice for Use with Backing Vocals

Matthew Roddy

August 2013

UNIVERSITY of LIMERICK

OLLSCOIL LUIMNIGH

Department of Computer Science and Information Systems

Digital Media and Arts Research Centre

Primary Supervisor: Dr. Jacqueline Walker (Electronic Engineering Dept.)

Secondary Supervisor: Dr. Nicholas Ward (Computer Science Dept.)

Contents

Abstract

The process proposed in this thesis is based on the empirical observation that it is often desirable for a backing vocalist to match his or her timbre to that of the lead vocalist when the two voices are singing the same phoenetic material concurrently. This is epitomized by the common studio practice of multitracking the lead singer's voice for use as backing vocal tracks. This thesis presents a novel application of voice morphing research for use with a source backing vocal and a target lead vocal in a studio context. Recent studies [CR13] have shown that the interpolation characteristics of line spectral frequencies (LSFs) result in a perceptually smooth transitions between spectral envelopes when LSFs are used as morphing parameters. As such, the potential for the usage of morphing as a subtle effect to attain intermediary timbres between two voices is facilitated. This thesis combines the aforementioned research with a deterministic plus stochastic spectral modeling synthesis (SMS) system to create a version of the proposed process using the simulator Matlab. Several optimizations of morphing techniques are proposed for the unique usage context including an algorithm that performs crossfades of synthesized and unsynthesized audio on the basis of voicedness. The effectiveness of the process is evaluated and design aspects for a real-time version are considered within an adaptive-effect framework.

1. Overview

1.1. Introduction

This work proposes a novel application of voice morphing techniques for use within a studio context. It is noted that often, in popular music recordings, it is desirable for a backing vocalist to blend his or her voice timbre with that of a lead vocalist. In such a situation, the backing vocalist can achieve this by altering the strengths, locations and bandwidths of formants. These parameters are outlined by the curve of the voice's spectral envelope which affects our perception of timbre. If there is a high level of dissimilarity between the spectral envelopes of the two vocalists it can sometimes result in one voice standing out, or harshness. Furthermore, it is observed that often when blending of two vocal parts is desired, pop artists will have the lead vocalist overdub the backing parts to achieve timbral consistancy.

There may be situations in which it is impractical for the lead vocalist to record the backing vocals due to limited studio time or logistical reasons. Or it is possible that the desired voice timbre lies in a perceptual middle ground between the two vocalists. In these situations the studio engineer or producer is limited in the options that are available. The use of equalization to attenuate offending frequencies can help if the backing voice is too harsh, however this approach is inadequate since a certain setting for one phoneme may be inappropriate for the next one.

The voice morphing process proposed in this thesis is built upon the observation that it is common for the backing vocalist to sing the same phoenetic material concurently with the lead vocalist. Therefore the formant structure of the two signals will be similar. If the spectral envelope of the lead vocalist is imposed on the harmonic spectrum of the backing vocalist a certain amount of timbral matching will occur.

1.2. Objectives

The aim of the proposed process is to impose timbral matching on a recording of a backing vocalist using information extracted from the recording of the lead vocalist. The process is designed to function on a frame by frame basis, extracting timbral information from the lead vocal audio through spectral analysis, and performing operations on the backing vocal audio with the extracted information. Since it is not always artistically desirable for the two voices to be fully matched and in some situations a perceptual midpoint may be preferred, the process also aims to achieve a smooth transition between the two different timbres so that a specific level of timbral matching can be achieved. This creation of hybrid timbres that occupy a perceptual midpoint between two voices allows a subtlety of use in a studio situation. Another objective of this work is to investigate the potential for a studio tool that could be used for real-time voice morphing with the intended puropose of transforming the timbre of backing vocals into that of a target lead vocal.

Given the above objectives the purpose of this thesis can be encapsulated in the research question:

> How can current voice morphing techniques be applied within the musical context of a source backing vocal and a target lead vocal framework? How can hybrid vocal timbres be achieved? How feasible is a real-time implementation of such a process and what are the relevant design considerations?

To answer this question a version of the proposed backing vocal morphing process was written in the simulator Matlab. The resulting code is attached in appendix A and can be found in the accompanying data files. Example audio files created using the code are also included and are referred to in chapter 7.

1.3. Problems and Considerations

There are several problems and questions that arise when designing such a process. Firstly, the question of how the voice will be modified must be addressed. There are two broad options, the first of which is the use of time-varying filtering of the original signal. This approach, typefied by linear predictive coding (LPC) and originally proposed in [AH71], has many useful applications in audio analysis and speech coding. However, the weakness inherent in the filter model is that all spectral properties are modelled with the filter when in fact there are multiple sources present within the voice production mechanism. Hence this option of voice modification does not produce satisfactory results for the level of realism required in a muscial context.

The second option is the use of an analysis/re-synthesis technique. The use of a synthesis process introduces a greater level of complexity but also increases the level of control. The analysis/synthesis method adopted in this thesis is spectral modeling synthesis (SMS). It uses two separate elements to model the voice, a harmonic element and a noise residual element. The system has its origins in the work of Xavier Serra [Ser89, SS90] and is discussed further in chapter five.

The specific purpose and context within which this morphing process is proposed presents its own unique problems. Firstly, the details of how the timbre morph is performed within the synthesis framework must be addressed. Also, since a backing vocalist will often sing at a different pitch than the lead vocalist, a requirement is that the fundamental frequency information of the backing vocalist is preserved whilst manipulating the timbre information separately. In addition, there will always be slight timing discrepencies between the onsets of the phonemes of two voices. Therefore a system of controlling the timing of the synthesis is also required. This is addressed using a crossfade algorithm, a process that is central to this application of morphing techniques.

A consideration that is worthy of note is that while the proposed process could be considered a "blending" process, there are other factors which can contribute to listener perception of whether two voices blend. Intonation and temporal allignment are two key elements which can potentially have a large perceptual impact. The process that is outlined in this thesis does not attempt to alter either of these elements and therefore the term blending is avoided in favour of timbral matching.

1.4. Existing Work

This thesis draws from existing work in voice synthesis, voice morphing and audio effect design. Existing work and concepts from these areas are discussed in depth in chapters 2-5 as well as chapter 8. The thesis discusses the most recent research in these fields and their relevance to the proposed backing vocal morphing process. In particular, the work of Marcelo Caetano and Xavier Rodet at IRCAM is of significant importance to the design of this process. Their research into the optimal morphing parameters for a linearity in perceptual feature interpolation [CR13, Cae11, CR09] is the basis upon which this morphing process is designed.

The development of the Matlab code utilized pre-existing code by J. Bonada et. al. published in [BSAL11] (also available online at http://mtg.upf.edu/technologies/sms) as a starting point. Their code implements the sinusoidal plus stochastic synthesis routine used by the proposed process. The analysis and morphing sections of the code are largely the authors own. However some of the resynthesis elements and external functions did not need to be altered. Specifically, the following matlab function files were unaltered: genspecsines.m, genbh92lob.m, peakinterp.m and TWM.m. These are functions that deal with technicalities of the SMS process and therefore did not need to be altered. The morphing element of the code found in the main morphBvLSF.m function is the authors own, as is the crossfade function, bvFade.m which controls the transitions between unaltered audio and synthesized audio.

1.5. Contributions

The contributions of this work stem from the proposed novel application of current morphing techniques. The specific nature of the application means that established morphing techniques can be optimized for the proposed usage context. The proposed application also negates the need for some of the more computationally intense processes associated with voice morphing and voice conversion such as automatic speech recognition (ASR), model training, and dynamic time warping (DTW). While the current process could potentially be expanded by to include some of these elements, they are unnecessary for the functionality of the basic process. One of the main contributions of this thesis is therefore the proposed usage context of morphing techniques and a discussion of relevant optimizations. This is combined with a discussion of the sonic results, and preliminary subjective evaluation of the optimized process in comparison with a baseline morphing system. This discussion is presented in chapter seven.

The primary technical contribution of this thesis is a specially tailored algorithm that controls the transitions between voiced and unvoiced sections of audio. This algorithm uses information from the lead vocal and the backing vocal to decide when and how transitions should occur between synthesized material and the unaltered original audio. Details of this process are discussed in chapter six. Another technical contribution lies in the method used to extract spectral envelope parameters. The method used is based on a state of the art approach proposed by Caetano and Rodet in [CR13]. The method used in this thesis builds on their approach and alters it for use within the proposed context. Certain elements of their method are kept fully intact whilst others are altered or omitted to suit the application. The theoretical aspects behind the envelope parameters are discussed in chapter three. The practical approach to envelope extraction, and the way it contrasts with the approach described in [CR13], is discussed in chapter six.

Another contribution is the discussion and design of an approach to creating a real-time plugin. The proposed process outlined in this thesis fits into the currently vibrant discussion of adaptive audio effects [VA00, VZA06, VWD06]. This potential for a real-time effect and some of the details involved in such an implementation are explored in chapter eight.

1.6. Thesis Outline

This thesis is divided into nine chapters. Chapters two through five represent a review of the relevant concepts and research that inform the work. Chapters six through nine include the novel ideas that are proposed for optimization of the process. The second chapter discusses the physiology of the voice production mechanism, its relation to timbre perception, and methods of estimating the spectral envelope. The third chapter discusses line spectral frequencies which is the envelope representation used in this thesis. The fourth chapter discusses morphing and the most recent research into morphing techniques. The fifth chapter discusses spectral modeling synthesis. The sixth chapter discusses the implementation of the process. It includes a section on the main control element of the process, the crossfade algorithm. It also includes a discussion of the envelope estimation strategy devised for this process and its difference to the one used by the most recent research. The Seventh chapter discusses the sonic results of the process and compares the context-optimized morphing process with a baseline SMS system that morphs using a simple magnitude interpolation. Chapter eight discusses the potential of the process within a real-time context and also within the context of adaptive effects. Finally, in chapter nine the thesis conclusions are presented with reference to the original objectives and the research question.

2. Voice Spectral Envelopes

This chapter outlines the main background concepts that this thesis is based upon. The concepts are used in the implementation and discussion of the morphing process in later chapters. In section 2.1 an introduction to the physical aspects of voice production is presented alongside a definition of formants and their relation to the interconnected tube model of the vocal tract. In section 2.2 timbre perception and a method of measuring salient features is discussed. In section 2.3 the source-filter model of voice production is discusses along with an overview of filter estimation methods.

2.1. Anatomy of the singing voice

A basic understanding of the mechanisms involved in singing voice production is necessary for three reasons. Firstly, the method of synthesis used in this thesis is spectrally related to elements of the voice production process in its decomposition of the signal into a harmonic element and a noise element. Secondly, the physiology of the human voice is inherently coupled with the perception of timbre. Thirdly, the physical representation of the voice production mechanism as found in the acoustic tube model of the human voice has relevance with regard to line spectral frequencies, the parameters that are used to morph the spectral envelope, and which are discussed in chapter three.

The anatomy of the voice can be divided into three main systems: the respiratory system, the larynx and the oropharynx. These are shown in figure 2.1. The respiratory system, which is comprised of the lungs and the diaphragm muscle, supplies the larynx with airflow. The larynx consists of a skeleton of cartilage within which the

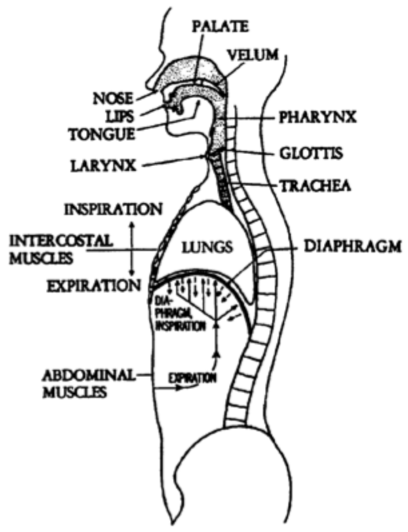

Figure 2.1.: Human vocal tract. From [Kim03].

vocal folds are housed. The vocal folds, or glottis, are made of elastic combination of muscle and ligament, and are the primary source of harmonic sounds. When the folds are apart, or abducted, the air is allowed to flow through unconstricted as is the case during breathing. When the folds are pulled together, or adducted, the airflow is constricted causing vibrations. The oropharynx is a combination of the different resonance cavities such as the oral cavity, the nasal cavity and the pharynx. These different sections of the vocal tract can assume a variety of different shapes, creating the many different resonance characteristics needed to create elements of speech or singing[Kim03].

2.1.1. Formants

These resonance conditions of the human vocal tract can be modeled using an acoustic tube representation. The tube is closed at one end, the vocal folds, and open at the other, the mouth. The wavelengths are proportional to the length of the tube :

$$\lambda_k = \frac{4}{k} L \text{ , where k is odd} \qquad (2.1)$$

where λ is the wavelength, k is the harmonic number and L is the length. Since frequency is simply the inverse of wavelength, the frequencies are also proportional to the length of the tube. The frequencies that are related to the length of the tube will be amplified since they reach a maximum amplitude at the end of the tube. These frequencies are normally called resonances but in the case of the human voice they are called formants. Notably, Perry Cook [Coo91] has proposed to physically model the human vocal tract using interconnected acoustic tubes. While physical modeling is a contrasting approach to the spectral one taken in in this thesis, it is worth mentioning in this case. This relationship between tube length and formants has significance in terms of the timbral characteristics of an individual's voice. Physiological aspects of the structure of an individual's vocal tract impacts our perception of the voice's characteristics due to its resonance behavior.

An important aspect to note about formants and vocal spectral envelopes is that they are independent from the pitch. If we merely shift the partials without changing their amplitudes this results in the sonic equivalent of changing the size of the vocal tube model referred to above. This produces unnatural results. To avoid this the spectral envelope needs to be kept constant while the partials move underneath it.

2.1.2. Voiced and Unvoiced

A key aspect of the morphing process proposed in this thesis revolves around the distinction between two different modes of voice production: voiced and unvoiced. In the case of voiced sounds, the vocal folds are adducted and the flow of air causes rapid vibrations. These rapid vibrations result in a largely harmonic sound source. In the case of unvoiced sounds, the vocal folds remain open, or abducted, and the flow of air flows through the larynx to the mouth where constrictions such as the teeth, tongue or lips cause turbulence creating sound. Sounds can also be a mixed combination of voiced and unvoiced. This is when there is both the vocal folds are vibrating, and there are sounds resulting from air turbulence.

In terms of the english language, all vowel sounds are voiced and some consonant sounds are voiced. Voiced consonants in which no turbulence is required ([l],[m],[n],[r]) are called semivowels. Phonemes which rely on air turbulence in the mouth ([f], [s]) are known as fricatives. Plosives are another class of consonants in which are is suddenly released after being constricted. Plosives can be either voiced ([p], [t], [k]) or unvoiced ([b], [d], [g]).

2.2. Timbre perception

Since the goal of this thesis is to create a method of matching the timbres of two voices, a definition of timbre perception must be identified. The American National Standards Institute (ANSI) has published a definition of timbre that has become widely adopted:

> "Timbre is that attribute of auditory sensation in terms of which a listener can judge that two sounds similarly presented and having the same loudness and pitch are dissimilar." [Ame60]

As such, timbre, pitch and loudness are generally considered to be separate dimensions. Timbre perception is complex and not always completely understood. Key aspects such as attack, spectral shape and harmonic content are some of the dimensions used as perceptual cues. The multidimensional nature of timbre makes it difficult to find ways of classifying timbre. Significant advancements in timbre classification occurred with the application of multidimensional scaling (MDS) techniques to timbre[Gre77]. In this thesis we are only concerned with the spectral aspects of timbre. The temporal aspects of timbre would need modification of attack and decay elements of the backing vocal sound. This is undesirable in the case of the proposed application.

2.2.1. Spectral Features

Finding ways of measuring timbral features from a spectral perspective poses problems in finding quantifiable features that can be ascribed to it. To effectively study morphing processes, ways of measuring timbre must be found. There are many different methods of extracting spectral shape features which have been shown to measure perceptually significant elements of timbre. The morphing research that is discussed in chapter four uses the following four: spectral centroid, spectral spread, spectral skewness and spectral kurtosis. There are other potential spectral feature measures, however these four are the ones that are relevant as they are the ones used in the research discussed in chapter four [CR13] to evaluate perceptual smoothness of morphing parameters.

The four spectral shape features are based on the normalized magnitude spectrum, $p(k)$ which is given by:

$$p(k) = \frac{|X(k)|}{\sum_k |X(k)|} \tag{2.2}$$

where $|X(k)|$ is the magnitude spectrum and k is the frequency index. The shape features are the first four moments of the normalized magnitude spectrum if it is viewed as a probability distribution. They are reprinted from definitions given in [Cae11].

Spectral centroid is commonly considered to be a measure of the "brightness" of a sound. It is obtained by evaluating what could be considered to be the sound's center of gravity. In mathematical terms it is the first moment, or mean of $p(k)$. It is given by the formula:

$$\delta_1 = \mu = \sum_k kp(k) \tag{2.3}$$

The spectral spread is the second moment or variance and is given by:

$$\delta_2 = \sigma^2 = \sum_k (k-\mu)^2 p(k) \tag{2.4}$$

The spectral skewness is the third moment and measures the amount of asymmetry with reference to the spectral centroid:

$$\delta_3 = \frac{\sum_k (k-\mu)^3 p(k)}{\sigma^4} \tag{2.5}$$

The spectral kurtosis is the fourth moment, and is a measure of whether the distribution is tall and skinny or short and squat:

$$\delta_4 = \frac{\sum_k (k-\mu)^4 p(k)}{\sigma^4} \tag{2.6}$$

2.3. Source-Filter Model

A common method of representing audio signals is to separate it into into an excitation source and a time-varying filter. This representation has relevance to most musical sounds but in particular to spoken and sung voice. The relationship between the spectral envelope and the source-filter model is given by:

$$S(\omega) = H(\omega)X(\omega) \tag{2.7}$$

where $H(\omega)$ is the filter transfer function, $X(\omega)$ is the excitation source and $S(\omega)$ is the spectral envelope. Within the context of spectral modeling discussed in chapter five the ideal residual would have a flat spectral envelope giving:

$$X(\omega) = 1 \tag{2.8}$$

In this ideal situation the transfer function of the filter directly defines the spectral envelope:

$$S(\omega) = H(\omega) \tag{2.9}$$

As such, representations of a spectral envelope are interchangeable with its corresponding filter transfer function in the frequency domain.

During the design stages of this process different envelope estimation strategies were experimented with before adopting the LSF representation discussed in chapter three. In this section four different strategies are discussed: linear prediction, cepstrum, discrete cepstrum and true envelope. The first, linear prediction, is a common time domain method used in applications from speech coding to audio compression. The LSFs used in this thesis are derived from linear prediction coefficients. Cepstrum, discrete cepstrum and true envelope, are three methods of envelope estimation that are performed in the frequency domain. They belong to a class of frequency domain filtering operations known as homomorphic filtering. These three methods are discussed briefly for purposes of comparison with the main LSF representation.

2.3.1. Linear Prediction

Traditional linear predictive coding (LPC) is based on the separation of a signal into an excitation source and a synthesis filter. The all-pole filter represents the spectral envelope of the sound. The synthesis filter models resonances and therefore only has poles. This representation is well suited to the human voice since it related to the tube model described in section 2.1.1. The synthesis filter models the human vocal tract and the excitation consists of pulses plus noise.

The basic premise of LPC, as originally defined in [AH71], is that the current sample $x(n)$ can be approximated by a linear combination of past samples of the input signal:

$$\hat{x}(n) = \sum_{k=1}^{p} a_k x(n-k) \tag{2.10}$$

where p is the prediction order, a_k are the linear prediction coefficients and $\hat{x}$ is the current sample. The residual, or prediction error, is calculated by:

$$e(n) = x(n) - \hat{x}(n) \tag{2.11}$$

The z-transform of the prediction filter is :

$$P(z) = \sum_{k=1}^{p} a_k z^{-k} \tag{2.12}$$

The prediction error filter is then given by:

$$A(z) = 1 - P(z) = 1 - \sum_{k=1}^{P} a_k z^{-k} \tag{2.13}$$

Since the ideal source signal is a maximally flat excitation, the spectral envelope is then represented as the transfer function of the all-pole filter:

$$H(z) = \frac{1}{A(z)} = \frac{1}{1 - P(z)} \tag{2.14}$$

Increasing the order of the prediction increases the accuracy of the envelope. However, if the order is too high it will incorporate elements of the troughs between partials which is undesirable.

2.3.1.1. Calculation of LP coefficients

There are several different strategies for calculating the minimum error filter coefficients such as: autocorrelation, covariance method and the Burg algorithm. The autocorrelation method is used in this thesis and can be efficiently implemented using Levinson-Durbin recursion. The details of these calculations are described in [MG74]. The technicalities of this process are omitted from this thesis as it suffices to know that the filter coefficients can be calculated from the normal equations:

$$\sum_{k=1}^{p} a_k r_{xx}(i-k) = r_{xx}(i) \qquad i = 1, \dots p. \tag{2.15}$$

where $r_{xx}(i)$ is the autocorrelation of a sequence defined by:

$$r_{xx}(i) = \sum_{n=i}^{N-1} x(n)x(n-i) \tag{2.16}$$

2.3.2. Cepstrum

There are several different variations of cepstrum methods with applications in fields such as speech processing, seismology and hydroacoustics [CSK77]. The *complex cepstrum* is calculated from the FFT of a signal ($X(k)$) by taking the logarithm of the complex spectrum and performing an inverse-FFT:

$$\hat{x} = \mathcal{F}^{-1} \log(X(k)) \tag{2.17}$$

The *real cepstrum* is calculated using the real part of the FFT spectrum:

$$\hat{c} = \mathcal{F}^{-1} \log(|X(k)|) \tag{2.18}$$

To compute the spectral envelope from the real cepstrum it is multiplied by a lowpass window given by:

$$w = \begin{cases} 1 & n = 0, N_1 \\ 2 & 1 \leq n < N_1 \\ 0 & N_1 < n \leq N - 1 \end{cases} \tag{2.19}$$

where N_1 is the cutoff point between the "highpass" and "lowpass" parts of the cepstrum. The FFT of the windowed real cepstrum gives the spectral envelope:

$$S_w(k) = \mathcal{F}(\hat{c} \cdot w) \tag{2.20}$$

2.3.3. Discrete Cepstrum

The discrete cepstrum as presented in [GR90] is a version of cepstrum that is optimized for harmonic sounds. It is used to solve the problem of estimating a continuous frequency-envelope when the values are specified only at discrete frequencies. This situation is common in analysis/synthesis systems such as the one used in this thesis, where the sound is separated into harmonic and residual elements. It calculates the spectral envelope that links the harmonic peaks, which are usually calculated using a peak picking algorithm. It calculates the spectral envelope by looking for the cepstral coefficients $c(n)$ of a real cepstrum which minimize the error criterion:

$$\epsilon = \sum_{l=1}^{L} || \log a(l) - \log(\bar{F}(f))||^2 \tag{2.21}$$

Where $a(l)$ are the amplitudes of the harmonic partials and $\bar{F}(f)$ is the frequency response of the spectrum given by the cepstral coefficients and calculated by:

$$\bar{F}(f) = c(0) + 2\sum_{i=1}^{N} c(i) \cdot \cos(2\pi f i) \tag{2.22}$$

2.3.4. True Envelope

True envelope is a cepstral based method that was first proposed in [IA79] but has received more attention in recent studies due to a significant reduction in computational cost proposed in[RR05] and [VRR06]. The method compares the K-point FFT $X(k)$ of a signal and the cepstral representation of the spectral envelope $C_i(k)$ at iteration i. The algorithm then updates the input spectrum $A_i(k)$ with the maximum of the original spectrum and the current cepstral iteration:

$$A_i(k) = \max(\log(|X(k)|), C_{i-1}(k)) \tag{2.23}$$

The algorithm is initialized with the setting $A_0(k) = \log(|X(k)|)$ and the algorithm stops when for all k values $A_i(k) < C_i(k) + \theta$ with θ being a user supplied threshold.

3. Line Spectral Frequencies

3.1. Overview

Line spectral frequencies (LSFs) or line spectral pairs (LSPs) are a representation of traditional linear prediction coefficients that are used in speech coding due to their good quantization properties and their relation to the spectrum of the data they represent. LSFs have particular relevence to voice morphing due to their interpolation properties. They are also related to the voice production mechanism since they are theoretically based on the resonance conditions arising from the interconnected tube model of the vocal tract as described in section 2.1. The most recent research has shown that voice morphing using LSPs results in a superior smoothness of perceptual shift between voice charateristics when compared with other envelope representations[CR13].

The LSF representation was originally proposed in [Ita75]. It is based on the concept that linear prediction coefficents can be transformed into a vocal tract resonance function whose boundary conditions are an opening at the lips and a matching termination at the glottis. Itakura proposed that the boundary condition at the glottis could be replaced by a complete opening, or a complete closure. If this is all the poles of the filter move onto the unit circle in the z-plane. This representation is create by two sets of polynomials derived from the linear prediction coefficients with extra terms representing the two different boundary conditions. In the matlab implementation of this process the conversions between linear prediction coefficents to LSFs are performed by the two functions 'poly2lsf' and 'lsf2poly'.

3.2. Conversion from LPC to LSF

LSFs are derived from the coefficients of the all-pole linear prediction filter used to model the resonances in voice signals. The LSF representation works by placing the the complex roots of two polynomials on the unit-circle in the z-plane. The configurate of the LSFs will be such that peaks in the vocal tract filter will correspond to areas where the LSFs are closer together.

For a pth order analysis the linear prediction coefficients are given by:

$$A_p(z) = 1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_p z^{-p} \tag{3.1}$$

Two $(p+1)$th order polynomials are defined, $P(z)$ and $Q(z)$. These are symmetric and antisymmetric polynomials that represent the tube resonance model of the human voice. They represent complete closure at the source end and a complete opening represented by the $(p+1)$th term. The resonance conditions are a combination of the two boundaries which is stated as

$$A_p(z) = \frac{P(z) + Q(z)}{2} \tag{3.2}$$

The two polynomials form an identity with the linear prediction coefficients, in that the exact values of the linear prediction coefficients may be recovered directly from the polynomials. Therefore there is no loss of accuracy in converting back and forth between the two representations.

The two polynomials are the LPC polynomial with an extra feedback term that is positive to model energy reflection at a completely closed glottis, and negative to model energy reflection at a completely closed glottis:

$$P(z) = A_p(z) - z^{-(p+1)} A_p(z^{-1}) \tag{3.3}$$

$$Q(z) = A_p(z) + z^{-(p+1)} A_p(z^{-1}) \tag{3.4}$$

The roots of these two polynomials are the set of LSFs, ω_k given in radians. An example spectrum showing the behaviour of LSFs is shown in figure 3.1.

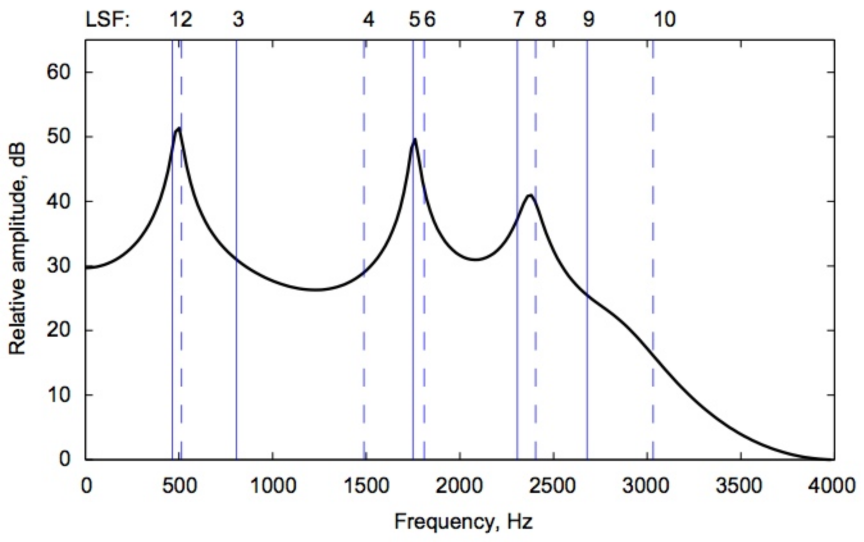

Figure 3.1.: An LPC spectrum with LSFs overlaid. The odd lines are solid and the even lines are dashed. From [McL08].

Converting back to linear prediction coefficients is performed by recreating the polynomials. If M is even:

$$P(z) = (1 + z^{-1}) \prod_{k=2,4,\ldots,M} (1 - 2z^{-1} \cos \omega_k + z^{-2}) \tag{3.5}$$

$$Q(z) = (1 - z^{-1}) \prod_{k=1,3,\ldots,M} (1 - 2z^{-1} \cos \omega_k + z^{-2}) \tag{3.6}$$

When M is odd:

$$P(z) = \prod_{k=2,4,\ldots,M} (1 - 2z^{-1}\cos\omega_k + z^{-2}) \tag{3.7}$$

$$Q(z) = (1 - z^{-2}) \prod_{k=1,3,\ldots,M} (1 - 2z^{-1}\cos\omega_k + z^{-2}) \tag{3.8}$$

3.3. Power Spectral Density Method

The method of LSF calculation used in this thesis computes linear prediction coefficients directly from the power spectrum of the Fourier transform rather than other time domain linear prediction methods such as autocorrelation. The theory behind the implementation is given in [Cae11]. The method treats the signal as a random wide-sense-stationary process in which the first statistical moment and the covariance do not vary with respect to time. The Wiener-Khinchin theorem states that the power spectral density (PSD) of a wide-sense-stationary process, $S_{xx}(\omega)$, is the fourier transform of the autocorrelation of the process. Since the power spectral density is given by the square of the magnitude spectrum we get:

$$S_{xx}(\omega) = |X(\omega)|^2 = \mathcal{F}\{r_{xx}(\tau)\} \tag{3.9}$$

As such, we can calculate the autocorrelation of the function using the real part of the inverse fourier transform of the PSD. To calculate the linear prediction coefficients Levinson-Durbin recursion is used to solve the normal equations given by equation 2.15 as discussed in section 2.3. A practical implementation of the method was written for this thesis and can be viewed in lines 173 to 184 of the matlab code in appendix A.

4. Morphing

4.1. Introduction

Sound morphing is generally considered to be the process of gradually blurring the distinction between a source sound and a target sound. For the human voice, this results in transitional states where perceptual features of the source voice change into those of the target voice. The manner in which this transition occurs is dependent on the paramters used to execute the morph. There are a wide variety of different methods that have been proposed to carry out morphs. The different approaches proposed have varying degrees of success in terms of the accuracy with which they represent the features being interpolated, as well as the smoothness of the perceptual shift from one sound to another. There is no consensus yet on what the definition of the morphing process is for sounds [Cae11]. It is widely accepted however that the resulting transitional sounds from a morph should feature hybridization of the two sounds, creating a perceptual fusion.

4.2. The challenges of morphing

To create a perceptually smooth transition between timbres the manner in which the spectral envelope changes must be relevant to how we perceive sound source. As described in section 2.1.1, vocal timbre is related to the interconnected tube resonance model. The resonances of the tubes model are manifested in formants. Therefore any model that aims to attain perceptually significant morphs must use parameters that address the formant structure of a sound. To alter the sound in ways that do not shift formants will create sounds that do not adhere to a linear shift in the physical attributes of the sound source.

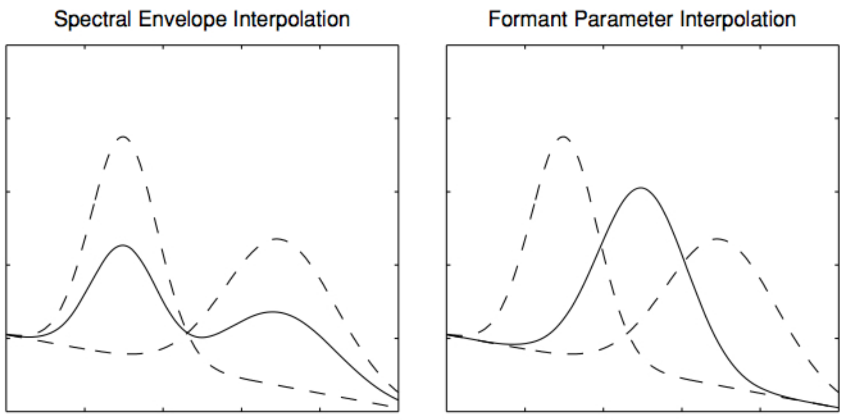

Figure 4.1.: Formant interpolation compared to formant shift. From [RS07].

Figure 4.1 shows two morphs between formants in which the dashed lines represent the two formants being morphed and the solid lines represent the spectral envelope at the midpoint created by the two different morphs. The figure on the left shows the envelope contour that would be created by simple interpolation of the curves. This type of interpolation creates an undesirable smoothed envelope that has two weak formants rather than one strong one. This contour has no perceptual relevance to a shift in a physical representation of the sound source. This type of behaviour is what occurs during bin magnitude interpolation of sound spectrums. The figure on the right shows a morph created by a formant parameterization. In this morph the formant frequency, amplitude and bandwidth occupy a midpoint between the two original formants and therefore the sonic result will occupy a perceptual midpoint.

This type of shift of a single formant can be performed using a process of interpolation of integrals [RS07]. However, in the case of morphing spectrums with more than one formant the morphing problem becomes complicated. The problem now becomes a matter of formant definition. There is also the added problem of how to perform a morph if the number of formants does not match. To parameterize a morph in terms of formants (bandwidths, frequency location and amplitude) they must be clearly defined and there must be an equal number of formants which correspond to one another in terms of indexes. In some morphing situations, such as the karaoke application [CLB+00] described in chapter eight, this type of parameterization is possible due to the fact the formants are predetermined. However, in many applications, including the one described in this thesis, the formants are not clearly defined. Therefore other perceptually relevant parameters must be found which cause shifts in formants.

4.3. Research into optimal morphing parameters

The research presented by Marcelo Caetano and Xavier Rodet [CR13, CR12, CR09, Cae11] investigates the effectiveness of different parameterizations of spectral envelopes in creating a smooth perceptual shift in timbre. Their work involves a study of morphs between a number of different musical instrument tones. In their research they make the distinction between parameters, which are the set of coefficients that are used for controlling the morph, and features, which are the timbral descriptors of the sound. The parameters are morphed using the morphing equation:

$$M(\alpha) = \alpha S_1 + [1 - \alpha] S_2 \tag{4.1}$$

where α is the morphing factor, a number between 0 and 1, and S_1 and S_2 are the spectral envelope parameters.

Their research uses the four spectral shape features described in section 2.2.1 (spectral centroid, spread, skewness and kurtosis) as measures of perceptual distance between timbres. Their work is based on the premise that a linearity in the trajectory that a feature displays when the parameters are interpolated, corresponds to a smoothness in timbral shift. The feature trajectories resulting from the use of different parameters are displayed together in figure 4.2 in a normalized form.

Their work compares a number of different parameters that have been proposed as valid morphing parameters and compares the linearity with which the perceptual features shift. The parameters shown in figure 4.2 are: envelope curve (ENV) [FH96], cepstral coefficients (CC) [SCL96], dynamic frequency warping (DFW) [Pfi04], linear prediction coefficients (LPC) [Moo79], reflection coefficients (RC) [Moo79], and

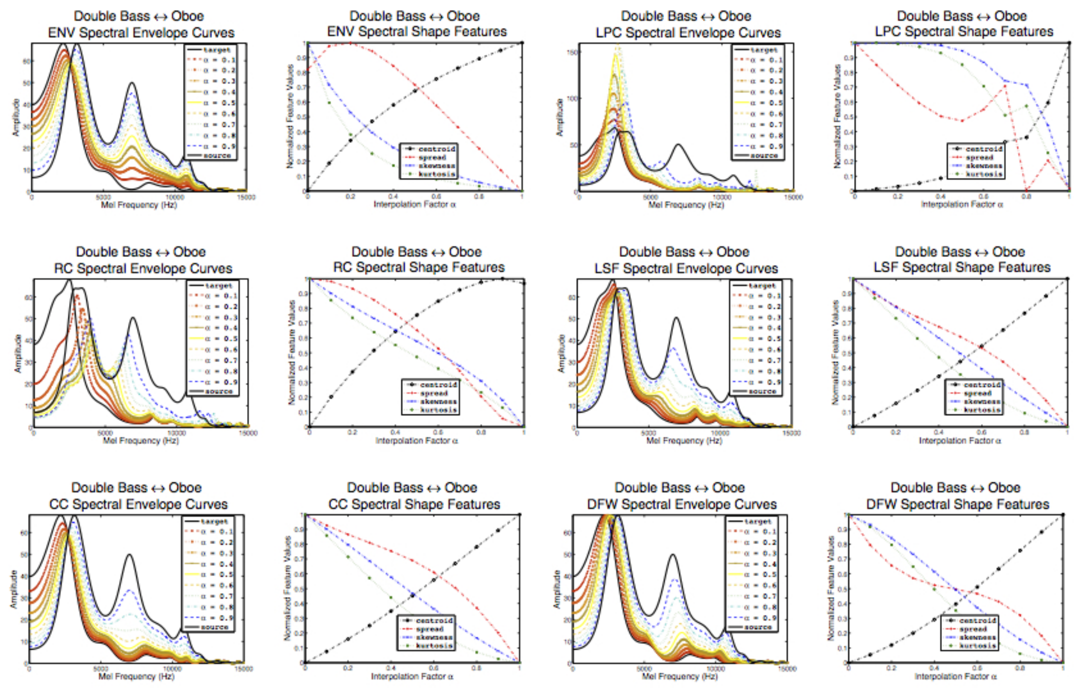

Figure 4.2.: A comparison of morphing parameters using spectral shape features. From [Cae11].

line spectral frequencies (LSF) [Pal95, MC02, McL08, Ita75]. Some of these parameters have been introduced in this thesis already such as cepstral coefficients, linear prediction coefficients and line spectral frequencies. The other five parameters are parameters that have been proposed by researchers using a wide variety of modeling techniques. These are not described in this thesis as they are not directly relevant to the process being proposed in this thesis. What is important is the result of the comparison, which shows that LSFs provide the most linear shift in feature values. It is as a result of this research that LSFs are adopted as the morphing parameters for the proposed morphing process. Reasons why LSFs provide a good representation for morphing is that they are linked to the interconnected tube model of the voice as well as being directly related to the frequency contour of the spectral envelope.

5. Spectral Modeling Synthesis

5.1. Introduction

The approaches to musical voice synthesis, and musical sound synthesis in general, can roughly be divided into two broad methods of parameterization: physical modeling and spectrum modeling. Physical modeling attempts to model the physical attributes of a sound source. Physical modeling benefits from a direct link between control parameters and the method of sound production. However it can suffer from a lack of realism if the model does not take into account all of the relevant physical features of the instrument. To add to this it can also suffer from a high degree of complexity if there is overparameterization. As such in physical modeling there is always a tradeoff between the realism of the model and the ease of user control. Spectral modeling attempts to parameterize the sound as it is perceived by the human ear, in terms of its frequency content. The main advantage of this method is that real sounds may be modified using analysis procedures to extract synthesis parameters.

The approach used in this thesis was pioneered by Xavier Serra[SS90] and is based on modeling sounds as stable sinusoids (partials) plus noise (residual). Sounds are analyzed and new sounds are generated from the generated parameters. This process is often referred to as analysis/synthesis. The analysis portion detects the partials, or harmonic element of the sound. This harmonic element is then subtracted from the original sound and the residual is then represented using filtered noise. It is a combination of additive synthesis for the harmonic element, and subtractive synthesis for the residual. This model is well suited to many musical sounds but is particularly relevant to the human voice.

5.2. Background

The roots of the method of spectral modeling used in this thesis lie in the digital phase vocoder [Moo79] which uses the short term fourier transform to create a frequency domain analysis of a signal. This representation can then be used to perform modifications such as pitch transposition and speed alteration. Since the phase vocoder is modeled as a sum of sine waves it is adept at modeling harmonic sounds. Inharmonic sounds however can prove problematic to analyze and resynthesize. Another limitation of the phase vocoder is that accurate measurement of instantaneous amplitude and frequency are restricted by the number of bins used, which is related to the FFT size.

Improvements to the standard phase vocoder were introduced with the PARSHL program[SS87] developed by Julius Smith in the mid eighties. PARSHL introduced a system whereby the FFT searched for peaks and the largest peaks were tracked from frame to frame. The main advancement this system made was to replace the phase derivative of each FFT bin by interpolated magnitude across FFT bins. This created a better representation of inharmonic sounds. Independently, Thomas Quatieri and Robert McAulay developed a similar system for speech analysis/synthesis which also modeled sounds as a sum of a number of sinusoids[MQ86].

A problem with PARSHL is that it was inefficient in representing noise-like sounds, like the attack of many instruments, since noise is the presence of every frequency within the band limits. This led to a proposed system by Xavier Serra and Julius Smith to represent sinusoids and noise as two separate components [Ser89, SS90].

5.3. Deterministic plus Stochastic Model

The deterministic plus stochastic model proposed by Serra and Smith models the main modes of vibration of a system with sinusoids. The excitation energy, which is energy that is not transformed into stationary vibrations is modeled as a stochastic residual. In the case of the singing voice, the deterministic element corresponds to the resonances created by the vibrations of pitched sounds created by a combination of the vibrations of the glottal folds and the resonance behavior of the vocal tract.

These resonances are modeled by sinusoids that are described by an amplitude and a frequency function. The stochastic element corresponds to aspiration noise and unvoiced consonant sounds which is best described by its power spectral density. As such it is not necessary to preserve the instantaneous phase or magnitude details of FFT frames.

As such the input sound is modeled by:

$$s(t) = \sum_{r=1}^{R} A_r(t)\cos[\theta_r(t)] + e(t)$$

where $A_r(t)$ and $\theta_r(t)$ are the instantaneous amplitude and phase of the r^{th}sinusoid, and $e(t)$ is the noise component.

It is presumed that the sinusoids have slowly changing amplitudes and frequencies. From this, the instantaneous phase is taken to be the integral of instantaneous frequency $\omega_r(t)$, therefore:

$$\theta_r(t) = \int_0^t \omega_r(\tau)d\tau$$

where $\omega_r(\tau)$ is the frequency in radians and r is the sinusoid number. $e(t)$ can be describe as filtered white noise with the equation:

$$e(t) = \int_0^t h(t,\tau)u(\tau)d\tau$$

where $u(t)$ is white noise and $h(t,\tau)$ is the time varying filter to an impulse at time t. A problem with this model is that the distinction between the higher partials of vocal sounds and the noise component is not always clear and the implementation of the process needs to allow user control of parameters that are flexible enough to be modified for different types of signals.

5.4. Analysis/Synthesis Process

There are several potential strategies for implementing the deterministic plus stochastic model. This thesis will only discuss the one used in the accompanying matlab files. Both the analysis and the synthesis process are computed on a frame by frame basis.

5.4.1. Peak Picking

The difference between the phase vocoder and a sinusoidal representation is that the phase vocoder is limited by its use of a fixed number of filters whereas the sinusoidal model can track any number of partials up to the nyquist limit. The detection of these peaks is carried out by a peak-picking algorithm which uses several user-input parameters to select the relevant peaks. The parameters are necessary since merely picking out the largest points in the spectrum can be inaccurate due to the fact that the spectrum is sampled.

The first parameter is a simple peak-height threshold that selects peaks in relation to adjacent frequency bins. This removes the weaker peaks and allows the stronger ones to remain. There are then two different strategies to determining the precise frequency of the peak. The first computes the instantaneous frequency by taking the derivative of the instantaneous phase value and adding it to the bin frequency. These calculated frequencies are the location of each partial which are organized into the frequency tracks. The second method uses a combination of zero-padding and parabolic interpolation to compute the instantaneous frequencies. The zero-padding before the FFT increases the spectral resolution sufficiently to get a reasonable distance between the points required for parabolic interpolation. This type of interpolation requires at least three peaks and has an accuracy of within 0.1%. Since this method provides more robust results it is preferred in this thesis.

5.4.2. Peak Matching

Since the locations and number of peaks change from window to window, a peak continuation algorithm must be used to control the 'birth' and 'death' of sinusoidal components. One method of approaching this is to define sine-wave tracks for 'nearest-neighour' frequencies. The sine wave tracks are allowed to be born at any frame and die at any frame. The peak continuation algorithm works by finding the peak in the next frame that is closest to its current value. If a partial in the current frame is found to be a continuation of a partial in a previous frame, a new track is created. Likewise, if a partial cannot be found to continue a certain track then the track is killed. The peak continuation algorithm can be enhanced by using fundamental frequency as a main control to determine the strength and location of harmonics. The resulting analysis for each frame contains the partial numbers and their corresponding amplitude, frequency, and phase.

5.4.3. Fundamental Frequency Estimation

Estimating the fundamental frequency of a given window is necessary to determine the sound is harmonic and if so what the harmonic structure should be. Usually decisions as to whether the given window is voiced or not are made during this process as well. Many different strategies for estimating the fundamental frequency of a sound exist. Two different methods were informally tested: the Yin algorithm and the two way mismatch algorithm. The Yin algorithm [dCK02] is efficient and works in the time domain using autocorrelation. The two-way mismatch algorithm[Can98] however, works in the frequency domain using the location of the spectral peaks and as such was found to integrate better with the SMS process and produced more consistent results.

5.4.4. Sinusoidal Synthesis

The sinusoidal element can be synthesized in the time domain using a bank of oscillators. However, there is a more efficient way of doing this in the frequency domain based on the inverse-FFT, where the instantaneous phase is not preserved. The method was devised by Rodet and Depalle[RD92] and uses the fact that a stationary sinusoid is a sinc-type function in the frequency domain. A sinusoid can be “painted in” in the frequency domain by creating a complex representation of the sinusoids using the appropriate magnitude, phase and frequency information. This is done using the main lobe of the window transform Using this process as many sinusoids as needed can be added together. More details on this process can be found in [RD92]. For the purposes of this thesis, this part of the synthesis process is performed by functions published in [BSAL11].

6. Discussion of Process Implementation

The following chapter presents the details of how the Matlab implementation of the process works. The elements described in this chapter represent an explanation of how the original technical ideas of this thesis are incorporated into the framework SMS code that this process builds upon. The code for the framework SMS process by J. Bonada et. al. published in [BSAL11] and also available online at http://mtg.upf.edu/technologies/sms is included within the Matlab folder in the accompanying data files for comparative/evaluation purposes. Since the details of how the SMS code works are explained in the source material, only the elements that are relevant to the novel work done in this thesis are discussed in this chapter. The code that is optimized for the vocal morphing process proposed is given in appendix A and is referenced throughout this chapter.

The overall structure of the process can be divided into three parts: analysis, morphing and resynthesis. The following chapter discusses these sections after an explanation of the input/output data elements. Following this there is a discussion of the main controlling element of the optimized process which performs the crossfades between synthesized and unsynthesized sections.

6.1. Input/Output data

The input data used for the main function “morphBvLSF” are:

- *bv*: the backing vocal audio data.

- *lv*: the lead vocal audio data which should be the same length as the backing vocal.
- *fs*: sampling frequency
- *w*: a precalculated analysis window. The size must be odd (e.g a hanning window of size 1025).
- *N*: the FFT size. The example audio files use a length of 2048.
- *t*: the threshold in -dB used by the peak picking algorithm. For the example audio files a threshold of -100 dB was used. If the threshold is set too high, the output will be completely synthesized as the stochastic residual. If it is set too low then the computation can take longer, as more sinusoids will be used. However, it is preferable to set it too low as the residual is also
- *nH*: this is the maximum number of harmonics used for synthesis
- *minf0,maxf0*: these settings are used by the fundamental frequency estimation algorithm. The normal human vocal range is 80 to 1100 Hz however if the voice range is known beforehand the minimum and maximum values can be narrowed. This decreases the processing time required.
- *vuvError*: this is a dimensionless number that represents the error threshold between voiced and unvoiced decision. The threshold uses the uncertainty measure calculated by the two way mismatch algorithm. Higher numbers represent a higher level of uncertainty of the decision. A value of 0.2 was used for the examples.
- *maxhd*: this is the maximum harmonic deviation allowed for a given spectral peak.
- *stocf*: this is the decimation factor used for the residual envelope. A value of 10 was used for the examples.
- *p*: this is the linear prediction order. A value of 60 was used for this in the examples.
- *fadeLen*: this is the length of the crossfade given in number of windows that a full fade will span. It must be a whole number. It was found that values between 1 and 4 produce good results. A value of 3 was used for the examples.

- *expon*: this the exponential factor used for the crossfade. A value of 1 produces a linear crossfade. A value of 2 was found to produce good results.
- *htintp, rintp*: these are morphing factors of the harmonic and residual components respectively. They must be values between 0 and 1.
- *y,yh,ys,yu*: These are the combined, harmonic, stochastic and unvoiced output components.

6.2. Analysis function

The analysis function is used to calculate all the relevant synthesis and morphing parameters that are needed for the deterministic plus stochastic SMS process. One of the central operations that the analysis function performs is to estimate the spectral envelope. A significant difference in the envelope estimation strategies between the method proposed in this thesis and that used by [CR12] is that after the sinusoidal component has been determined using the peak picking algorithm and quadratic interpolation the envelope contour is estimated in a different way. In [CR12], the envelope is is estimated from the peaks using true envelope. Being an iterative version of cepstrum processing, true envelope requires a significant amount of computation. In this thesis the envelope is estimated using simple linear interpolation of the spectral peaks. It was reasoned from the sonic results that using this strategy would not yield significantly different results from one using true envelope and would greatly reduce computation time. Further investigation into this conjecture is required.

In lines 128 and 129 two external functions are called. The first is peakinterp.m is a function which refines the peak values from the peak picking algorithm. The second is the two-way mismatch algorithm that calculates the fundamental frequency and the error percentage which is used to decide whether the the frame is voiced or not. In lines l44 -153 the peaks are refined into harmonics using the harmonic deviation variable, *maxhd*.

Lines 155 - 169 calculate the power spectral density of the derived harmonic spectrum. The strategy adopted to accomplish this was to first convert the harmonic values from decibel to power (line 165) and then use linear interpolation to create the power spectrum. Since the data from the harmonic peaks does not include points at 0 Hz and Fs/2 Hz, these points are added in, using an arbitrarily low value of -150 dB.

The conversion from the power spectrum to linear prediction coefficients takes place in lines 173 to 184. As discussed in chapter three, the autocorrelation of a signal, $r_{xx}(\tau)$, can be given by:

$$\mathcal{F}\{r_{xx}(\tau)\} = |X(\omega)|^2 \tag{6.1}$$

Hence, in line 174, the real part of the inverse fourier transform of the power spectrum is used to calculate the autocorrelation. The LPC coefficients are given by the normal equation:

$$\sum_{k=1}^{p} a_k r_{xx}(i-k) = r_{xx}(i) \quad i = 1, \dots p \tag{6.2}$$

The first p autocorrelation values are used to solve the normal equations using Levinson-Durbin recursion in line 177. To calculate the LSFs from the LPC coefficients the Matlab function “poly2lsf” was used. A separate function could have been included for this process, however this function is included in most standard Matlab distributions since 2006 and it is unlikely that the user is using a release that is older than that.

In lines 188-198 the residual signal is calculated in the frequency domain. The locations of the harmonics and their magnitudes are used to generate a complex spectrum using the “genspecsines” function from [BSAL11]. The harmonic spectrum is then subtracted from the FFT of the input signal to give the residual spectrum.

There are several potential strategies when choosing a representation of the residual. The method chosen here was the same as that used in the original code. It uses a combination of lowpass filtering and resampling, otherwise known as decimation. Decimation creates a smoothed version of the spectrum that can be interpolated.

Since the residual is modeled with noise, interpolating the spectrum with a decimation factor of 10 or less does not create any perceptual problems. To be more precise with morphing of the residual envelopes it would be possible to use a different envelope estimation strategy, such as those mentioned in section 2.3. The same approach as the one used with harmonic component could be used and LSFs could be computed for the residual. However, the perceptual gains would not be significant and the computation time would be significantly increased, as the process of finding polynomial roots in the poly2lsf function requires a computationally significant number of operations.

6.3. Morphing and Synthesis

Once the LSF parameters have been extracted using the analysis function, the parameters are morphed using the morphing formula:

$$M(\alpha) = \alpha S_1 + [1-\alpha] S_2 \tag{6.3}$$

where α is the morphing factor, a number between 0 and 1, and S_1 and S_2 are the spectral envelope parameters, in this case the LSFs, and the residual envelope. The morphing equation is used in lines 76 and 79. Once the LSFs have been morphed (lines 73 -76), they are converted back into LP coefficients using the 'lsf2poly' Matlab function (line 80). From the LP coefficients a magnitude spectrum is calculated which is then interpolated using the harmonic locations. Since the original pitch of the backing vocal needs to be kept, the original harmonic locations of the backing vocalist are used for the interpolation.

The synthesis process (lines 86-105) first uses the 'genspecsines' function to generate the spectral sinusoids of the harmonic signal from the interpolated magnitudes, the harmonic locations of the backing vocal, and the phase values of the backing vocal. The phase values from the backing vocals can remain unaltered because the original harmonic locations remain the same, only the magnitudes are changed. In line 88 the residual envelope is interpolated to its original size. The noise element is then generated by using the matlab 'rand' function to create random phase values. The stochastic element is generated in the frequency domain using the random phase values and the interpolated residual envelope (line 95). In lines 96-97 the harmonic and stochastic frequency domain signals are then converting into the time domain using inverse FFTs.

6.4. Crossfade Algorithm

The sinusoidal plus stochastic synthesis model adopted in this thesis provides a powerful framework for the manipulation of harmonic sounds. However, one of its deficiencies lies in the lack of accuracy with which it represents inharmonic sounds. In the original SMS morphing code as given in [BSAL11], unvoiced consonants are represented by filtered noise. This method of synthesizing consonant sounds creates an unrealistically smooth temporal envelope, since the realism of many consonant sounds is dependent on high-frequency details that are lost when using normal window sizes. The approach adopted in this thesis to make up for the lack of realism of synthesized unvoiced consonants is to reinsert the original consonants of the backing vocalist. The crossfade algorithm controls when transitions between the original audio and the synthesized audio occur, as well as the speed of the transition, and the exponential curve used to create the crossfade. The algorithm was created in response to the observation that fades which spanned only the length of the normal hop size (256) were too short and caused unnatural sounding transitions from one voice to another.

The algorithm also addresses one of the main considerations of this process which is that morphing should only occur when both voices are singing voiced material. Since the intention of this process is to work on a frame by frame basis, if the content of the lead vocal frame is unvoiced while the content of the backing vocal frame is voiced, the resulting effect would be to morph the spectral envelope of consonant with a sung vowel, creating noise. In the inverse situation, when the lead vocal is voiced and the backing vocal is unvoiced this would also create an undesirable effect if the two sounds were morphed. Therefore, in this process, rules are set up within the matlab code so that the only time the synthesized audio is used in full is when both voices are singing pitched, voiced material. There are therefore two purposes for the algorithm: to add realism to the consonants by reintroducing original, unsynthesized material, and secondly, to control when the morphing process is performed.

Creating a fade between voiced and unvoiced sections of audio is complicated by inaccuracies in the voiced/unvoiced detection and the hop size. The two way mismatch voiced/unvoiced detection strategy is not always accurate in that short voiced segments of audio may be incorrectly detected as unvoiced due to a shift of pitch or change in timbre. If a short segment of audio of one or two windows is judged to be unvoiced when the surrounding audio is voiced, this can lead to clicky unpleasant shifts between the two waveforms. Increasing the error threshold of the VUV detection algorithm is often ineffective in these cases due to the nature of the waveform.

To counteract this behavior, a transition period between voiced and unvoiced audio segments is used where an exponential crossfade minimizes the impact that erroneously classified sections have on the final combined waveform. For example, given a window size of 1024, a hop size of 256 and that the initial state of the crossfade is voiced (only synthesized material, none of original signal), then a full transition between voiced to unvoiced will require four consecutive unvoiced classifications in either the backing vocal or the lead vocal windows. If we have a linear crossfade and instead have two consecutive unvoiced classifications then the gain of the gain of the original signal and the synthesized signal will increase and decrease to 0.5 respectively over the course of 512 samples. If these two unvoiced windows are followed by two voiced classifications in both the backing vocal and the lead vocal windows, then the gain will return to its original state over the course of another 512 samples.

Based on subjective observation, this algorithm greatly diminished the perceptual impact that erroneously classified windows have on the overall process. To add to this, increasing the transition length can also have a positive effect in that misclassifications of one or two windows will have less of a perceptual impact. However, if the length of transition is too long then there is the risk that unvoiced consonant sounds are synthesized rather than left to the original waveform. In the matlab code the length of the transition is user specified.

The use of an exponential as shown in figure 6.1, rather than a linear crossfade can

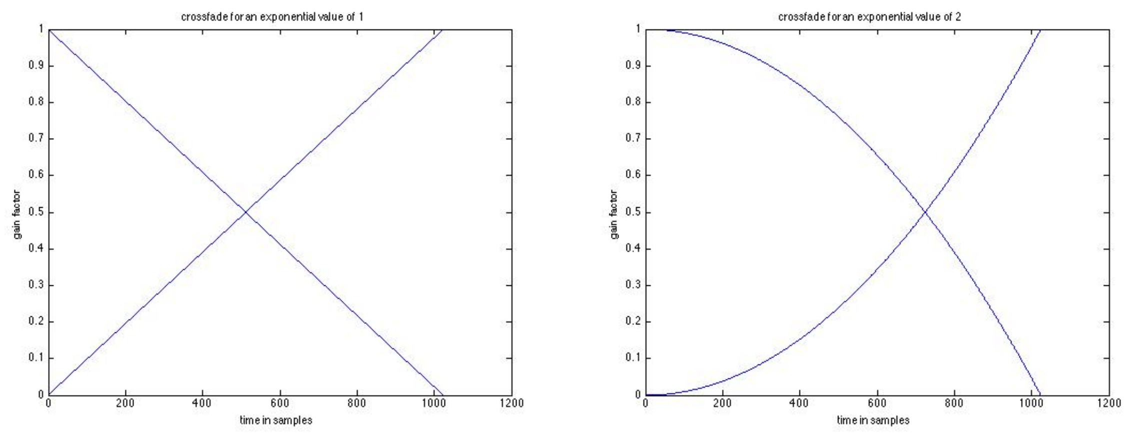

Figure 6.1.: Crossfade gain factors using different exponential values.

also increase the effectiveness of the algorithm. Given a transition over the course of four windows, if an exponential curve with an exponential factor of two is used, the first two windows will have a slower rate of transition than the next two. Using an exponential curve is also more in keeping with the non-linear nature of human hearing. It was considered to use a fade system in decibels, however this can be simulated using the exponential and using a decibel scale results in problems to do with the final gain amount never reaching zero. The matlab file uses two wavetables that are calculated in lines 18-20 of bvFade.m using the parameters supplied by the user. The decreasing fade is inverted to create the increasing fade. This results in a crossfade that has balanced gain.

7. Discussion of Sonic Results

While no formal evaluation of the sonic results was performed the following section will present a number of subjective observations of the results. The accompanying audio files are referred to as examples. The audio files consist of recordings of two singers from a rock and roll band singing a passage from an original song. The two original source recordings are given in tracks 01 and 02. The singer with the lower pitch material, singer A, is taken to be the lead vocalist and the higher singer, singer B, the backing vocalist. The morphing algorithm is applied to the backing vocalist in different configurations and with different α morphing factors. For comparative purposes, audio files of the baseline harmonic plus stochastic synthesis algorithm (without the use of LSFs and the crossfade algorithm) are also supplied. The code used for these files is written by Bonada/Serra and is published in [BSAL11]. All of the analysis/synthesis settings, such as window sizes, hop sizes, FFT sizes, etc., are kept the same for both the original Bonada/Serra code and the code optimized for this process which is given in appendix A.

7.1. Comparison of unmorphed synthesis methods

Tracks 03 and 04 present the two different synthesis methods performed on the backing vocalist, and using an α value of zero. Track 03, uses the optimized code and 04 uses the original code. It is observed that the version using the original code, track 04, synthesizes the unvoiced consonants at the beginning of each diphone. The fact that it is synthesized is apparent in the audio due to the “phasey” quality of the consonants. This is caused by the presence of the sinusoidal element of the synthesis. In the version synthesized using the optimized code, this phasey quality is gone due to the re-introduction of the unsynthesized original consonant using the crossfade algorithm. When track 03 is compared to the original source recording (track 01), it is surmised that it would be difficult for most listeners to tell that it was synthesized.

7.2. Comparison of morph factors of 0.5

Tracks 05 and 06 present $\alpha = 0.5$ using the optimized code and the original code, respectively. Comparing the two segments, the benefits of using LSFs in the optimized version over a simple crossfade of envelopes in the original code can be heard. The original code creates a morph in which the vowels loose perceptual definition due to the smoothing of the spectral envelope caused by the linear crossfade. Although the vowels are still intelligible, there is a lack of realism. This lack of realism makes it difficult to judge whether the segment belongs to a perceptual middle ground between the two singers.

In comparison, the version created using LSFs exhibits vowel definition and realism. There is no apparent smoothing of the spectral envelope and the perceptual space that it occupies could be considered to belong to a middle ground between the two singers. Although the vocal timbre created by the spectral envelope morph does not belong to a real person, it could be mistaken as belonging to one. As expected, these segments audibly support the research discussed in chapter four and subjectively confirms that LSFs serve as good morphing parameters when a hybrid perceptual intermediary between two sounds is desired. It is also noteworthy that, as in the case of the unmorphed segments in section 7.1, the use of real consonants is a significant contributing factor to the sense of realism.

7.3. Comparison of morph factors of 1.0

Tracks 07 and 08 present $\alpha = 1.0$ using the the optimized code and the original code. The segment created with the original code (track 08) no longer exhibits the envelope smoothing that is heard in the $\alpha = 0.5$ version as the envelope has been fully morphed. As such the peaks and troughs outlining the formats belonging to the lead vocal are used with the harmonics of the backing vocal. When track 08 is compared with the original unaltered sound file of the lead vocal (track 02) it could be considered that both sounds were sung by the same person. However, despite the significant realism of the voiced vowel sections, this method still suffers from a lack of realism in the unvoiced/consonant sections as was discussed in the unmorphed synthesis section.

The version using the optimized code (track 07) exhibits all the positive attributes of the original code in that there is a realism of the vowels and the perceptual identity of the backing vocalist has been altered to assume that of the lead vocalist. There is also the added enhancement that the introduction of the real consonants produces a higher level of realism.

Despite the significant improvements that the optimized code offers, there is a perceivable negative side during the transitions between voiced and unvoiced segments. In track 07 slight “glottal” artifacts can be heard during these transitions due to the rapid shift in the spectral envelope contour of the singer during the transition. These perceptual impact of these sounds can be diminished by increasing the length of the crossfade from three to six, as can be heard on track 09. The reduction of these transitional artifacts by increasing the length of the crossfade comes at the expense of the definition of the consonants, as discussed in chapter six. From these observations it is suggested that an increase in the morphing factor α should correspond to an increase in the length of the crossfade. However, the amount that the length is increased by should depend on the level of difference between the two envelopes.

7.4. Discussion of process within a two voice context

Tracks 11 through 15 combine the original lead vocal recording with processed versions of the backing vocals with different α values at 0, 0.25,0.5, 0.75 and 1. By playing through the tracks in sequence we can hear how the backing vocal track perceptually “sits in” more with the lead vocalist as the morph factor is increased.

7.5. Limitations

On track 17 the main limitation of using morphing within the context proposed in this thesis is heard. This track swaps the two original voice files and uses the voice with the lower fundamental frequency contour as the source and the voice with the higher fundamental contour as the target. With a morph factor of 1.0, the result is an undesirable effect in which the the partial magnitudes are incorrect for the given vowel. This phenomenon is best explained with reference to the figure 7.1. The figure shows the spectral envelopes of the first three formants of a vowel and the magnitudes of harmonics given at two different pitches.

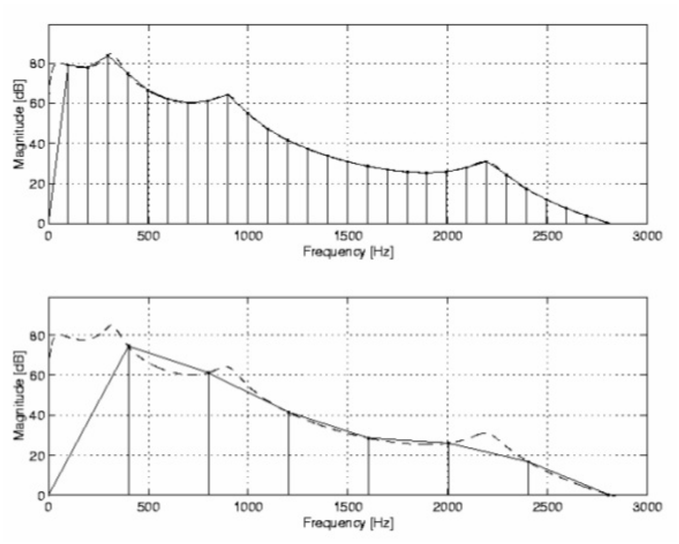

Figure 7.1.: Spectrum of the phoneme [u] with $f_0 = 100$ Hz and $f_0 = 400$ Hz. From [Los07].

As can be seen from the lower image, the spectral envelope, as given by the higher pitched vowel (delineated by the solid line), does not supply enough information to correctly synthesize a lower pitch. If we were to take the spectral envelope from the magnitudes acquired from the higher pitch, and use the harmonics from the harmonic structure with a lower fundamental, the formant locations would not be represented properly. To add to this, if there is a formant that is below the fundamental frequency of the higher pitched voice, there is no way of determining the location of this formant directly from the vowel.

This formant behavior places a significant limitation on the utility of the proposed timbre matching process. It means that the proposed process, in its current form, can only be used with backing vocals that are higher in pitch than the lead vocal. There are several potential solutions to this problem. One would be to artificially add in the formants based on statistical models such as hidden markov models and/or codebook style referencing. A solution could also potentially incorporate information recorded from previous frames to determine the formant behavior. However a proper investigation into these potential additions are beyond the scope of this thesis.

8. Real-time implementation

8.1. Overview

The potential for the morphing process proposed in this thesis to be implemented in a studio situation as a digital audio workstation (DAW) plugin is discussed in the following chapter. Using a side-chain input, the signal from the lead vocalist could be used to control the effect on the backing vocal. This control structure of the process fits into the adaptive effect framework discussed by Verfaille[VA00]. Adaptive effects are briefly discussed, an example of a real-time SMS voice morphing system is discussed, followed by a discussion of potential implementation details.

8.2. Adaptive Effects

Adaptive effects are audio effects that incorporate an adaptive control stage that is controlled by features extracted from a sound [VA00]. This control structure is depicted in figure 8.1.

Two common examples of adaptive effects are the side-chain compressor and auto-tune. There are four different types of adaptive effects that are identified by Verfaille:

- Auto-adaptive: where effects use features that are extracted from the input signal. E.g: compressors, auto-tuners.
- External adaptive: where effects use features that are extracted from at least one other input signal. E.g: side-chain compression.
- Feedback adaptive: where effects use features that are extracted from the output signal.

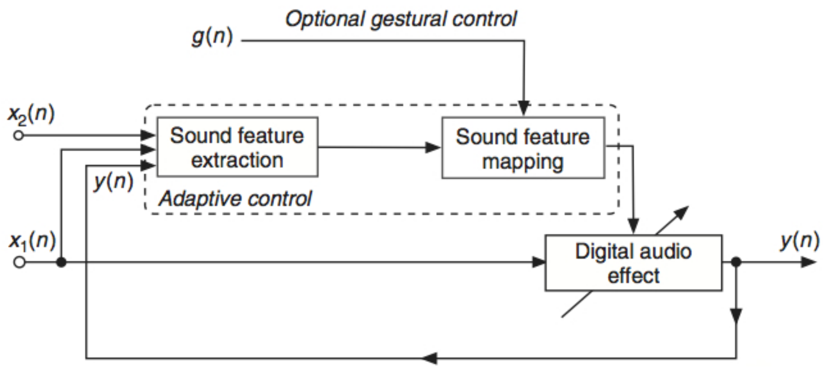

Figure 8.1.: Adaptive effect control. Sound features are extracted from X_1 or X_2, or the output signal Y. From [VZA06].

- Cross-adaptive: where effects use features that are extracted from more than one external source. E.g: Automatic mixing systems.

The process outlined in this thesis falls into the category of external adaptive since the features are extracted from a single external signal. In this case the features being extracted are the voicedness of the signal and the LSFs of the external signal. These are used as control features for the resynthesis of the source signal.

Within the context of the research by Verfaille, the evaluation of an adaptive effect relies on the accuracy of the feature computation, and the mapping to the control. Within this context it can be said that the accuracy of the voicedness extraction is not accurate enough to be mapped directly to a single-window crossfade. However this is accounted for in the mapping process by using a fade over several windows. The accuracy of the extraction of LSFs could be said to be accurate, and since the LSFs are mapped directly onto the LSFs of the backing vocals, the mapping can be said to provide good control.

8.3. Examples of Real-Time SMS systems

To assess the feasibility for the process proposed in this thesis to be used in real-time examples of real-time SMS-systems must be examined. Several real-time SMS implementations have been designed for voice manipulation/synthesis purposes. Most of these systems have been built at the Music Technology Group (MTG) in conjunction with Yamaha. The most well documented real-time SMS system is discussed in [CLB$^+$00, DBBC$^+$00, BLCS01]. This voice morphing karaoke system was designed to let users sing like their favorite singer.

In the application, the user sings along to the song and is given the option of setting parameters that control pitch, timbre vibrato and articulations. These features of the user are morphed with those of the target singer. In order to do this the complete song is analyzed beforehand. The system first segments what the user is singing on the basis of phonemes and notes. It then looks for the same sounds in the target performance, morphs the features, and synthesizes the otuput voice. The system uses a similar base-line synthesis system as the one used in this thesis. However it also uses a Hidden Markov Model-based speech recognizer which is responsible for matching the singing voice of the user to that of the target. As in the process proposed in this thesis, only voiced phonemes are morphed so that the unvoiced consonants are always in the output.

The system requires a phonetic transcription of the lyrics, the melody as midi data and the original target voice recording. A non-real-time system is used to analyze the original recording and perform the segmentation. During the process each analysis frame is matched with a target frame and the parameters are morphed. The pre-analysis of the audio allows the undesirable behaviour involving lower backing vocal fundamentals exhibited by this process (see sections 7.5 and 4.2) to be avoided since formant areas are well defined. This pre-definition of formant areas also addresses the problem of spectral envelope smoothing that occurs during simple interpolation of envelopes.

8.4. Audio Plugin considerations

Since it has been established that the use of SMS in conjunction with real-time applications is possible, a few considerations for how such a plugin might function are discussed. Firstly, the basic implementation should involve a dual input structure, with a main input for the backing vocal and a side chain input for the lead vocal signal. Side chain inputs are supported by all the main plugin platforms such as

Steinberg's Virtual Studio Technology (VST), Apple's Audio Units (AU), and Avid's Real Time AudioSuite (RTAS).

Since the matlab code is designed to be flexible for the purposes of experimentation, there are a large number of input parameters that can be discarded when transfering the process to realtime. A proposed list of input parameters could be:

- morph factor
- cross fade length
- cross fade exponential
- harmonic threshold
- voiced/unvoiced error threshold

It is suggested that optimal settings for the other parameters can be either predetermined or evaluated during the analysis stage. For example, the proper prediction order can be estimated by using the number of relevant formant peaks, as discussed in [Vas08]. Another optimization that could benefit a real-time process is the use of multithreading in a similar fashion to that used in [CLB+00]. By using multithreading the two analysis stages and the synthesis stage could be processed independently. A final proposed addition that would improve the real-time functionality is cross-checking feature that assesses the probability that the backing vocalist is singing the same phoenetic material as the lead vocalist. If the probability is above a certain amount then the morphed audio is engaged. If not, the backing vocal is left unmodified.

9. Conclusion

To evaluate the success of the research in this thesis the research question posed in the introduction and reprinted below is discussed:

> How can current voice morphing techniques be applied within the musical context of a source backing vocal and a target lead vocal framework? How can hybrid vocal timbres be achieved? How feasible is a real-time implementation of such a process and what are the relevant design considerations?

This question is divided up into its three different parts and each one addressed separately.

9.1. Application of morphing techniques

This thesis addresses the question of how current voice morphing techniques can be applied to backing vocals by proposing a frame by frame morphing process in which the spectral envelope of the lead vocalist is used to alter the spectral envelope of the backing vocalist on a one-to-one basis. The thesis identifies the state of the art in morphing practices as given in [CR13] and incorporates them into the musical context. The application of the morphing process is then optimized by an original controlling process which functions on the basis of voicedness information from the two frames. The algorithm has two purposes. Firstly, it is used to solve the problem of the lack of realism of unvoiced consonant sounds by crossfading the synthesized segments with the original unsynthesized audio. Secondly the algorithm addresses the problem of timing discrepancies between the two voices by favoring unsynthesized audio when either of the voices is unvoiced.

Another optimization that is suggested is an alternative method of envelope estimation to the one described in the state of the art literature. The method suggested in [CR12] utilizes the true envelope method, while in this thesis a simple linear interpolation of the harmonic element's spectrum is used. While it is not questioned that the accuracy of the true envelope method is greater, it is questioned whether the substantial increase computation required warrants its use in this situation. A definitive answer to this question is not given in this thesis, the question is merely posed as a result of a subjective assessment of the quality of the audio obtained with the interpolation method.

The thesis also presents a discussion of the sonic results of the proposed application of morphing techniques in chapter seven. In this chapter the primary limitation of the application is identified: that using the process to morph backing vocals with a fundamental frequency that is lower than the lead vocalist will result in unpleasant timbres due to a loss of detail in the formant structure of the spectral envelope.

9.2. Hybrid vocal timbres

The thesis addresses the second part of the research question regarding hybrid timbres by the adoption of of LSFs as morphing parameters. The research presented in [CR13] and discussed in chapter four, shows that these parameters provided the best linearity of feature interpolation behavior when used in the context of musical instrument morphing. This thesis applies this research to the human voice to create hybrid voices that occupy a perceptual midpoint between the two voices. This aspect of the process is a major contribution of the work as it allows the proposed process to be used as a subtle artistic tool. The subjective conclusions drawn from the audio results discussed in chapter seven point towards this aspect of the process potentially being useful in a studio context.

9.3. Feasibility of real-time implementation

The feasibility of implementing this process as a plugin is high as was discussed in chapter eight. The plugin architecture described should support the process in the form of an adaptive effect. The question that then stems from the third part of the research question is whether such an implementation is worth the time needed to write such a plugin? To answer this question it is worth taking into account the behavior it produces with backing vocals that have a lower fundamental than the lead vocalist. If a method of minimizing the effect of this behavior was found the attractiveness of creating such a plugin would increase.

Acknowledgments

I would like to thank Dr. Jacqueline Walker for very kindly agreeing to help me with this thesis. Her advice and guidance was greatly appreciated. I would like to thank Jürgen Simpson for his memorable lectures and his help during the early stages of this thesis. I would like to thank my parents for their support in everything I do. Most of all, I would like to thank Nick Ward for his help with this thesis, and his all-round advice on matters big and small.

A. Appendix A: Matlab Code

A.1. Notes on Using the Matlab Code

The following matlab code is included with the necessary external functions (peak-interp.m, genbh92lobe.m, peakinterp.m and TWM.m from [BSAL11]) in the accompanying data files. An example Matlab workspace is provided with the necessary data to test the code. The audio data in the matrices *bv* and *lv* are the author's own recordings of a backing vocalist and a lead vocalist. The original function upon which the author's morphBvLSF.m function is based is supplied in the "original code" folder for comparative purposes.

A.2. Code

A.2.1. morphBvLSF.m

```matlab
function [y,yh,ys,yu] = morphBvLSF(bv,lv,fs,w,N,t,nH,minf0,maxf0,vuvError,...
    maxhd,stocf,p,fadeLen,expon,htintp,rintp)

% Author: Matthew Roddy, August 2013
% Builds upon code published in DAFX: Digital Audio Effects, 2nd ed.
% by J. Bonada, X. Serra, X. Amatriain, A. Loscos
%
% morph between two sounds using LSFs and the harmonic plus stochastic model
%
% bv: backing vocal, lv: lead vocal
% fs: sampling rate, w: analysis window (odd size),
% N: FFT size (minimum 512), t: threshold in negative dB,
% nH: maximum number of harmonics, minf0: minimum f0 frequency in Hz,
% maxf0: maximim f0 frequency in Hz,
% vuvError: error threshold in the f0 detection (ex: 0.2),
% maxhd: max. relative deviation in harmonic detection (ex: 5)
% stocf: decimation factor of mag spectrum for stochastic analysis
% p: linear prediction order (ex: 60)
% fadeLen: length of crossfade, in windows. Must be a whole number (ex: 3)
% expon: exponential factor for crossfade (ex: 2)
% htintp: harmonic timbre interpolation factor,
% rintp: residual interpolation factor,f0et
% yh: harmonic component, ys: stochastic component, yu: unvoiced component
% y: output sound

if length(htintp)==1
    htintp =[ 0       length(bv)/fs;     % input time
        htintp  htintp        ];  % control value
end
if length(rintp)==1
    rintp  =[ 0      length(bv)/fs;       % input time
        rintp  rintp           ];    % control value
end
M = length(w);                          % analysis window size
Ns = 1024;                              % FFT size for synthesis
H = 256;                                % hop size for analysis and synthesis
N2 = N/2+1;                             % half-size of spectrum
soundlength = length(bv);                % length of input sound array
hNs = Ns/2;                             % half synthesis window size
hM = (M-1)/2;                           % half analysis window size
pin = max(H+1,1+hM);                    % initialize sound pointer to middle of analysis window
pend = soundlength-max(hM,H);           % last sample to start a frame
yh = zeros(soundlength+Ns/2,1);         % output sine component
ys = zeros(soundlength+Ns/2,1);         % output residual component
yu = zeros(soundlength+Ns/2,1);         % output residual component
w = w/sum(w);                           % normalize analysis window
sw = zeros(Ns,1);
ow = triang(2*H-1);                     % overlapping window
ovidx = Ns/2+1-H+1:Ns/2+H;              % overlap indexes
sw(ovidx) = ow(1:2*H-1);
bh = blackmanharris(Ns);                % synthesis window
bh = bh ./ sum(bh);                     % normalize synthesis window
wr = bh;                                % window for residual
sw(ovidx) = sw(ovidx) ./ bh(ovidx);
sws = H*hanning(Ns)/2;                  % synthesis window for stochastic
minpin2 = max(H+1,1+hM);                % minimum sample value for lv
maxpin2 = min(length(lv)-hM-1);         % maximum sample value for lv
inFadePos = Ns;


while pin<pend

    %%%% First sound analysis %%%%
    [f0Bv,hlocBv,mXsenvBv,lsfBv,gainBv,hiBv,hphaseBv,bvVuV] = analysis(bv,fs,w,wr,pin,M,hM,N,
    N2,Ns,hNs,nH,t,vuvError,minf0,maxf0,maxhd,stocf,p);
    %%%% Second sound analysis %%%%
    pin2 = round(pin/length(bv)*length(lv)); % linear time mapping between inputs
    pin2 = min(maxpin2,max(minpin2,pin2));
    [f0Lv,hlocLv,mXsenvLv,lsfLv,gainLv,hiLv,hphaseLv,lvVuV] = analysis(lv,fs,w,wr,pin2,M,hM,N,
    N2,Ns,hNs,nH,t,vuvError,minf0,maxf0,maxhd,stocf,p);


    %%%% Morph %%%%
    chtintp = interp1(htintp(1,:),htintp(2,:),pin/fs);    % get control value
    crintp = interp1(rintp(1,:),rintp(2,:),pin/fs);        % get control value
    mYsenv = mXsenvBv*(1-crintp) + mXsenvLv*crintp; % morph residual envelope

    %%%% LSFs %%%%
    ylsf = lsfBv*(1-chtintp) + lsfLv*chtintp; % interpolate the lsfs
    ya = lsf2poly(ylsf);  %Linear prediction coefficents
    A=-20*log10(abs(fft(ya,N2)));%  Magnitude spectrum from LP
    envelope=A+gainBv; % spectral envelope
    bins = 1:N2;
    hMag = interp1(bins,envelope,hlocBv(1:hiBv-1)); % magnitudes of harmonics

    %%%%% Re-synthesis %%%%
    Yh = genspecsines(hlocBv(1:hiBv-1),hMag,hphaseBv,Ns);                 % generate sines
    mYs = interp(mYsenv,stocf);                      % interpolate to original size
    roffset = ceil(stocf/2)-1;                    % interpolated array offset
    mYs = [ mYs(1)*ones(roffset,1); mYs(1:Ns/2+1-roffset) ];
    mYs = 10.^(mYs/20);                              % dB to linear magnitude
    pYs = 2*pi*rand(Ns/2+1,1);                    % generate phase random values
    mYs1 = [mYs(1:Ns/2+1); mYs(Ns/2:-1:2)];       % create magnitude spectrum
    pYs1 = [pYs(1:Ns/2+1); -1*pYs(Ns/2:-1:2)]; % create phase spectrum
    Ys = mYs1.*cos(pYs1)+1i*mYs1.*sin(pYs1);    % compute complex spectrum
    yhw = fftshift(real(ifft(Yh)));               % sines in time domain using IFFT
    ysw = fftshift(real(ifft(Ys)));               % stoc. in time domain using IFFT
    ro = pin-hNs;                         % output sound pointer for overlap
    yh(ro:ro+Ns-1) = yh(ro:ro+Ns-1)+yhw(1:Ns).*sw;  % overlap-add for sines
    ys(ro:ro+Ns-1) = ys(ro:ro+Ns-1)+ysw(1:Ns).*sws; % overlap-add for stochastic
    % crossfade algorithm
    [synthFade,uvFade,outFadePos] = bvFade( inFadePos,Ns,H,bvVuV,lvVuV,fadeLen,expon );
    inFadePos = outFadePos;

    %apply fade to hop size amount of samples
    yh(pin-hM:pin-hM+H-1) = yh(pin-hM:pin-hM+H-1).*synthFade(:);
    ys(pin-hM:pin-hM+H-1) = ys(pin-hM:pin-hM+H-1).*synthFade(:);
    yu(pin-hM:pin-hM+H-1) = bv(pin-hM:pin-hM+H-1).*uvFade(:);
    pin = pin+H;                                    % advance the sound pointer

end
y = yh+ys+yu;                                              % sum sines and stochastic

end

function [f0,hloc,mXsenv,lsf,gain,hi,hphase,yVuV] = analysis(x,fs,w,wr,pin,M,hM,
N,N2,Ns,hNs,nH,t,vuvError,minf0,maxf0,maxhd,stocf,p)

xw = x(pin-hM:pin+hM).*w(1:M);                   % window the input sound
fftbuffer = zeros(N,1);                          % initialize buffer for FFT
fftbuffer(1:(M+1)/2) = xw((M+1)/2:M);            % zero-phase window in fftbuffer
fftbuffer(N-(M-1)/2+1:N) = xw(1:(M-1)/2);
X = fft(fftbuffer);                          % compute the FFT
mX = 20*log10(abs(X(1:N2)));                 % magnitude spectrum
pX = unwrap(angle(X(1:N/2+1)));              % unwrapped phase spectrum
ploc = 1 + find((mX(2:N2-1)>t) .* (mX(2:N2-1)>mX(3:N2)) ...
    .* (mX(2:N2-1)>mX(1:N2-2)));                 % find peaks
[ploc,pmag,pphase] = peakinterp(mX,pX,ploc);                 % refine peak values
[f0, error] = TWM(ploc,pmag,N,fs,minf0,maxf0);      % find f0

if(error>vuvError)
    yVuV= 0;
else
    yVuV = 1;
end

hloc = zeros(nH,1);                          % initialize harmonic locations
hmag = zeros(nH,1)-100;                      % initialize harmonic magnitudes
hphase = zeros(nH,1);                        % initialize harmonic phases
hf = (f0>0).*(f0.*(1:nH));                   % initialize harmonic frequencies
hi = 1;                                      % initialize harmonic index
npeaks = length(ploc);                       % number of peaks found

while (f0>0 && hi<=nH && hf(hi)<fs/2)  % find harmonic peaks
    [dev,pei] = min(abs((ploc(1:npeaks)-1)/N*fs-hf(hi)));     % closest peak
    if ((hi==1 || ~any(hloc(1:hi-1)==ploc(pei))) && dev<maxhd*hf(hi))
        hloc(hi) = ploc(pei);                  % harmonic locations
        hmag(hi) = pmag(pei);                  % harmonic magnitudes
        hphase(hi) = pphase(pei);              % harmonic phases
    end
   hi = hi+1;                                % increase harmonic index
end
hloc(1:hi-1) = (hloc(1:hi-1)~=0).*((hloc(1:hi-1)-1)*Ns/N);  % synth. locs

%%%%%%%%%%
```

```
% Calculate Power Spectral Density from harmonic Peaks %%%%%%%%%
% First get rid of zeros in peak data and change harmonic index (hi) value
% Then add endpoints at zero and Fs/2, followed by interpolation.
I = find(hloc); %Get indices of nonzero elements
hi2 = length(I)+2;
hloc2 = zeros(hi2,1);hloc2(1)=0;hloc2(2:hi2-1)=hloc(I);
hloc2(hi2) = N2;
hmag2 = zeros(hi2,1); hmag2(2:hi2-1) = hmag(I);
hmag2(1) = -150; hmag2(hi2) = -150; % Arbitrarily low magnitude for 0 Hz and fs/2 Hz
hPSD = power(10,hmag2(1:hi2).*((1/20)*2)); % First half of power spectrum
hFreq = hloc2(1:hi2).*(fs/N2);
xi = 1:(0.5*fs)/N2:0.5*fs;
yi = interp1(hFreq,hPSD,xi, 'linear'); % yi are squared absolute value
yPSD = [yi(1:N2-1) fliplr(yi(1:N2-1))]; % power spectral density


%%%%%%%%%
% Linear Predicion Coefficients from Power Spectral Density
rxx = real(ifft(yPSD)); % Autocorrelation is IFFT of PSD
R = rxx(1:p);
if norm(R)~=0
    a=levinson(R,p);     % Levinson-Durbin recursion
else
    a=[1, zeros(1,p)];
end
lsf = poly2lsf(a);  % Calculate LSFs from LPC coefficients
R=R(:)'; a=a(:)';   % row vectors
g=sqrt(sum(a.*R));  % gain factor
gain=20*log10(g);   % Linear prediction gain in dB


%%%%%%%%%
% Calculate Residual envelope by subtracting complex harmonic spectrum
% from original signal.

ri= pin-hNs;                          % input sound pointer for residual analysis
xr = x(ri:ri+Ns-1).*wr(1:Ns);         % window the input sound
Xr = fft(fftshift(xr));               % compute FFT for residual analysis
Xh = genspecsines(hloc,hmag,hphase,Ns);            % generate sine
Xr = Xr-Xh;                           % get the residual complex spectrum
mXr = 20*log10(abs(Xr(1:Ns/2+1)));    % magnitude spectrum of residual
mXsenv = decimate(max(-200,mXr),stocf); % decimate the magnitude spectrum
end
```

A.2.2. bvFade.m

```
function [ synthFade,uvFade,outFadePos ] = bvFade( inFadePos, wLen,hopSize,
bvVUVbool,lvVUVbool,fadeLength,expon )
% bvFADE  Calculates crossfade arrays for transitions between voiced and
%unvoiced sections.
% inFadePos: number between 0 and 1 specifying fade position. 1 is all
% unvoiced unsynthesized waveform. 0 is all synthesized.
% wLen: length of synthesis window
% hopSize: the hopsize specifies how long the output vectors will be
% bvVUVbool/lvVUVbool: booleans specifying whether current BV or LV windows
% are voiced.
% fadelength: the length over which the crossfade occurs. Specified in
% amount of windows (ie 1-3). Must be a whole number
% expon: exponent controling the curve of the crossfade. 1 is a linear
% crossfade. 2 produces good results.

synthFade = zeros(hopSize,1);
uvFade = zeros(hopSize,1);
```

```
fadeOutTable = (1/(fadeLength*wLen)):(1/(fadeLength*wLen)):1;
fadeOutTable = 1-power(fadeOutTable,expon);
fadeInTable = (1-fadeOutTable);
if (bvVUVbool==0 || lvVUVbool==0)% if either unvoiced
            endPos = inFadePos-1+hopSize;
    if(endPos < wLen*fadeLength-1)
       uvFade(:) = fadeInTable(inFadePos:endPos);
       synthFade(:) = fadeOutTable(inFadePos:endPos);
       outFadePos = endPos;
    else
        synthFade(:) = 0;
        uvFade(:) = 1;
        outFadePos = inFadePos;
    end
end
if (bvVUVbool && lvVUVbool)% if both BV and LV are voiced
            endPos = inFadePos-hopSize;

    if(endPos > 1)
       uvFade(:) = fadeInTable(inFadePos-1:-1:endPos);
       synthFade(:) = fadeOutTable(inFadePos-1:-1:endPos);
       outFadePos = endPos;
    else
        synthFade(:) = 1;
        uvFade(:) = 0;
        outFadePos = inFadePos;
    end
end
end
```

A.3. Audio track listings

The audio files included with the accompanying data are files that were synthesized with the morphBvLSF.m function written for this thesis. These are presented alongside audio synthesized with the unoptimized morphing code from [BSAL11] using the hpsmodelmorph.m function so that the audible optimizations for the proposed context can be heard. The settings that the two morphing functions have in common (FFT size, window type, hop sizes, etc.) were kept the same. The effect of changing the crossfade length in the optimized code is demonstrated (track 9) as well as the effect of having a backing vocalist with a lower fundamental than the lead vocalist (track 17).

1. Original unsynthesized backing vocal audio
2. Original unsynthesized lead vocal audio
3. morphBvLSF: No morphing.
4. hpsmodelmorph: No morphing.
5. morphBvLSF: morph factor of 0.5.
6. hpsmodelmorph: morph factor of 0.5.
7. morphBvLSF: morph factor of 1.
8. hpsmodelmorph: morph factor of 1.
9. morphBvLSF: number of windows for crossfade changed from 3 to 6.
10. morphBvLSF: Demonstration of a gradual morph from 0 to 1.
11. morphBvLSF: Backing and lead vocals mixed together. No morph.
12. morphBvLSF: Backing and lead vocals mixed together. Morph factor of 0.25.
13. morphBvLSF: Backing and lead vocals mixed together. Morph factor of 0.5.
14. morphBvLSF: Backing and lead vocals mixed together. Morph factor of 0.75.
15. morphBvLSF: Backing and lead vocals mixed together. Morph factor of 1.
16. hpsmodelmorph: Backing and lead vocals mixed together. Morph factor of 1.
17. morphBvLSF: Effect of morphing a lower f_0 source voice with a higher f_0 target voice.

Bibliography

[Abe96] M. Abe, “Speech morphing by gradually changing spectrum parameter and fundamental frequency,” in *ICSLP Proceedings of the Fourth International Conference on Spoken Language*, vol. 4, 1996, pp. 2235–2238.

[ABLS01] X. Amatriain, J. Bonada, A. Loscos, and X. Serra, “Spectral modeling for higher-level sound transformations,” in *Proceedings of MOSART Workshop on Current Research Directions in Computer Music*, 2001.

[AH71] B. S. Atal and S. L. Hanauer, “Speech analysis and synthesis by linear prediction of the speech wave,” *The Journal of the Acoustical Society of America*, vol. 50, p. 637, 1971.

[Ame60] *ANSI(1960)*, American Standard Acoustical Terminology Std., 1960.

[BLCS01] J. Bonada, A. Loscos, P. Cano, and X. Serra, “Spectral approach to the modeling of the singing voice,” in *111th AES Convention*, New York, 30/11/2001 2001.

[BM06] T. Backstrom and C. Magi, “Properties of line spectrum pair polynomials: A review,” *Signal Processing*, vol. 86, no. 11, pp. 3286–3298, Nov. 2006.

[BSAL11] J. Bonada, X. Serra, X. Amatriain, and A. Loscos, *Spectral Processing.* John Wiley & Sons, Ltd, 2011, pp. 393–445. [Online]. Available: http://dx.doi.org/10.1002/9781119991298.ch10

[Cae11] M. Caetano, “Morphing isolated quasi-harmonic acoustic musical instrument sounds guided by perceptually motivated features,” Ph.D. dissertation, PhD thesis, UPMC-Ircam, 2011.

[Can98] P. Cano, “Fundamental frequency estimation in sms analysis,” in *Proceedings of COST G6 Conference on Digital Audio Effects*, 1998.

[CLB$^+$00] P. Cano, A. Loscos, J. Bonada, M. De Boer, and X. Serra, “Voice morphing system for impersonating in karaoke applications,” in *In Proceedings of the 2000 International Computer Music Conference (ICMC ’00)*, 2000, p. 109Ð112.

[CMSW05] A. Caclin, S. McAdams, B. K. Smith, and S. Winsberg, “Acoustic correlates of timbre space dimensions: A confirmatory study using synthetic tones,” *The Journal of the Acoustical Society of America*, vol. 118, no. 1, p. 471, 2005.

[Coo91] P. R. Cook, “Identification of control parameters in an articulatory vocal tract model, with applications to the synthesis of singing,” Ph.D. dissertation, Stanford, CA, USA, 1991.

[CR09] M. Caetano and X. Rodet, *Evolutionary spectral envelope morphing by spectral shape descriptors.* University of Michigan Library, 2009.

[CR12] ——, “A source-filter model for musical instrument sound transformation,” in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, 2012, p. 137Ð140.

[CR13] ——, “Musical instrument sound morphing guided by perceptually motivated features,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 8, pp. 1666–1675, Aug. 2013.

[CSK77] D. G. Childers, D. P. Skinner, and R. C. Kemerait, “The cepstrum: A guide to processing,” *Proceedings of the IEEE*, vol. 65, no. 10, p. 1428–1443, 1977.

[DBBC$^+$00] M. De Boer, J. Bonada, P. Cano, A. Loscos, and X. Serra, “Singing voice impersonator application for PC,” in *Proceedings of the International Computer Music Conference*, 2000.

[dCK02] A. de Cheveigne and H. Kawahara, “Yin, a fundamental frequency estimator for speech and music,” *The Journal of the Acoustical Society of America*, vol. 111, no. 4, p. 1917, 2002.

[EMGP05] T. Ezzat, E. Meyers, J. Glass, and T. Poggio, “Morphing spectral envelopes using audio flow,” in *Proc. ICASSP*, 2005, pp. 1029–1032.

[ES09] R. T. Eakin and X. Serra, "Smspd, libsms and a real-time sms instrument," in *Proceedings of the 12th International Conference on Digital Audio Effects (DAFx09), Como, Italy, September 1*, vol. 4, 2009.

[FCR10] M. Freitas Caetano and X. Rodet, "Independent manipulation of high-level spectral envelope shape features for sound morphing by means of evolutionary computation," in *Proc. of the 13th Int. Conference on Digital Audio Effects (DAFx-10)*, 2010, pp. 11–21.

[FCR11] ——, "Sound morphing by feature interpolation," in *Proceedings-ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing*, 2011, pp. 11–23.

[FH96] K. Fitz and L. Haken, "Sinusoidal modeling and manipulation using lemur," *Computer Music Journal*, vol. 20, no. 4, pp. 44–59, 1996.

[FMO07] K. Furuya, T. Moriyama, and S. Ozawa, "Generation of speaker mixture voice using spectrum morphing," in *Multimedia and Expo, 2007 IEEE International Conference on*, 2007, pp. 344–347.

[GR90] T. Galas and X. Rodet, "An improved cepstral method for deconvolution of source–filter systems with discrete spectra: Application to musical sound signals," in *Proceedings of the International Computer Music Conference (ICMC)*, Glasgow, September 1990.

[Gre77] J. M. Grey, "Multidimensional perceptual scaling of musical timbres," *The Journal of the Acoustical Society of America*, vol. 61, p. 1270, 1977.

[GS97] E. B. George and M. J. Smith, "Speech analysis/synthesis and modification using an analysis-by-synthesis/overlap-add sinusoidal model," *Speech and Audio Processing, IEEE Transactions on*, vol. 5, no. 5, pp. 389–406, 1997.

[IA79] S. Imain and Y. Abe, "Spectral envelope extraction by improved cepstral method," *Electron. and Commun. in Japan*, vol. 62-A(4), pp. 10–17 (In Japanese), 1979.

[Ita75] F. Itakura, "Line spectrum representation of linear predictor coefficients of speech signals," *The Journal of the Acoustical Society of America*, vol. 57, p. 35, 1975.

[KBA03] W. Kleijn, T. Backstrom, and P. Alku, “On line spectral frequencies,” *IEEE Signal Processing Letters*, vol. 10, no. 3, pp. 75–77, Mar. 2003.

[KC08] H. K. Kim and S. H. Choi, “Cepstral domain interpretations of line spectral frequencies,” *Signal Processing*, vol. 88, no. 3, pp. 756–760, Mar. 2008.

[Kim03] Y. E. Kim, “Singing voice Analysis/Synthesis,” Ph.D. dissertation, Massachusetts Institute of Technology, 2003.

[Los07] A. Loscos, “Spectral processing of the singing voice,” Ph.D. dissertation, Citeseer, 2007.

[MC02] R. W. Morris and M. A. Clements, “Modification of formants in the line spectrum domain,” *Signal Processing Letters, IEEE*, vol. 9, no. 1, pp. 19–21, 2002.

[McL08] I. V. McLoughlin, “Line spectral pairs,” *Signal Processing*, vol. 88, no. 3, pp. 448–467, Mar. 2008.

[MG74] J. Markel and J. Gray, A., “A linear prediction vocoder simulation based upon the autocorrelation method,” *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 22, no. 2, pp. 124–134, 1974.

[Moo79] J. A. Moorer, “The use of linear prediction of speech in computer music applications,” *Journal of the Audio Engineering Society*, vol. 27, no. 3, pp. 134–140, 1979.

[MQ86] R. McAulay and T. Quatieri, “Speech analysis/synthesis based on a sinusoidal representation,” *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 34, no. 4, p. 744–754, 1986.

[MWD$^+$95] S. McAdams, S. Winsberg, S. Donnadieu, G. De Soete, and J. Krimphoff, “Perceptual scaling of synthesized musical timbres: Common dimensions, specificities, and latent subject classes,” *Psychological research*, vol. 58, p. 177–192, 1995.

[OG03] S. OÕLeary and N. J. J. Griffith, “A hybrid approach to timbral consistency in a virtual instrument,” in *Proceedings of 6th Conference on Digital Audio Effects*, 2003.

[Pal95] K. K. Paliwal, “Interpolation properties of linear prediction parametric representations,” in *Fourth European Conference on Speech Communication and Technology*, 1995.

[Pfi04] H. R. Pfitzinger, “Dfw-based spectral smoothing for concatenative speech synthesis.” in *INTERSPEECH*, 2004.

[RD92] X. Rodet and P. Depalle, “Spectral envelopes and inverse FFT synthesis,” in *Audio Engineering Society Convention 93*, 1992.

[RR05] A. Robel and X. Rodet, “Efficient spectral envelope estimation and its application to pitch shifting and envelope preservation,” in *Proc. DAFx*, 2005.

[RS07] X. Rodet and D. Schwarz, “Spectral envelopes and additive+ residual analysis/synthesis,” in *Analysis, synthesis, and perception of musical sounds.* Springer, 2007, pp. 175–227.

[RVR07] A. Robel, F. Villavicencio, and X. Rodet, “On cepstral and all-pole based spectral envelope modeling with unknown model order,” *Pattern Recognition Letters*, vol. 28, no. 11, pp. 1343–1350, Aug. 2007.

[SB98] X. Serra and J. Bonada, “Sound transformations based on the sms high level attributes,” in *Proceedings of the Digital Audio Effects Workshop*, 1998.

[SCL96] M. Slaney, M. Covell, and B. Lassiter, “Automatic audio morphing,” in *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, vol. 2, 1996, pp. 1001–1004.

[Ser89] X. Serra, “A system for sound analysis/transformation/synthesis based on a deterministic plus stochastic decomposition,” Ph.D. dissertation, Stanford University, 1989.

[Ser93] ——, “Spectral modeling synthesis: Past and present,” in *Proceedings of DAFX London*, 1993.

[Ser97] ——, “Musical sound modeling with sinusoids plus noise,” *Musical signal processing*, vol. 1997, pp. 91–122, 1997.

[SR99] D. Schwarz and X. Rodet, "Spectral envelope estimation and representation for sound analysis-synthesis," in *Proceedings of the International Computer Music Conference, Beijing, China*, 1999.

[SS87] J. Smith and X. Serra, "Parshl an analysis/synthesis program for non-harmonic sounds based on a sinusoidal representation," in *International Computer Music Conference*, Urbana, Illinois, USA, 23/08/1987 1987, pp. 290–297.

[SS90] X. Serra and J. Smith, "Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition," *Computer Music Journal*, vol. 14, no. 4, pp. 12–24, 1990.

[VA00] V. Verfaille and D. Arfib, "A-DAFx: adaptive digital audio effects," *energy*, vol. 4000, p. 6000, 2000.

[Vas08] S. Vaseghi, *Advanced Digital Signal Processing and Noise Reduction.* Wiley, 2008.

[VD04] V. Verfaille and P. Depalle, "Adaptive effects based on stft, using a source-filter model," in *Proc. of the 7th International Conference on Digital Audio Effects (DAFx)*, 2004.

[VM00] T. S. Verma and T. H. Meng, "Extending spectral modeling synthesis with transient modeling synthesis," *Computer Music Journal*, vol. 24, no. 2, p. 47Ð59, 2000.

[VRR06] F. Villavicencio, A. Robel, and X. Rodet, "Improving lpc spectral envelope extraction of voiced speech by true-envelope estimation," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, vol. 1, 2006.

[VRR09] ——, "Applying improved spectral modeling for high quality voice conversion," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, 2009, pp. 4285–4288.

[VWD06] V. Verfaille, M. M. Wanderley, and P. Depalle, "Mapping strategies for gestural and adaptive control of digital audio effects," *Journal of New Music Research*, vol. 35, no. 1, pp. 71–93, Mar. 2006.

[VZA06] V. Verfaille, U. Zolzer, and D. Arfib, “Adaptive digital audio effects (a-DAFx): a new class of sound transformations,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 5, pp. 1817–1831, Sep. 2006.

[YY04] H. Ye and S. Young, “High quality voice morphing,” in *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP’04). IEEE International Conference on*, vol. 1, 2004, pp. 1–9.