Efficient Approaches for Voice Change and Voice Conversion Systems

By

Yuhang Ye

Thesis presented for the Degree of

Master of Engineering Science

to

Electronic Engineering Department

Maynooth University

Supervisor:

Dr Bob Lawlor

Submitted to Electronic Engineering Department, Maynooth University

October 2018

DECLARATION

I hereby declare that this thesis is my own work and has not been submitted in any form for another award at any other university or institute of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is given.

Signed: Yuhang Ye

Yuhang Ye

Date: 28/01/2019

ABSTRACT

In this thesis, the study and design of Voice Change and Voice Conversion systems are presented. Particularly, a voice change system manipulates a speaker's voice to be perceived as it is not spoken by this speaker; and voice conversion system modifies a speaker's voice, such that it is perceived as being spoken by a target speaker.

This thesis mainly includes two sub-parts. The first part is to develop a low latency and low complexity voice change system (i.e. includes frequency/pitch scale modification and formant scale modification algorithms), which can be executed on the smartphones in 2012 with very limited computational capability. Although some low-complexity voice change algorithms have been proposed and studied, the real-time implementations are very rare. According to the experimental results, the proposed voice change system achieves the same quality as the baseline approach but requires much less computational complexity and satisfies the requirement of real-time. Moreover, the proposed system has been implemented in C language and was released as a commercial software application. The second part of this thesis is to investigate a novel low-complexity voice conversion system (i.e. from a source speaker A to a target speaker B) that improves the perceptual quality and identity without introducing large processing latencies. The proposed scheme directly manipulates the spectrum using an effective and physically motivated method – Continuous Frequency Warping and Magnitude Scaling (CFWMS) to guarantee high perceptual naturalness and quality. In addition, a trajectory limitation strategy is proposed to prevent the frame-by-frame discontinuity to further enhance the speech quality. The experimental results show that the proposed method outperforms the conventional baseline solutions in terms of either objective tests or subjective tests.

Table of Content

List of Figures

List of Tables

Acronyms

ADMM	Alternating Direction Method of Multipliers
AM	Amplitude Modulation
AMDF	Average Magnitude Difference Function
ANN	Artificial Neural Networks
AR	Autoregressive model
ASR	Automatic Speech Recognition
BFS	Breadth-First Searching
BPTT	BackPropagation Through Time
CFWMS	Continuous Frequency Warping and Magnitude Scaling
CNN	Convolutional Neural Network
CTL	Classification Trajectory Limitation
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DKPLS	Dynamic Kernel Partial Least Square
DNN	Deep Neural Network
DTW	Dynamic Time Warping
EM	Expectation Maximization
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
GMM	Gaussian Mixture Model
GRU	Gate Recurrent Unit
HMM	Hidden Markov Model
HNM	Harmonic + Noise Model
HSM	Harmonic/Stochastic Model
IFFT	Inverse Fast Fourier transform
IIR	Infinite Impulse Response
LPC	Linear Predictive Coding
LSF	Line Spectral Frequency
LSP	Line Spectral Pair

LSTM	Long Short-Term Memory Network
MCC	Mel Cepstral Coefficients
MCD	Mel-Cepstral Distortion
MFCC	Mel-Frequency Cepstral Coefficients
MLP	Multi-Layer Perceptron
NMD	Non-negative Matrix Deconvolution
NMF	Non-negative Matrix Factorization
OLA	OverLap and Add
PAOLA	Pitch Alignment OLA
PCA	Principal Component Analysis
PSOLA	Pitch Synchronous Overlap-and-Add
RBF	Radial Basis Function
RBM	Restricted Boltzmann Machine
RNN	Recurrent Neural Network
RTRBM	Recurrent Temporal Restricted Boltzmann Machine
STASC	Speaker Transformation Algorithm using Segmental Codebooks
VLTN	Vocal Tract Length Normalization
VOIP	Voice over IP
VQ	Vector Quantization
WSOLA	Waveform Similarity and Overlap Add

Chapter 1

Introduction

Speech is an indispensable communication tool between individuals. To this end, it is interesting, valuable and essential to developing an automatic speech processing system for real-life applications. A specific instance of these systems is the voice transformation system that concerns modifying the perceptual identity of speech signals. There are two common types of voice transformation system that will be studied in this thesis, named voice change and voice conversion.

# Voice Change

Voice change is defined as the process that morphs the utterance of a speaker such that the processed speech is not perceived as having been spoken by the original speaker. There are various applications that fall into this category, such as voice gender conversion [1], voice masking [2] and voice imitation (e.g. robotic, Christmas Santa and Talking Tom® [3]). For voice change systems, there is NO need for converting the speech of a source speaker to be perceived as being spoken by another target speaker. Instead, the real-world application usually provides users with a friendly interface to adjust the pitch and timbre of the changed speech.

The key principle of an effective voice change system is to enable flexible modification both the timbre and the pitch of the speech signals without introducing noticeable distortions. This is realized by decomposing speech signal into the meaningful spectral parameters (e.g. formants and residues), manipulating these parameters, and finally reconstructing the speech signal from them. The voice change aims to provide user-friendly functions to manipulate the spectral components, such that the output speech signal meets the user's requirement.

# Voice Conversion

Voice conversion is defined as the process that transforms the utterance of a source speaker, such that the converted speech sounds like been spoken by another specific target speaker [4]. In other words, it keeps the linguistic information and modifies the non-linguistic information such as pitches, formants, breathes and behaviours. This process is theoretically achievable, as the physical characteristics of each individual are associated with vocal tract and glottal source which can be quantized into numerical values (e.g. spectral features).

An ideal voice conversion system would record all feasible relationships between the source and target speakers via an effective data structure which can be a database, a neural network, a statistical model, a dynamic system or even a hybrid of recognition and synthesis systems. However, as it is not possible to collect sufficient data (e.g. speech signal) to cover all possible acoustic characteristics of each individual, a narrower type of voice conversion system is defined as a function that converts some "important" (e.g. timbre, pitch and prosody) characteristics by modifying temporal/spectral features. Specifically, temporal features contain the sequential information (e.g. vibrato and pronunciation length of a phoneme) of the speech signals, while the spectral features reflect the short-term static information (e.g. pitch, timbre, energy) of speech. Spectral features are considered to convey more physical characteristics of a speaker's individuality and are easier to extract and model [5], thus being the main concentrations of our work.

1.1 Motivation

1.1.1 Voice Change

The study of voice change is motivated by the commercialization project "MagicFriends" [6] that aims to develop the real-time vocal effect engine for the Smartphone in 2012. As the computing ability of certain smartphones (especially the devices with Android OS) at that time is very limited, the target vocal effect engine is expected to be efficient, which enables running the proposed algorithm on the majority of these Android devices.

1.1.2 Voice Conversion

The study of voice conversion is a further step of voice change because it aims to change a speaker's identity to another one. This research was motivated by a specific type of real-time application (i.e. VOIP). With a voice conversion system that matches the quality standards, it would be valuable to develop it in real-time without noticeable latencies and costly computational burdens.

1.2 Aim and Objectives

1.2.1 Aim – Voice Change

The aim is to develop an efficient voice change system which does not require high computation load as well as keeping sufficient speech quality. In addition, the proposed system aims to provide higher flexibilities for timbre modifications, such as formant scaling.

The aim is broken down into the following research objectives,

(1) To identify a proper scheme to efficiently realize a high-quality frequency/pitch scale modification (for real-time applications).

(2) To design an efficient and flexible formant scaling scheme to enable manipulating the spectral envelope to achieve the desired vocal effects.

1.2.2 Aim – Voice Conversion

The aim is to develop an efficient voice conversion system that does not require buffering a long frame of the speech signal. Moreover, the proposed scheme is expected to achieve both high quality and sufficient perceptual identity (as the target speaker).

The aim is broken down into the following research objectives,

(1) To review the existing solutions to voice conversion and study the characteristics (e.g. pros and cons) of them.

(2) To design an effective and efficient frame-level formant conversion scheme to achieve high perceptual quality and identity.

(3) To develop a low-latency clustering filtering scheme to correct the discontinuity caused by inaccurate classifications to further enhance the quality.

1.3 Contributions

1.3.1 Voice Change

As the main concern about the voice change system is efficiency, the proposed solution introduced low-complexity frequency/pitch scaling algorithms that can largely reduce the computing burden. In addition, an effective formant warping scheme was proposed that enables flexibly adjusting the shape of formant envelope. As the proposed solution only requires short and fixed buffering length, it is expected to work in real-time. In addition, the proposed algorithms were implemented in C language for the commercialization products (i.e. PC and Smartphone applications).

1.3.2 Voice Conversion

The proposed solution intends to contribute with a step towards a complete voice conversion system without introducing high computational loads. In order to do so, the thesis introduces a continuous frequency warping and magnitude scaling scheme based on a Finite Mixture Model (Gaussian Mixture Model is used in this thesis). Furthermore, to improve the perceptual quality of converted speech without large computational overheads, a trajectory limitation (e.g. spectrum filtering) scheme has been developed to remove the discontinuity between adjustment frames.

1.4 Thesis Outline

This remaining content of this thesis is divided into six chapters:

***Chapter 2*:** The background of voice changing applications are presented and discussed, as well as the basic concepts that are critical to understanding the general voice change/conversion system framework.

***Chapter 3*:** This chapter describes an efficient real-time frequency scale modification solution that achieves low computational complexity and guarantees a high perceptual quality.

***Chapter 4*:** A novel voice change system is presented. It combines the real-time frequency scale modification solution (Chapter 3) and an effective formant scale modification, which enables flexibly adjusting the speaker's perceptual identity.

Chapter 5**:** This chapter reviews the existing voice conversion systems with discussions on pros and cons of different approaches.

Chapter 6**:** This chapter presents a voice conversion system using a clustering model – Gaussian Mixture Model and frequency warping techniques. In addition, the quality of the output speech is further improved using a trajectory regulation scheme.

Chapter 7**:** The summary of contributions, the suggestions for future work and the conclusions of this thesis are given in this final chapter.

Chapter 2

Background

2.1 Speaker Identity and Identification

Speaker identification is a speech research area that determines the identification of an unknown speaker by comparing it with a set of known speakers. In particular, speaker identification finds the speaker whose vocal characteristic (speaker identity) is the closest to that of the unknown speaker.

Humans can easily recognize people by hearing their voices. Especially, if the familiarity to a subject speaker is high, even with a short non-linguistic speech sample, it is sufficient for a human to recognize the identity. The unattended process of identifying a speaker with speech analysis and decision making via machines is known as automatic speaker recognition [7]. The key component of a successful automatic speaker identification system is to extract the discriminant features that facilitate the identification.

Speaker identifying may also encounter challenges, even for a human listener. In real life, the perceptual vocal characteristic of a speaker is affected by numerous factors, e.g. health condition, mood and channel noise. Thus, it is essential to find robust characteristics that can almost be immune to the impacts of external factors. The mainstream studies believe the robust vocal characteristics are mainly decided by the physical characteristics of vocal tract (vocal cord) and moving behaviour of articulation. From the view of a source-filter model, the vocal cord acts as a pulse signal generator (source) and the vocal tract acts a filter that is affected by the physical attributes (e.g. length, width and elasticity) of the pharynx, the nasal and oral cavities. Furthermore, the speech sounds are broadly classified into two categories, namely voiced sounds and unvoiced sounds. The waveform of voiced sounds is quasi-periodic that corresponds to the vibrations of the vocal cords, while unvoiced sounds are produced by the frictions of mouth (e.g. lips and tongue). The waveform of the unvoiced sounds resembles fugacious noise without periodicities.

According to the state-of-the-art studies [7], the spectral features mainly include the fundamental frequency (F_0) and formants (i.e. spectral envelope), which are highly related to the identity perceptions and can be utilized in voice conversion consequently. F_0 (in Hertz) is the reciprocal of pitch period (in Seconds) and is correlated to the vibrating frequency of the vocal cord. The spectral envelope modifies the gains of pulses [8] from the glottis at different frequencies and is the reflection of the physical characteristic of a vocal tract. It is worth noting that the formants (i.e. peaks and bandwidths) in spectral envelope help in distinguishing the vowel phonemes and tell the differences of speaker identities.

In addition to spectral features, there are also some long-term features that affect the speaker identities. For instance, the pitch variation with time can be different even for the same word. This can also be extended to the time variation of sound intensity. Moreover, physiological factors (e.g. stuttering) affect the performance of speaker identification.

2.2 Voice Change

Voice change is a process that transforms the perceptual identity of a speaker such that the processed speech is not perceived as been spoken by the same speaker. As the perceptual identity of a speaker is determined by the formant and the pitch, the voice change system enables manipulating the formant and the pitch independently. Thus a typical voice change system should be able to decompose the speech signal into formant and pitch-related representations, modify the representations effectively and reconstruct the output speech from the modified representations with high perceptual quality.

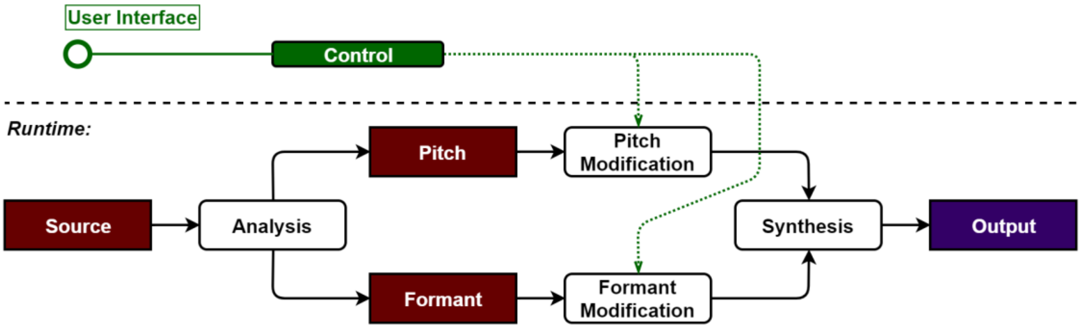


Figure 2-1 Framework of Voice Change

A typical framework for a voice change system is represented in Figure 2-1. As there is no need for converting the speech of a source speaker to a specific target, the system usually does not contain a training stage. Instead, the system modifies the speech based on the instructions from the users. The detailed designs and solutions will be further discussed in 2.4 and Chapter 4.

2.3 Voice Conversion

Voice conversion aims to modify a source speaker's speech identity, such that they are perceived as if they are spoken by a specified target speaker. In particular, the voice conversion systems that are discussed in this thesis only change the non-linguistic information such as pitch, formant and rhythm without modifying the linguistic information (e.g. language content) of the original speech.

A typical framework for a voice conversion system is represented in Figure 2-2 and it usually contains two stages of operations: an offline training stage and a runtime conversion stage.

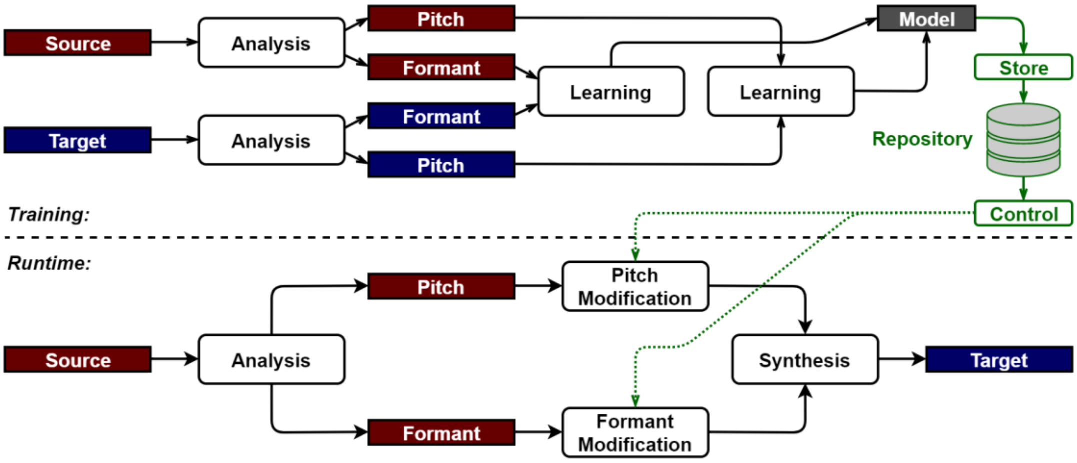


Figure 2-2 Framework of Voice Conversion

During the training stage, the initial step is speech analysis that extracts the acoustic features (e.g. fundamental frequency (F_0) and spectrum envelope) from the source and target corpuses. Then, these features are normalized (e.g. energy normalization and time alignment) to facilitate learning processes. Finally, a mapping function that maps source features to target features is stored in the repository.

During the runtime stage, the source speech signals are transformed into features. Then, the features are converted by the mapping function in the repository to generate the predicted "target" features. For the consideration of obtaining a high quality (e.g. clean and clear) output speech, the speech analysis function must be reversible, such that the speech signals can be recovered from the features.

In addition to the spectral features, the temporal features are also effective and valuable for modelling. Usually, the temporal features contain long-term sequential information that reflects the habit of a speaker. Different frameworks have been considered for temporal features, which are mainly fallen into two categories: 1) state labelling and 2) recurrent modelling. The first category focuses on applying the sequential classification models (e.g. HMM and RNN) to recognize the state (i.e. classification trajectory) of the signal in each frame, and each state maintains a unique mapping function for processing the features. The second category uses a recurrent model (e.g. RNN) to extract the higher level representations from the spectral features. Although the temporal information is interesting and valuable for modelling, it may be not suitable for some systems with low-latency constraints. For instance, if the system intends to model and transform the temporal variants of pitches, it must know the duration of each word, such that the pitches can be re-programmed to sound like the target. Obviously, monitoring the duration of a word largely increase the latency. As the work in this project focuses on the low-latency and real-time applications, the solution of modelling and converting of temporal features is limited to a simple non-recursive strategy that requires low-latency buffers.

2.3.1 Speech Analysis and Synthesis

The speech analysis and synthesis modules are essential in a voice conversion system, while the two modules are the inverse functions to each other. In particular, the analysis module is designed to decompose the speech signals into different components (e.g. F_0, formant, aperiodicity) to enable flexible modifications; and the synthesis module aims to recover the speech signals from the independent components. The basic criterion for effective analysis/synthesis modules is that, without voice conversion, the synthesized speech signals from the analysed spectral features should maintain a similar perceptual quality as the original speech signal. Meanwhile, this criterion is essential for voice conversion: the synthesized speech signals from the **modified** spectral features should keep a sufficient perceptual quality (i.e. without audible artefacts). The proposed voice conversion solution will not focus on developing the analysis and synthesis modules. Instead, an off-the-shelf approach is employed to achieve a reasonable conversion quality.

A VOCODER is a pair of modules that provides both the analysis and synthesis functionalities, especially for speech signals. During the early stage, the common VOCODERs used predominantly time-domain methods (e.g. PSOLA [9]). Later, the source-filter modelling methods (e.g. Linear Predictive Coding (LPC) [10], STRAIGHT [11]) and sinusoidal modelling methods (e.g. Harmonic + Noise Model (HNM) [12], Harmonic/Stochastic Model (HSM) [13]) came out to address the limitations, such as improving efficiency and flexibility of the pure time domain approaches. Note that this thesis will ignore the phase vocoder [14] for comparison. The rationale is that, although the phase vocoder can be used for time/frequency scaling, it is a general technique for audio signals, which does not utilize the characteristics (e.g. pitch) of speech signals. The following content of this section will provide a brief discussion on different types of VOCODERs with classic examples.

2.3.1.1 Time-Domain Example: Pitch Synchronized Overlap and Add (PSOLA)

PSOLA is a time-domain technique that decomposes the signals using the synchronized windowing method and recovers them by overlap and add (OLA), as shown in Figure 2-3.

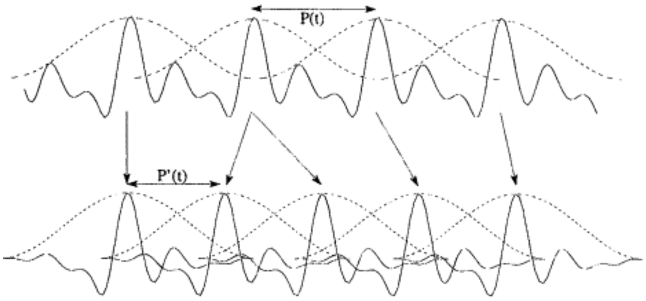


Figure 2-3 Pitch Synchronized Overlap and Add [9]

PSOLA divides the speech signals into small overlapping segments that are centred at the peak samples and with the duration of two pitches. PSOLA enables scaling the pitch of the signal by moving the segments further apart (to decrease the pitch) or closer together (to increase the pitch); scaling the formant by shrinking down the segments (to increase the formant) or expanding up the segments (to decrease the formant); and scaling the time duration by repeating the segments (to increase the duration) or eliminating the segments (to decrease the duration). The PSOLA is an effective VOCODER with reasonable computational costs. However, as the original PSOLA is operating in the time-domain, thus it cannot flexibly manipulate the spectrum in frequency domain, so that the formant scaling capability is restricted.

2.3.1.2 Source-filter Modelling Example: Linear Predictive Coding (LPC)

The source-filter model analogizes the production scheme of the vocal speech, which assumes a short-term speech signal is composed of an excitation signal (impulse train) and a filter, as shown in Figure 2-4. In particular, the frequency response of the filter is considered as the resonance characteristic (formant) of a vocal tract and the excitations (residues) are the pulses generated from vocal cords.

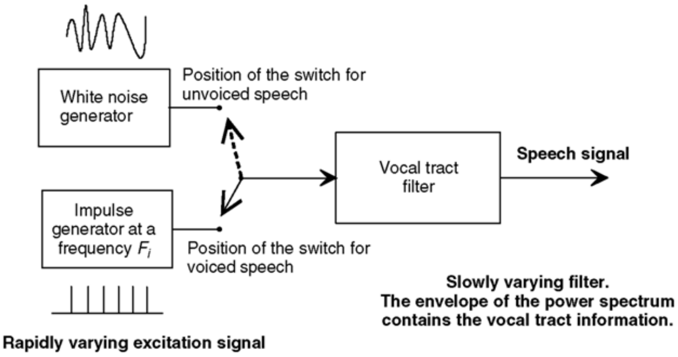

White noise generator
Position of the switch for unvoiced speech
Vocal tract filter
Speech signal
Impulse generator at a frequency F_i
Position of the switch for voiced speech
Slowly varying filter.
The envelope of the power spectrum contains the vocal tract information.
Rapidly varying excitation signal

Figure 2-4 Linear Predictive Coding [10]

Linear Predictive Coding (LPC) is a light-weight approach in the source-filter models, where the filter is limited to be a linear time-invariant all-pole structure (i.e. by estimating the coefficients of an autoregressive model, AR model), and the excitations are obtained by minimizing the amount of energy (i.e. summed squared error loss). LPC enables scaling the formants of the signal without affecting the duration and the pitches by manipulating the all-pole filters. In particular, as the conjugate poles bring resonances at certain frequencies, the angular coordinates and the radial distances of these complex poles reflect the spectral formants. An effective method is to move these poles to change formants. Moreover, other methods, such as converting the all-pole filter into a spectrum representation and manipulate it (Section 4.2.2), can also adjust the formants with even better quality. However, the LPC itself does not have the ability to scale the duration or the pitch of speech signals. Thus, the time-domain approaches (e.g. PSOLA) are usually jointly used to adjust the duration and pitch of the signal.

2.3.1.3 Sinusoidal Modelling: Example: Harmonic + Noise Model

The sinusoidal modelling analyses the frequency-domain spectrum of short-term speech signals by decomposing the spectrum into harmonics (i.e. sinusoids) and noise components. Usually, sinusoidal models are also known as the frequency-domain methods since the harmonic and formant detection is directly carried out in the frequency domain. This type of model has a weak relationship to the speech production mechanism, but also it is valid and valuable from the view of auditory perception [15].

Harmonic + Noise Model (HNM) assumes speech signals are composed of a harmonic part and a noise part, as shown in Figure 2-5. The harmonic part accounts for the quasi-periodic components while the noise part accounts for its non-periodic components (e.g., fricative or aspiration noise).

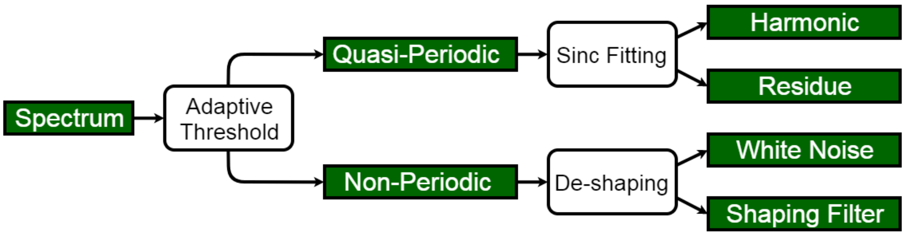


Figure 2-5 Harmonic + Noise Model

The two components are split in the frequency domain by an adaptive threshold, referred to as maximum voiced frequency F_u. The lower band of the spectrum (below F_u) is assumed to be represented solely by harmonics, while the upper band (above F_u) is represented by a modulated (i.e. via an energy envelope and a low-order AR shaping filter) Gaussian noise. From the perspective of voice conversion, the energy and frequency of each harmonic can be manipulated to adjust the formant and pitch, and the coefficients of the AR filter can be adjusted to transform the noise part.

The significant benefit of HNM is that it decomposes the speech signal into compact parameters that are convenient for speech synthesis. In addition, compared to the linear predictive coefficients (AR model) obtained from LPC, the HNM's sinusoidal coefficients are inherently stable (no recursive modelling in the harmonic part). Thus, the mapping model can be estimated with fewer restrictions (i.e. subject to a stable system).

2.3.1.4 Multi-Band Excitation Model Example: STRAIGHT

For the voice conversion solution proposed in this thesis, an efficient implementation – WORLD of the STRAIGHT VOCODER is chosen [11]. "STRAIGHT" stands for "Speech Transformation and Representation using Adaptive Interpolation of weiGHTed spectrum" which also is based on the source-filter model. STRAIGHT works as a high-quality vocoder that analyses and separates the speech into three independent components: a smooth spectrogram free from periodicity in time and frequency, a fundamental frequency (F_0) contour, and a time-frequency periodicity map that captures the spectral shape of the noise and its temporal envelope. Our choice is based on the flexibility STRAIGHT has for feature manipulation and the high quality of speech it produces. For the same reasons, STRAIGHT has been widely used in voice conversion research [16], [17]. Although the other models are not discussed in this thesis, the reader can refer to [18] for a brief description of the other speech analysis and reconstruction models.

2.3.2 Low-dimension Feature Extraction

In the speech synthesis and voice conversion tasks, the waveforms or the spectrums are usually not fed into the system for training, as the dimension of the raw samples or the spectrums are high, which leads to a low training efficiency (e.g. higher computation requirement and longer training time). Considering the sparsity of the speech spectrums [19], a further transformation is usually applied to obtain the compact representations that can facilitate the training procedures. However, there are exceptions, the first is the dynamic frequency warping approach [5] which require the full-size spectrum for calculating the warping function, and the spectrum is manipulated directly. The second exception is the unit selection approaches which require both the full-size spectrums/waveforms and the compact features as inputs. The compact feature is used for an efficient waveform indexing, and the full-size spectrums/waveforms are used for high-quality synthesis.

The following sub-sections give an overview of two popular feature extraction methods which are widely used in voice conversion and speech synthesis.

2.3.2.1 Line Spectral Frequency (LSF)

LSF is a derivation of LPC. As LSF maintain several nice properties (e.g. it is less sensitive to quantization error and is inherently stable for interpolation) that make them suitable for low-bit-rate coding and transmission [20]. For speech processing, LSF is a good representation of the formant structure, as the elements in an LSF vector correspond to the vocal tract with the glottis closed and open. LSF is widely used in speech synthesis and coding, which achieves better performance than other features (e.g. MFCC-MLSA). However, for voice conversion, the mapping function may not guarantee the converted LSF to be in the ascending order and lying in the area between 0 and π. In addition, the correlations between elements in an LSF vector are large, which increases the difficulty (e.g. overfitting) of modelling the distribution accurately.

The LSF $c_{lsp}(m)$ of a real sequence $x(n)$ is defined as to further decomposition of the LPC polynomial $C(z) = 1 - \sum_{m=1}^{M} c_{lsp}(m) z^{-m}$ (estimated from $x(n)$) into a pair of polynomials – Line Spectral Pair (LSP) that consist of a palindromic polynomial $P(z)$ and anti-palindromic polynomial $Q(z)$, which is $C(z) = 0.5\left(P(z) + Q(z)\right)$. The two polynomials are estimated as shown in equation (2-1),

$$\begin{aligned} P(z) &= C(z) + z^{-(M+1)} C\left(z^{-1}\right) \\ Q(z) &= C(z) - z^{-(M+1)} C\left(z^{-1}\right) \end{aligned} \qquad \textbf{(2-1)}$$

Due to the fact that all roots of P and Q are conjugated and located on the unit circle, it is only necessary to specify the arguments (angle) ω within range $(0,\ \pi]$ (within the upper semicircle) to represent LSP which is named as line spectrum frequencies (LSF) as shown in Figure 2-6.

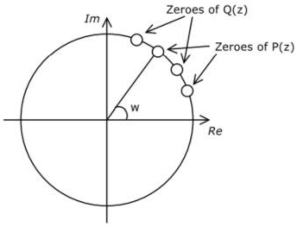


Figure 2-6 Polynomial of Line Spectrum Pairs

2.3.2.2 Mel Cepstral Coefficients (MCC)

Cepstral analysis is a technique widely used in periodic/quasi-periodic signal processing. Due to the periodicity of speech signals, a spectrum $X(f)$ can be view as a spectral envelope $G(f)$ modulated (AM) by a periodic signal (carrier) $H(f)$.

$$X(f)=H(f)\times G(f) \tag{2-2}$$

By taking the logarithm of the spectrum, the envelope $G(f)$ and the carrier are assumed to be linearly repeatable.

$$\log\left(X(f)\right)=\log\left(H(f)\right)+\log\left(G(f)\right) \tag{2-3}$$

The rationale is that the fundamental frequency of $\log\left(H(f)\right)$ is assumed to higher than the vast majority of the frequency components in $\log\left(G(f)\right)$. Thus, by taking the normalized FFT/DCT on $\log\left(X(f)\right)$, it is possible to design a filter that approximately extracts the spectral envelope $G(f)$ and the periodicity of the original speech signal $H(f)$. In fact, both $G(f)$ and $H(f)$ have overlapping components within the high frequency, but it is sufficiently effective for speech applications.

In voice conversion, the cepstral analysis is usually applied to the spectral envelope obtained from VOCODERs, which seems to be contradictory from its original motivation. There are two reasons; firstly, the spectral envelope achieved by VOCODERs is usually more accurate than that by cepstral analysis; secondly, the cepstral coefficients maintains low correlations between each other (high diagonal covariance, low cross covariance), which facilitate modelling the acoustic information using GMM and HMM.

In fact, cepstral analysis is a rule-based non-linear feature extraction (coefficient decorrelation) technique that is speaker independent; other training-based feature extraction methods can also be considered, such as Principal Component Analysis (PCA) and Deep Neural Network (DNN). Note that the training-based methods usually require the large database to avoid overfitting. For instance, if the decorrelation matrix is only trained using the database of speaker A, it is unlike to be valid for any other speaker.

Mel Cepstral Coefficients (MCCs) are a special type of cepstral coefficients calculated on the Mel frequency scale instead of the linear frequency scale. Mel scale emphasizes the low-frequency components rather than the high-frequency components, which matches the perceptions of human ears [21]. From the view of spectral distortion, MCCs increase the penalty of low-frequency distortion when encoding spectrums, therefore the decoded speech signals from MCCs will maintain less distortion in lower frequency but also more distortion in higher frequency.

An MCC c_α of a real sequence $x(n)$ is defined as the inverse Fourier transform of the frequency warped logarithmic spectrum as shown in equation (2-4).

$$\begin{aligned} \log X\left(e^{j\omega}\right) &= \sum_{m=-\infty}^{\infty} c_\alpha(m) e^{-j\beta(\omega;\alpha)m} \\ \beta(\omega;\alpha) &= \tan^{-1}\frac{\left(1-\alpha^2\right)\sin\omega}{\left(1+\alpha^2\right)\cos\omega - 2\alpha} \end{aligned} \tag{2-4}$$

Here, α is the warping factor (α = 0.33 for Mel frequency scale), $X\left(e^{j\omega}\right)$ is the Fourier transform of $x(n)$, $\beta(\omega;\alpha)$ is the frequency warping function defined as the phase response of an all-pass filter. Commonly, the estimation of MCC is realized via an iterative/recursive algorithm [21].

2.3.3 Frame Alignment and Non-parallel voice conversion systems

From the view of minimizing the conversion error, it is not feasible to train a voice conversion system without source-to-target feature pairs. Therefore, the speech features from the source and target speakers are required to be aligned to construct the one-to-one relationships. Due to the difference between the parallel corpus and the non-parallel corpus, the alignment techniques are also different.

2.3.3.1 Parallel Corpus

In the case of using parallel corpus that the utterances from two speakers are with same linguistic content, the alignment of speech signals is quite intuitive. A popular technique for frame alignment is Dynamic Time Warping (DTW) [22], [23] that searches for the warping trajectory by minimizing the spectral distance between the source speaker's features and the target speaker's features.

2.3.3.2 Non-Parallel Corpus

In some scenarios, the linguistic content in the source corpus and the target corpus are totally different. In this case, there are two types of solutions to realize voice conversion. The first one is to iteratively find the match frames from source to target, and the voice conversion system is trained based on the high-score matched frames. However, the key problem of this type of solutions is high complexity. As the complexity of searching the whole matching space is $O(mn)$, where m and n are the total numbers of frames of source and target. It becomes impractical if m and n are large. The second type is to train separate generative models for both the source speaker and the target speaker. The generative model can store states for different phonemes. First, the source speaker's speech is converted to a state by the generative model. Then, the state is used for synthesizing the speech of the target speaker. The key challenge in this type of solution is to match the states of source and target. To this end, a practical solution can be phoneme recognition.

2.3.4 Prosody Conversion

In addition to the spectral envelope, another factor that is critical to the perceptual speaker identities is the prosody. The prosody includes fundamental frequencies (F_0), intonations and the duration. In general, the majority of the existing voice conversion systems consider converting the fundamental frequencies without modifying the intonations and the duration. The rationale is that the intonations and the durations can be influenced by the content (semantic) of the discourse, the mood of the speaker and other paralinguistic factors. If the size of the training corpus is relatively small, the utterances generated by the overfitting voice conversion system may sound unnatural.

The most popular technique to convert the fundamental frequencies is to map the mean and variance from the source F_0 distribution to the target F_0 distribution. This technique operates on each frame without the requirements of the context information, the estimated target F_0 of a frame n is given in equation (2-5),

$$\hat{F}_y[n] = \frac{\sigma_y}{\sigma_x}\left(F_x[n] - \mu_x\right) + \mu_y \tag{2-5}$$

Here, $\hat{F}_y[n]$ is the estimated target F_0, $F_x[n]$ is the source F_0, μ_x and μ_y are the means, and σ_x and σ_y are the standard deviations of the source and the target speakers' F_0 distributions. This mapping function enables the converted mean F_0 and the variation range from the source speaker to approximate that of the target speaker. Note that the mean/variance mapping technique can also be applied to the log scale of the fundamental frequencies, which were stated to achieve similar performance. In addition to the linear transformation, some nonlinear extensions, such as polynomial regression, Gaussian mixture regression and local linear regression, are utilized for converting the fundamental frequencies at frame-level.

In order to manipulate intonations and durations [24], [25], two types of techniques are usually utilized. The first type trains DTW functions to match the variable-length speech segments (usually at syllable level) from source to target, which enables modification of the duration of each speech segment. The second one encodes the variable-length speech segments (from source and from target) into fixed-dimensional vectors and enables decoding the speech segments from the encoded vectors, which enables generating the target speech segments (with more reasonable lengths and F_0 contours) from the encoded vectors of the source speaker.

2.3.5 Evaluation Metrics

The aim of a plausible voice conversion system is to convert the source speaker's speech to mimic the perceptual identity of the target speaker. A successful system should output the speech with a natural and intelligible quality and a distinguishable identity.

To the speech processing applications that target at satisfying the humans' auditory perception, the evaluation section usually contains two sub-sections, namely the objective and subjective measures. The objective measure is carried as a rule-based quantitative testing of rigorously evaluating the performance, while the subjective measure is a statistic-based quantitative testing that reflects the averaging subjective experiences. Both of the two measures are essential, as the objective optimal solution may not satisfy the humans' ear, and the subjective optimal solution can lead to bias if the testing group is insufficiently large.

2.3.5.1 Objective Test

Mel-Cepstral Distortion (MCD) [26] is a widely-accepted testing metric for objective evaluations, especially for applications like text-to-speech synthesis and speech manipulation. The rationale of using MCD is that the MCD value is correlated to the subject test according to statistic results [26]. It calculates the difference between the converted frame and target frame in the cepstral domain as shown in equation (2-6).

$$\epsilon_{MCD}\left[dB\right]=\frac{10}{\ln 10}\sqrt{2\sum\nolimits_{q=1}^{Q}\left(c_q-\hat{c}_q\right)^2} \tag{2-6}$$

Here, c_q and $\hat{c}_q$ denote the q^{th} order cepstral coefficients that can be MCC, MGC or MFCC, Q is the maximal orders of the cepstral coefficients for calculating MCD; usually, the zero-order cepstral coefficient c_0 is removed for evaluations, as it only reflects the energy that does not make a significant contribution to the speaker identity.

2.3.5.2 Subjective Test

Two popular testing methods are used for voice conversion research, namely Mean Opinion Score (MOS) test [27] and ABX test [28]. The subjective evaluation maintains the advantage in fitting the humans' perception and has the value in commercializing the designed system.

# Mean Opinion Score

The MOS test targets to quantize the satisfaction into a single rational number (score), typically in range 1 – 5. For voice conversion system, the listeners are required to rate the naturalness of the converted speech using the scores: 1) bad, 2) poor, 3) fair, 4) good, and 5) excellent. Different utterances are evaluated by a single listener, and the MOS scores of the utterance are averaged to achieve the reliable performance evaluation. A significant disadvantage of the MOS test is to design the upper and lower boundaries. In other words, the listener should at least know the standard of the score 1) bad and the score 5) excellent. Otherwise, the listeners cannot determine what does the score 3) fair mean.

# ABX Test

In comparison to the MOS test, the ABX test is quite straightforward for listeners, which does not need to pre-quantize the scores or the upper/lower boundaries. In particular, the ABX test makes the listeners compare the converted utterances (A by the first system and B by the second system) and choose the one that sounds more similar to the target utterance. In order to prevent the bias, the utterance A and utterance B are shuffled before being played to the listeners. In ABX, the preferences are often divided into 3 scales: 1) preferring A than B, 2) preferring B than A, and 3) no preference.

2.4 Summary and Conclusions

This chapter reviews the background knowledge and technologies that are related to the speech manipulation applications, which are the solid foundations for developing the following voice change and voice conversion solutions.

Chapter 3

Real-time Frequency Scale Modification

Frequency scale modification is a special type of voice change application, which scales the frequency of each harmonic component to achieve the perceptual pitch shifting and formant shifting as shown in Figure 3-1.

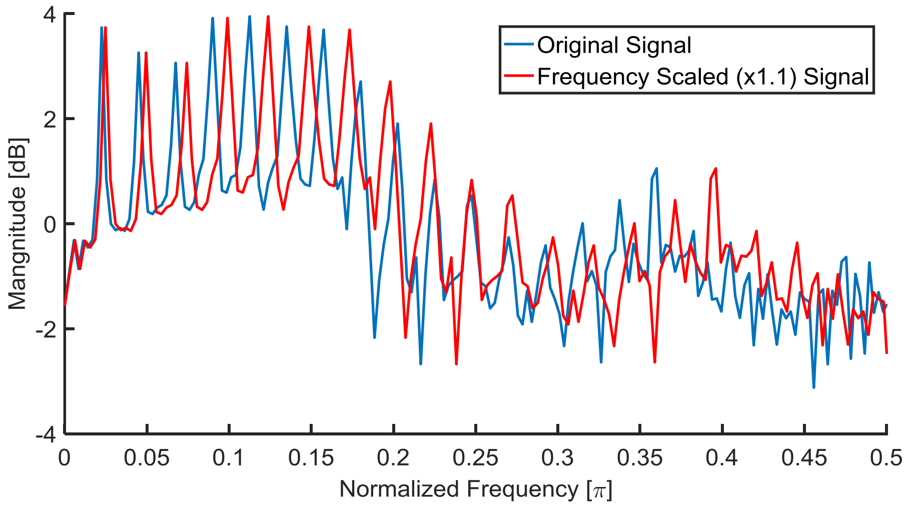


Figure 3-1 Frequency Scale modification

The real-time frequency scaling system takes a frame of a fixed (time) length from the input speech signal, which is buffered in the audio interface, and then processes it. After scaling the frequency component, the audio interface outputs the modified frame with the same length. A conceptual system-level diagram is shown in Figure 3-2

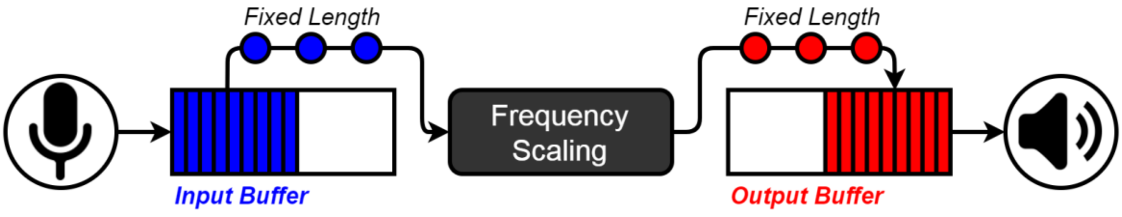


Figure 3-2 System Diagram of Frequency Scale modification

The requirement is that the output frame must retain the same length as the input frame to avoid discontinuity or backlogging delay. For instance, a frequency scale modification system can be realized by a resampling process and a time scale modification process. An example of a frequency scale modification is shown in Figure 3-3.

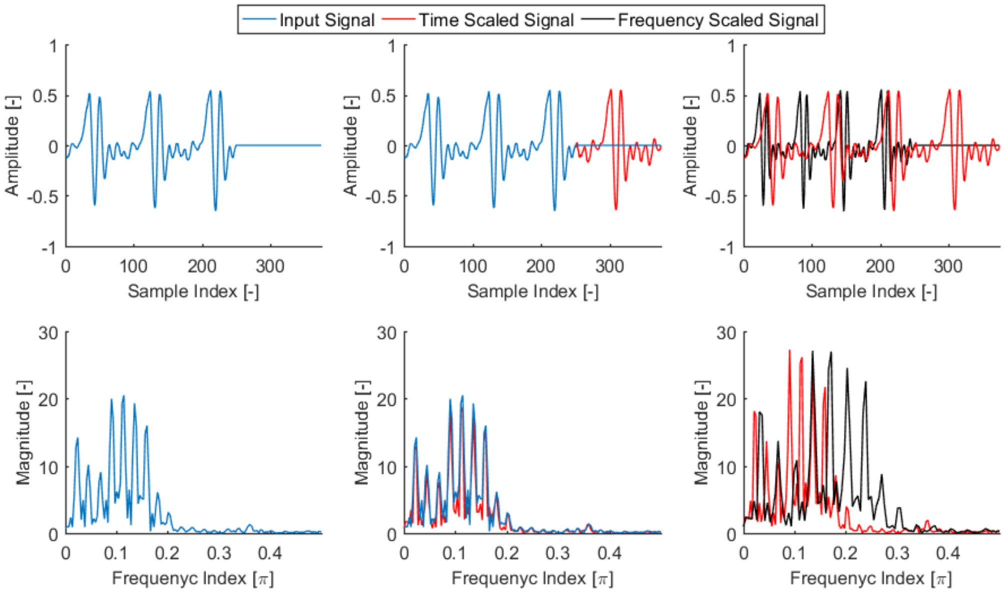


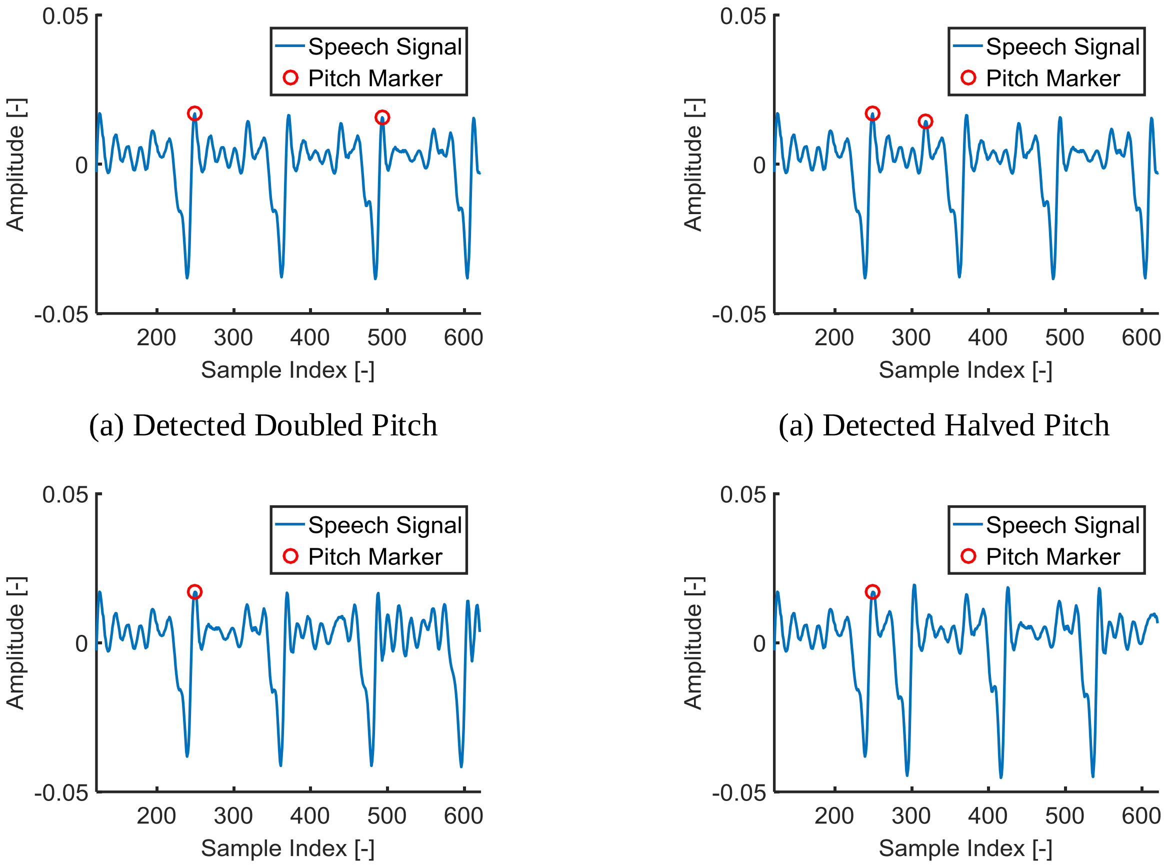
Figure 3-3 (Left) Input, (Mid) Time Scale modification, (Right) Frequency Scale modification

The time scale modification extends the length of the speech signal by a given scaling factor and the resampling process restores the length of the time scaled signal into the length of the original input signal. The resampling technique can be implemented via an off-the-shelf technique (e.g. a polyphase filter [29] or a polynomial interpolation). In this thesis, linear interpolation is used for resampling for simplicity. In contrast, the time scale modification will be a key design which will be presented in the following sections of this chapter.

3.1 Pitch Alignment for Overlap-and-Add

Overlap-and-Add (OLA) is a mature technique to extend/compress the length of the quasi-stationary signals (e.g. speech signals) [30]. In order to scale the signal without introducing significant distortions (e.g. discontinuity), pitch alignment is an essential step for OLA to enable extending/compressing the pitch periods by integer times.

The accuracy of the pitch alignment for OLA does not need to be as “accurate” as some other applications (e.g. pitch tracking). Specifically, OLA is robust to the doubled pitch alignment.



(a) Detected Doubled Pitch

(a) Detected Halved Pitch

(c) Doubled Pitch OLA Result

(d) Halved Pitch OLA Result

Figure 3-4 OLA results by Doubled Pitch and Halved Pitch Detections

As shown in Figure 3-4 (a, c), if the actual pitch period is 120 samples and the estimation is 240 samples, OLA works fine; i.e. the scaled signal is still quasi-stationary. In contrast, the halved pitch (as shown in Figure 3-4 (b)) alignment will cause distortions (as shown in Figure 3-4 (d), the OLA signal becomes strongly non-periodic near the pitch marker). This is because the non-periodicity will have different harmonics distribution compared to the original quasi-periodic signal.

In this section, two types of conventional pitch detection functions (i.e. 512-Sample Radix-4 Autocorrelation Function and Average Magnitude Difference Function) are compared to an efficient pitch alignment solution – Peak Picking Function. The analysis aims at verifying if a high computational load pitch detection algorithm is really necessary for a time/frequency scale modification process.

3.1.1 Complexity

Pitch Detection: Radix-4 Autocorrelation Function (R4AF)

The computation complexity of R4AF consists of 4 parts: 1) Radix-4 FFT, 2) square function, 3) Radix-4 IFFT, and 4) peak picking function, as shown in Table 3-I.

Table 3-I Complexity of R4AF

Function	Radix-4 FFT	Square	Radix-4 IFFT	Peak Picking	Total
Multiplication	3Nlog$_2$N/8	2N	3Nlog$_2$N/8	0	3Nlog$_2$N/4 +2N
Addition	Nlog$_2$N	N	Nlog$_2$N	0	2Nlog$_2$N+N
Comparison	0	0	0	2N	2N

Pitch Detection: Average Magnitude Difference Function (AMDF)

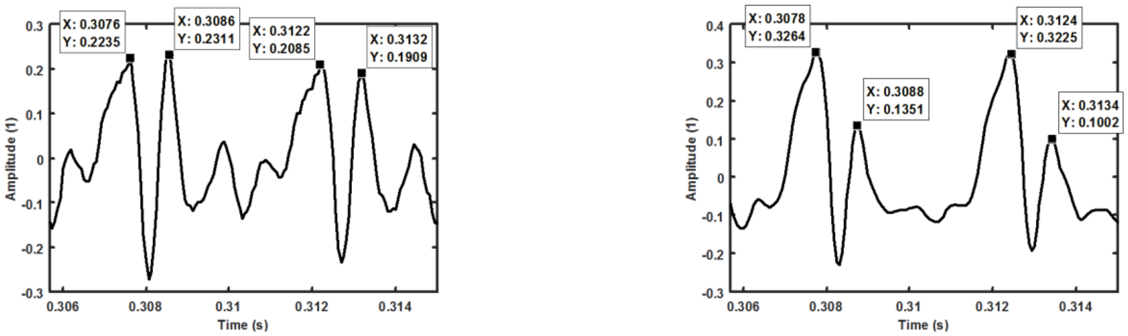
The computation complexity of AMDF pitch detection consists of 2 parts: 1) AMDF function, and 2) peak picking function, as shown in Table 3-II.

Table 3-II Complexity of AMDF

Function	AMDF	Peak Picking	Total
Addition	N^2/2	0	N^2/2
Absolute	N^2/2	0	N^2/2
Comparison	0	2N	2N

Pitch Alignment: Peak Picking Function (PPF) on filtered signal

Before applying the peak picking function, the speech signal is filtered using a lowpass filter to remove disturbances of the high-frequency components. As shown in Figure 3-5, the peak in a pitch period of the lowpass filtered signal is more significant than that of the unfiltered signal.



(a) Original Signal (b) Filtered Residue

Figure 3-5 Peak Detection Error and Correction (by low a pass filter)

The cut-off frequency of the lowpass filter is calculated according to the prior knowledge of the vocal characteristic of the speaker. Particularly, a speaker is required to say several phoneme-rich sentences (e.g. “she had your dark suit in greasy wash water all year” [31]) to collect the regular pitch range of this speaker. Then, the lowpass filter is designed to cover 0% ~ 95% of the pitch distribution. For the filtering design, a 2-order Chebyshev II model is used. The rationale is that an IIR filter can achieve a higher cutting off slope with a much lower order system compared to an FIR filter. The IIR filter usually causes phase distortions, but the phase distortion will not affect the accuracy of picking peaks. Then, the peak picking function is applied to the filtered signal to extract two peaks from the frame. The interval between the locations of the two peaks is assumed to be integer-times pitch period. In order to further enhance the robustness of extracting peaks, the peak picking is applied both the original signal and the negative signal. By comparing two intervals, the larger one is selected, which reduces the possibility of taking a halved pitch.

The computation complexity of PPF consists of 2 parts: 1) lowpass filtering, and 2) peak picking function, as shown in Table 3-III

Table 3-III Complexity of PPF

Function	Lowpass Filter	Peak Picking	Total
Multiplication	5N	0	5N
Addition	4N	0	4N
Comparison	0	4N	4N

It is straightforward that the computational complexity of the PPF approach is significantly lower than that of the previous two approaches.

3.1.2 Accuracy

The performance is tested according to the frequency of detecting the **halved** pitches which degenerates the performance of OLA. By using a complex and robust pitch detection algorithm (i.e. BaNa pitch detection [32]), it is feasible to achieve very accurate pitch detection results. For the same sentence, if the detected pitch period (by R4AF, AMDF and PPF) of a frame is smaller than 4/5 of the accurate result (by BaNa), the detection result is wrong. The accuracy of each solution is shown in Table 3-IV.

Table 3-IV Accuracy of Pitch Alignment

Accuracy	R4AF	AMDF	PPF
Sentence 1	96.5%	98.5%	98.1%
Sentence 2	97.8%	98.3%	97.9%
Sentence 3	96.2%	98.4%	97.7%
Sentence 4	95.1%	98.5%	97.5%
Sentence 5	98.8%	99.1%	99.0%
Average	96.88%	98.56%	98.04%

The experimental results show that the PPF pitch alignment scheme achieves similar accuracy as the baseline approaches, which is sufficient for the OLA applications. Moreover, PPF requires less computational complexity compared to the baselines.

3.2 Real-time Frequency Scaling using Intra-Frame PAOLA

In this section and the next section, the OLA using the peak picking function is named as Pitch Alignment OLA (POLA). Theoretically, the signal within a frame (with the length of more than a single pitch period) can be scaled without context information (i.e. previous or next frames). This is because it is feasible to eliminate or duplicate the periodic signal (with a pitch length) to manipulate the length of the entire signal.

To extend the frame length, the frame is first duplicated and shifted by detecting peak interval τ samples (to align the pitch peaks), then the duplicated and shifted frame is added to the original frame using a cross fading function, as shown in Figure 3-6, where the red box denotes the speech signal, and the blue dash line denotes the fading function.

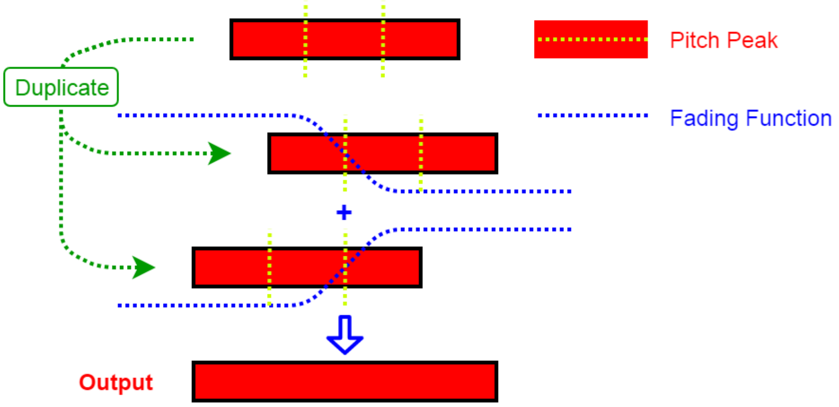


Figure 3-6 Example of Intra-Frame PAOLA

#Example:

For a given frequency scaling factor $1/\eta$, the time scaling factor is equal to *η*. In other words, the target length after time scaling is $w\eta$, where *w* is the length of the original frame. It indicates that the desired scaling length *δ* should be $\delta = w\eta - w$. As the desired scaling length *δ* is usually not the integer times of the detected peak interval *τ*, e.g. $\delta \neq n\tau$, it is usually impossible to extend the signal in the frame by the desired scaling length *δ*. Thus, an idea is to approach the desired scaling length *δ* by control the duplications *n*. In addition, as the real-time applications require seamless outputs, the actual time scaling factor $\hat{\eta}$ must be larger than the desired time scaling factor *η*, such that the lengths of the frequency-scaled signals are always larger than_*w*. To this end, the duplication *n* must satisfy equation (3-1).

$$\arg\min_{n} \quad \delta \leq n\tau, n \in \mathbf{Z} \qquad \textbf{(3-1)}$$

The duplicated peak interval also introduces the excessive length $\varepsilon = n\tau - \delta$. To prevent backlogging the excessive length *ε*, the following paragraph will present a novel **scaling factor correction scheme** to ensure the accumulated excessive length *ε* to be bounded.

In concept, the proposed scheme utilizes the actual scaling length of the previous frame to correct the scaling length of the current frame. For example, the length of a frame is 500 samples, the desired scaling length is 750 samples (scaling factor: x1.5) and the detected peak interval is 100 samples. First, the system calculates the actual scaling length to be 800 = 500 + 3 × 100, and 800 > 750 and 700 < 750. Thus, the excessive length *ε* = 800 – 750 = 50. The excessive 50 samples are directly appended to the next frame with the length of 500 samples. Similarly, the detected peak interval of the next frame is 110. The system calculates the actual scaling length to be 770 = 550 + 2 × 110. Thus the excessive length *ε* = 800 – 770 = 30. There is excessive 30 samples are directly appended to the next frame with the length of 500 samples. Specifically, after appending, if the length is larger than the desired scaling length, the samples of the desired scaling length are directly outputted without PAOLA.

3.3 Real-time Frequency Scaling using Inter-Frame PAOLA

Different from Intra-Frame PAOLA that modifies the signal within a frame, Inter-Frame PAOLA is another proposed approach that supports time scaling by manipulating the consecutive frames. The key principle is to adjust the frame length without changing the stepping length.

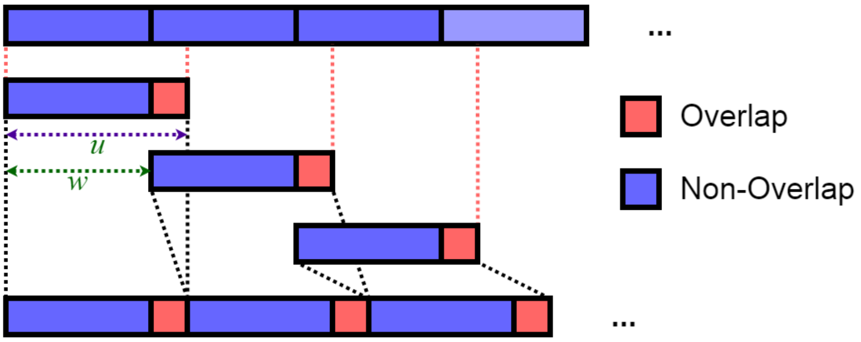


Figure 3-7 Example of Inter-Frame OLA without additional overlapping

For instance, as shown in Figure 3-7, the requirement of a time scaling factor $\eta = 1.2$ is realized by setting the frame length size u to be $u = 1.2w$ (the step size w). By concatenating k frames without any overlapping, the total length is expected to be 1.2kw. As direct concatenation leads to noticeable discontinuities, PAOLA is required. However, if PAOLA is directly applied to these frames (with the length of 1.2w), the length of the actual output is always smaller than 1.2w, this is because PAOLA requires overlapping which always shortens the signal. In order to solve this problem, a compensating length m is added to the frame length 1.2w to ensure the actual output length (after PAOLA) to be larger than 1.2w, as shown in Figure 3-8.

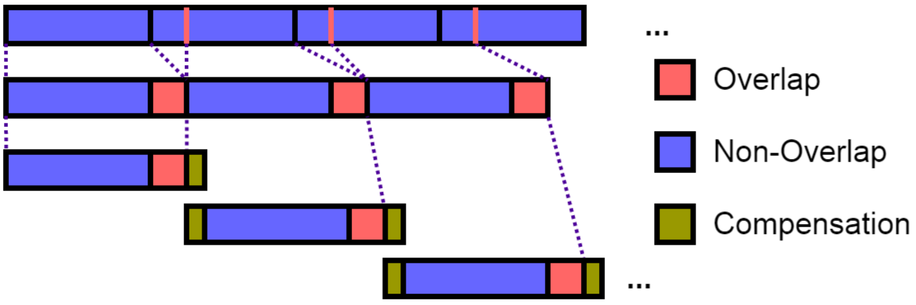


Figure 3-8 Example of Inter-Frame OLA with additional overlapping

In order to pick up at least one pitch peak within m for alignment, the compensating length m is defined as the pitch interval τ. Within the last section (τ samples) of the previous frame and the first section (τ samples) of the current frame, the approach tries to find one peak in each section, and then the two adjacent frames are aligned using the two peaks. After crossfading, the output length is guaranteed to be larger than $1.2w$. The excessive samples are directly discarded to prevent accumulating excessive lengths, as shown in Figure 3-9.

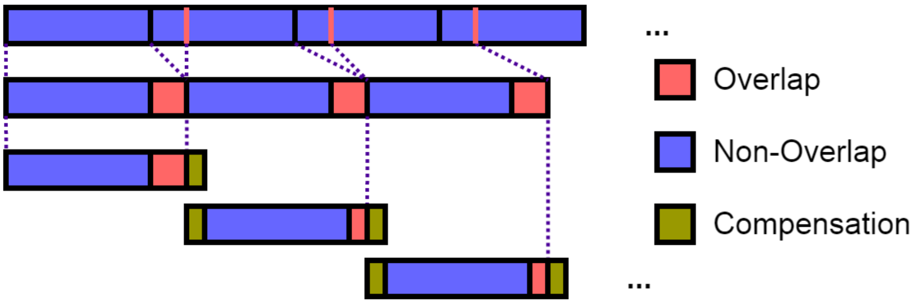


Figure 3-9 Example of Inter-Frame OLA

3.4 Performance Evaluation

The performance of the proposed frequency scaling scheme is compared to the conventional Waveform Similarity and Overlap Add (WSOLA) algorithm [33]. As the frequency scaled speech signal is highly distorted without a reference, the objective metrics/tests are not applicable for evaluating the performance of the proposed system.

To this end, a MOS test was employed for subjective evaluation, where the frequency scaled speech signals (i.e. 8 sentences, modified by WSOLA and Intra/Inter-PAOLA) are played to 5 listeners blindly. The listeners were asked to score the quality of each sentence. The comparative results are provided in Figure 3-10, which presents the averaged quality of all sentences which are scaled by different factors.

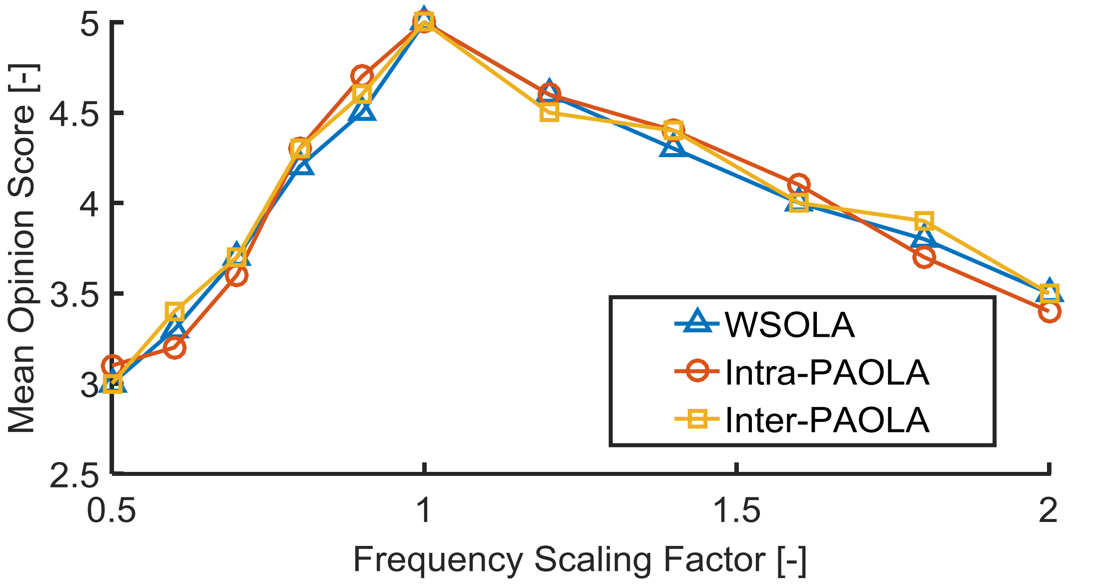


Figure 3-10 Mean Opinion Score for Different Frequency Scale Modifications

The results show that the proposed approaches achieve similar perceptual quality compared to the conventional WSOLA algorithm. As the scaling factor is too far away from 1, the OLA algorithm with frequency resampling causes more noticeable distortion which has been extensively studied in the previous works. According to the analysis from Section 3.1.1, it is straightforward that the proposed approaches largely reduce the computation load as it does not need the cross-correlation process which is required in WSOLA.

3.5 Summary and Conclusions

This chapter develops a computationally efficient pitch alignment algorithm based on a peak picking method. Then, it is applied to the proposed frequency scaling algorithms, i.e. PAOLA, for low-latency real-time mobile applications. The experimental results show that the two PAOLA algorithms achieve similar perceptual qualities compared to the baseline approach WSOLA with reduced computational intensities.

Chapter 4

Real-time Voice Change Systems

This chapter is dedicated to the efficient solution to change the speaker's voice identity. The first section discusses the desired functionality in voice change that can manipulate the perceptual identity of a speaker. Then, an efficient solution is proposed to realize the manipulation for real-time application, which includes a pitch modification and a formant warping technique. Finally, two voice change applications are presented.

4.1 Voice Change Framework

The voice identity change is based on an idea that speech signals contain pitches and formants. A typical speech analysis-by-synthesis (AbS) system [34] is able to decompose speech signals into pitch parameters and formant parameters, which enables changing the perceptual identity by manipulating the parameters.

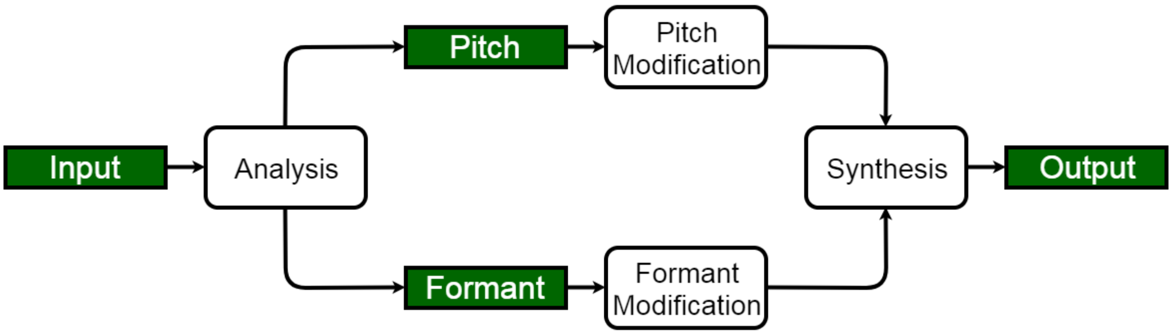


Figure 4-1 Voice Change System

Generally, a voice change system can be divided into four components as shown in Figure 4-1: 1) speech analysis, 2) pitch modification, 3) formant modification 4) speech synthesis. In speech analysis, an analysis system transforms a frame of the speech signal into parameters (e.g. pitch and formants). Then, the parameters are modified by a pitch modification and a formant modification. Based on the modified parameters, the synthesis system re-generates the speech frame as the final step.

4.2 Linear Predictive Coding based Voice Change System

As discussed in Section 2.3.1, Linear Predictive Coding (LPC) is an efficient vocoder that can decompose/reconstruct the speech signals to/from residues and spectral shaping filters, where the residues carry the pitch information and the spectral shaping filters contain the formant information. The LPC-based voice change system is shown in Figure 4-2.

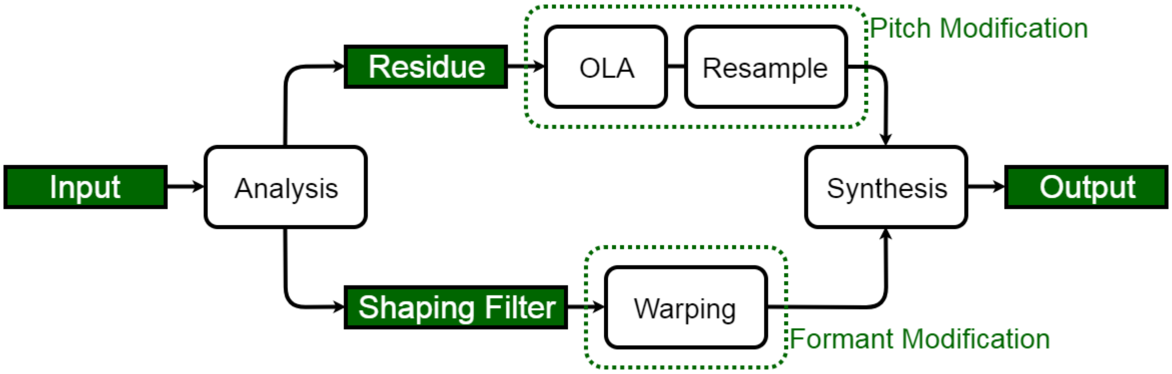
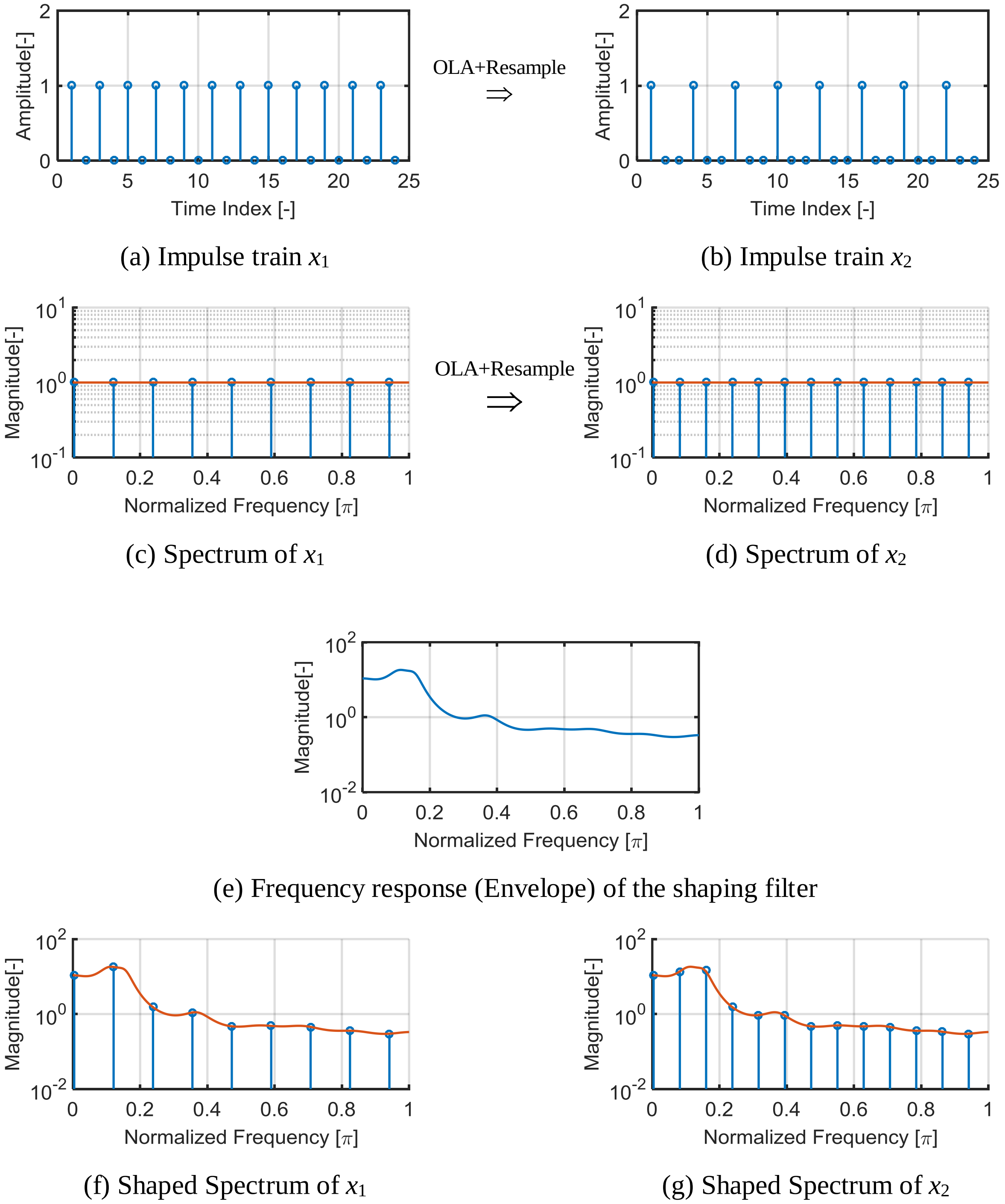


Figure 4-2 LPC-based Voice Change System

4.2.1 Pitch Modification

As been discussed in 2.4, time scale modification with a resampling process can manipulate the frequency components by shifting the locations of harmonics, namely frequency scale modification. The speech signal in the frequency domain can be approximated by multiplying “energy-normalized” harmonics (i.e. impulse stimulations in the time domain) by the formant envelope. If the frequency scale modification is directly applied to the original speech signal, not only the fundamental frequencies but also the formants are modified, which limits the flexibility of the system.

In order to manipulate pitch without affecting formant, the frequency scale modification is only applied to the residue signal instead of the original speech signal. The rationale is that the residue is an approximation of the vocal cord stimulation without formant information. Although it is not strictly true, this approximation is usually accurate enough for voice change applications.



(a) Impulse train x_1

(b) Impulse train x_2

(c) Spectrum of x_1

(d) Spectrum of x_2

(e) Frequency response (Envelope) of the shaping filter

(f) Shaped Spectrum of x_1

(g) Shaped Spectrum of x_2

Figure 4-3 Pitch Scale modification without Formant Scaling

An example is shown in Figure 4-3, if the frequency scaling is only applied to the residue signal without changing the shaping filter, the formant envelope of the speech signal will remain the same, which indicates that the timbre of the scaled signal is unaltered.

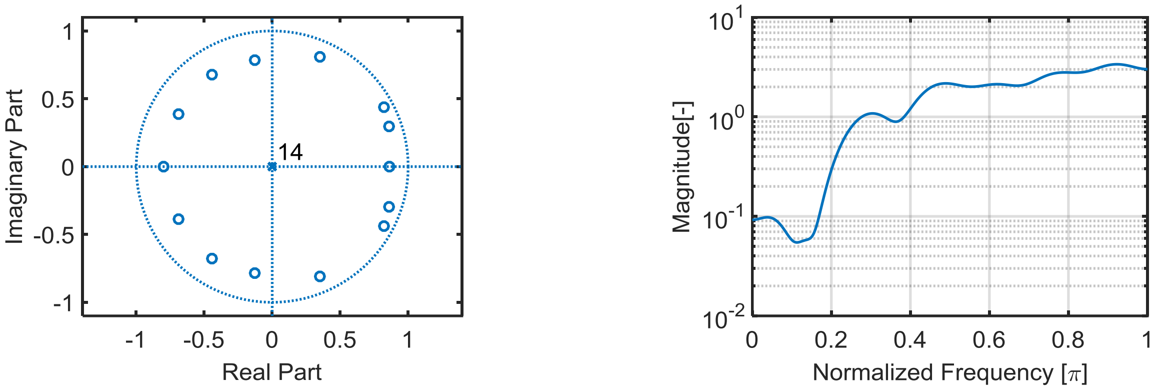
4.2.2 Formant Modification

In addition to pitch scale modification, another essential step is formant scale modification. As shown in section 2.3.1, the formant information is converted into the LPC coefficients. These coefficients formulate a polynomial that is the denominator of an all-pole filtering system which models the resonances of the vocal tract [10]. Each pair of the conjugate roots of the polynomial reflects a resonant frequency of the vocal tract. It has been shown that [35], the frequencies and bandwidths of resonance peaks can be modified by changing the modulus and phase of the conjugated roots. However, directly manipulating the conjugate roots may degenerate the quality of the synthesized speech. The evaluation part (Section 4.4) in this chapter will further compare the different approaches which include directly manipulating the roots.

In this section, a high-quality and flexible formant scale modification scheme is proposed. It is achieved by 1) mapping the LPC coefficients into (spectral) formant envelope, 2) modifying the envelope, and 3) recovering the modified LPC coefficients from the modified envelope.

4.2.2.1 LPC Coefficients to Formant Envelope

Firstly, the formant envelope is achieved by applying a 128-point Discrete Fourier Transform (DFT) on the LPC coefficients as an all-zero filter (Figure 4-4)



(a) LPC coefficients as an all-zero filter $h(z)$

(b) Frequency response of $h(z)$

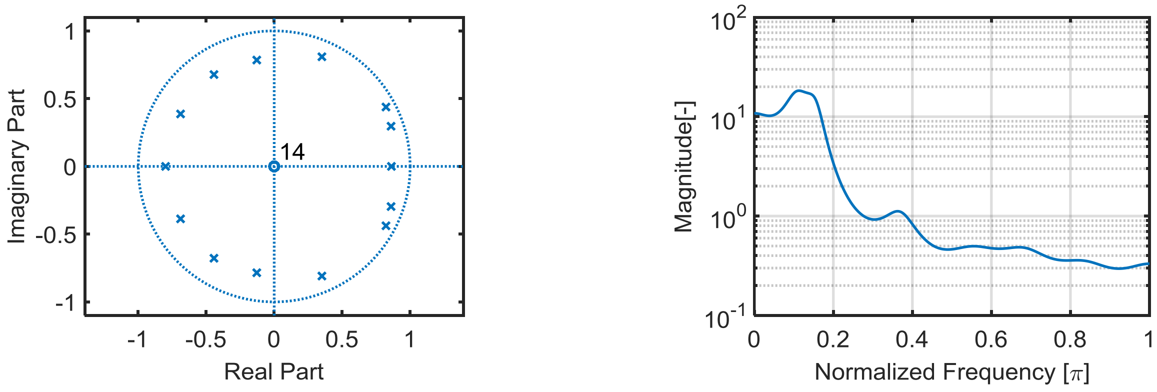
Figure 4-4 Calculate Frequency Response of LPC coefficients as an all-zero filter

The rationale behind this is that processing a signal using an all-zero filter is as same as convolving the signal with the coefficients of the all-zero filer, which is equivalent to multiplying the frequency responses of the all-zero filer and the signal (equation (4-1)).

$$h\left(x\left[n\right]\right)=h\left[m\right]*x\left[n\right]\Leftrightarrow H\left(\omega\right)\times X\left(\omega\right) \tag{4-1}$$

Here, h is the all-zero filter in the time domain, $x\left[n\right]$ is the input signal in time domain, $H\left(\omega\right)$ is the frequency domain representation of the filter and $X\left(\omega\right)$ is the frequency domain representation of the signal.

As the inversion of a Z-domain transfer function corresponds to the inverse response in the frequency domain, the frequency response of the all-pole filer can be calculated by taking the reciprocal of the frequency response of the all-zero coefficients as shown in Figure 4-5.



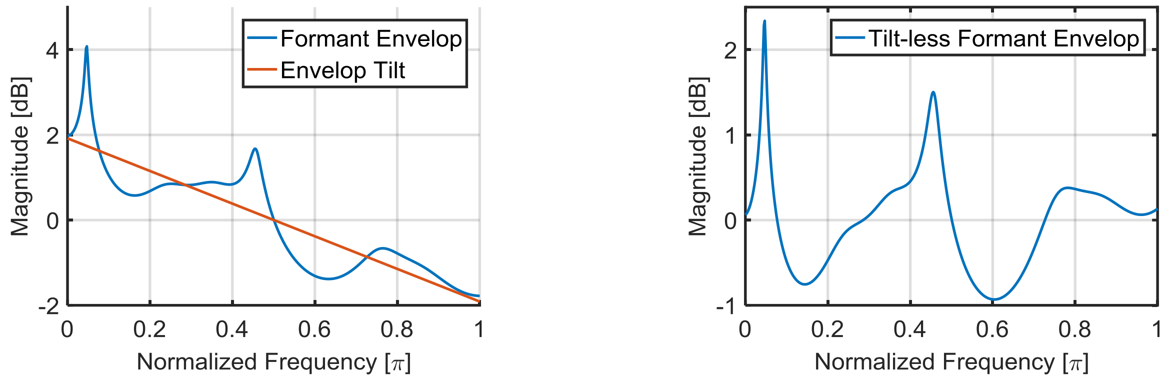
(a) LPC coefficients of $1/h\left(z\right)$

(b) Frequency response of $1/h\left(z\right)$

Figure 4-5 Frequency Response of LPC coefficients as an all-pole filter

4.2.2.2 Formant Modification

The formant modification will include two parts: 1) frequency warping and 2) magnitude scaling, which is applied to the frequency response (i.e. formant envelope) of the filter $1/h\left(z\right)$. The formant tilt is removed before modification and is added back after modification to prevent the unnatural outputs [36]. The formant tilt is extracted by applying an ordinary least square regression on the logarithm-scale modulus of the envelope. Figure 4-6 illustrates an example of the tilt removal procedure.



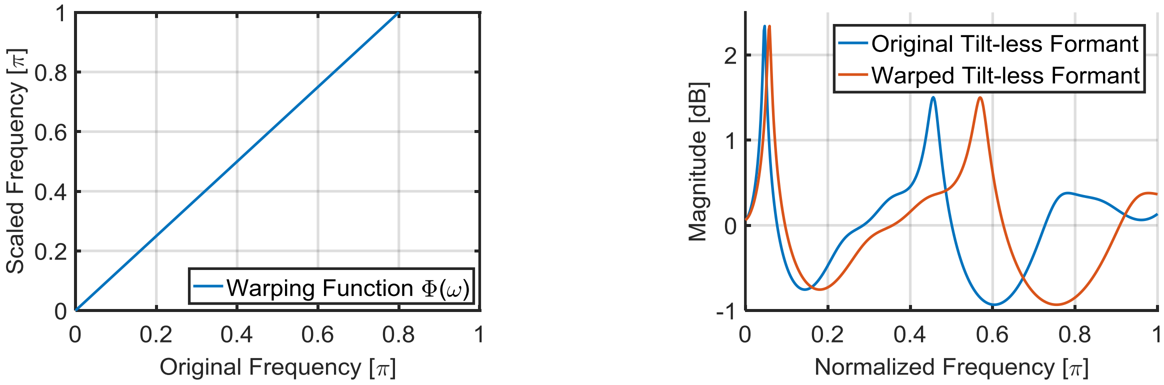
(a) Formant Envelope and its Tilt

(b) Tilt-less Formant Envelope

Figure 4-6 Removing Tilt from Formant Envelope

#Frequency Warping:

Based on the tilt-less formant envelope, the frequency warping can be achieved by moving/scaling the frequency indexes based on the given function $\Phi(\omega)$.



(a) Scaling function $\Phi(\omega)=1.25\omega$

(b) Frequency warped result

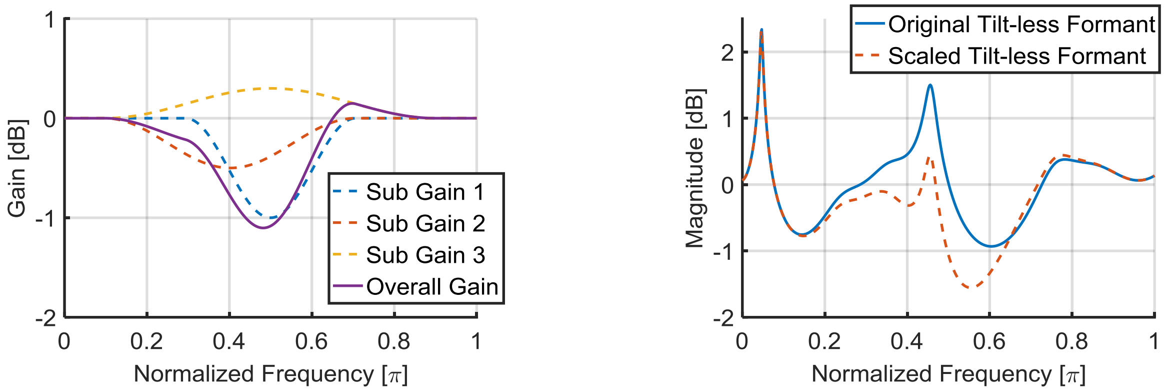
Figure 4-7 Frequency Warping

As an example shown in Figure 4-7, the frequency indices of the tilt-less envelope are multiplied by 1.25 to achieve the horizontal stretching. Then, an interpolation function recovers the modulus of the normalized indexes (the same as the frequency index of the input formant) from the stretched indexes. Considering the efficiency requirement, a 1st-order (linear) interpolation function is used, which can satisfy the majority of the speech applications.

In addition to the linear scaling function (Figure 4-7 (a)), other functions such as piece-wise linear function, logarithm function and exponential function can also be applied to scale the formant envelope. As they are out of the scope of this thesis, the details will not be discussed.

#Magnitude Scaling

The magnitude scaling is meant to adjust (i.e. increase or decrease) the energy of the tilt-less envelope. An example is as shown in Figure 4-8,



(a) Three sub-band gain functions

(b) Magnitude scaled result

Figure 4-8 Magnitude Scaling

This process can be considered as a sub-band equalizer which modifies the loudness on different frequencies. After modification, the formant tilt is added back, and the envelope is converted back to a linear scale.

4.2.2.3 Formant Envelope to LPC Coefficients

The modified LPC coefficients are recovered from the modified formant envelope using a recursive algorithm that is proposed by Tokuda et al. [21], as this algorithm can accurately approximate the frequency response with an acceptable computing burden.

4.3 Application: Voice Gender Conversion

Voice gender conversion is a typical vocal effect that transforms a male’s (female’s) voice as it is spoken as a female’s (male) voice, which can be realized via the proposed system.

According to previous studies, the pitch relationship between male speakers and female speakers is shown in Figure 4-9.

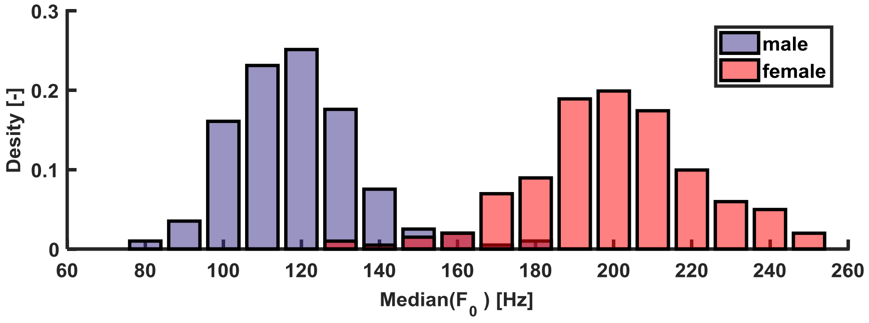


Figure 4-9 Pitch Difference between male speakers and female speakers [37]

The mean value of the pitch frequencies of male and female is around 110Hz and 200Hz. This indicates a general pitch rate of 200Hz/110Hz, which is equivalent to a constant factor of 1.82.

For formant scaling, the LPC coefficients are converted into the formant envelope and modified using a piece-wise linear warping function. The warping function is obtained from a previous statistical analysis [37] which shows the general (phoneme-independent) relationship between the males and females' formants frequencies, as shown in Table 4-I and Figure 4-10.

Table 4-I Formant Relationship between Male and Female

[Hz]	F_1	F_2	F_3	F_4
M	511.2	1411.5	2370.5	3428.0
F	619.0	1686.1	2848.6	4053.3

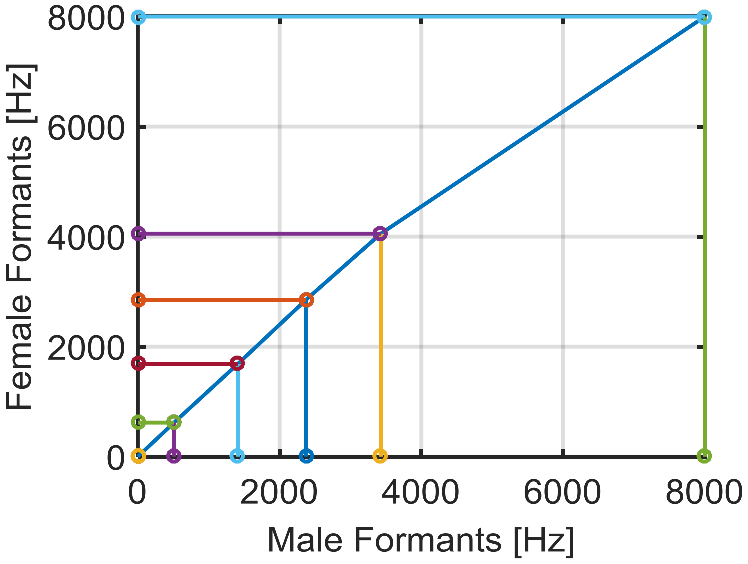


Figure 4-10 Formant Relationship between Male and Female Speakers

According to this table, the piecewise warping function is constructed as shown in equation

(4-2),

$$\Phi(\omega)=\begin{cases}\dfrac{619.0}{511.2}\omega & 0\le\omega\le 511.2\\ \dfrac{1686.1-619.0}{1411.5-511.2}(\omega-511.2)+619.0 & 511.2<\omega\le 1411.5\\ \dfrac{2848.6-1686.1}{2370.5-1411.5}(\omega-1411.5)+1686.1 & 1411.5<\omega\le 2370.5\\ \dfrac{4053.3-2848.6}{3428.0-2370.5}(\omega-2370.5)+2848.6 & 2370.5<\omega\le 3428.0\\ \dfrac{8000-4053.3}{8000-3428.0}(\omega-3428.0)+4053.3 & 3428.0<\omega\le 8000\end{cases} \quad \textbf{(4-2)}$$

4.4 Performance Evaluation on Voice Gender Conversion

This section will show the performance of the proposed system on the voice gender conversion application. Four different formant scaling methods are compared, which include:

- The proposed formant modification (Method A; Section 4.2.2): the obtained LPC polynomial is first converted to the formant envelope; the formant scaling is achieved by warping the formant envelope by the specified scaling factors; the LPC coefficients are then recovered from the warped formant envelope.

- Polynomial-based formant modification (Method B): the roots of the LPC polynomial are decomposed into an argument (phase) and module parameters; the formant scaling is achieved by adjusting the arguments of the roots using the specified scaling factors; the LPC coefficients are then recovered from the modified roots.

- LSF-based formant modification (Method C): the LPC polynomial is first converted to the Linear Spectral Frequency (LSF); the formant scaling is achieved by changing the LSFs using the specified scaling factors; the LPC coefficients are then recovered from the modified LSF.

- Time-domain formant modification (Method D): the LPC polynomial remains unchanged. Instead, both the input signal and the residue signal are frequency scaled to realize the different pitch and formant modifications. This solution lacks the feasibility of nonlinearly scaling the formant, such as piecewise warping the spectrum. In this respect, the formant scaling factor is defined as 1.2 as the linear approximation of the data in Figure 4-10.

Due to the nature that no reference signal is available, the speech signal that is converted by the systems can hardly be evaluated using a pre-defined objective metric. Moreover, as there are more than two instances to be compared, the unbiased ABX test requires cross comparisons between every two instances of the four instances, which largely increases the testing costs. Thus, the MOS subjective test is used here for illustrating the performance of different solutions. In particular, the converted signals (i.e. 8 sentences; each sentence is played twice) are shuffled and played to 5 listeners blindly. The listeners were asked to score the similarity (to the target gender) and quality of each sentence. The experiments will include two parts, the first part is from female to male, and the second part is from male to female.

The MOS results of the male-to-female conversions are shown in Figure 4-11.

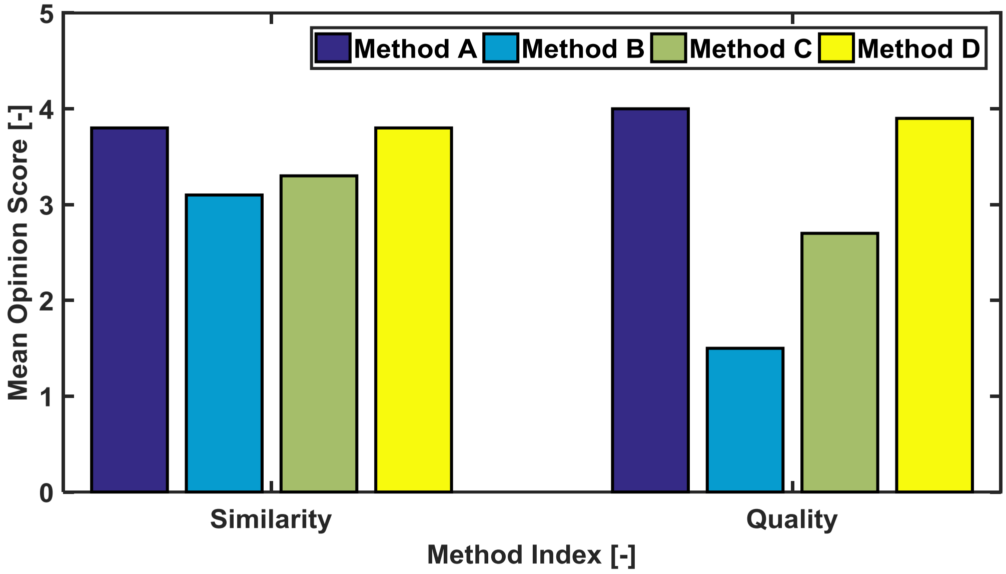


Figure 4-11 Voice Gender Conversion (from Male to Female)

The MOS results of the female-to-male conversions are shown in Figure 4-12

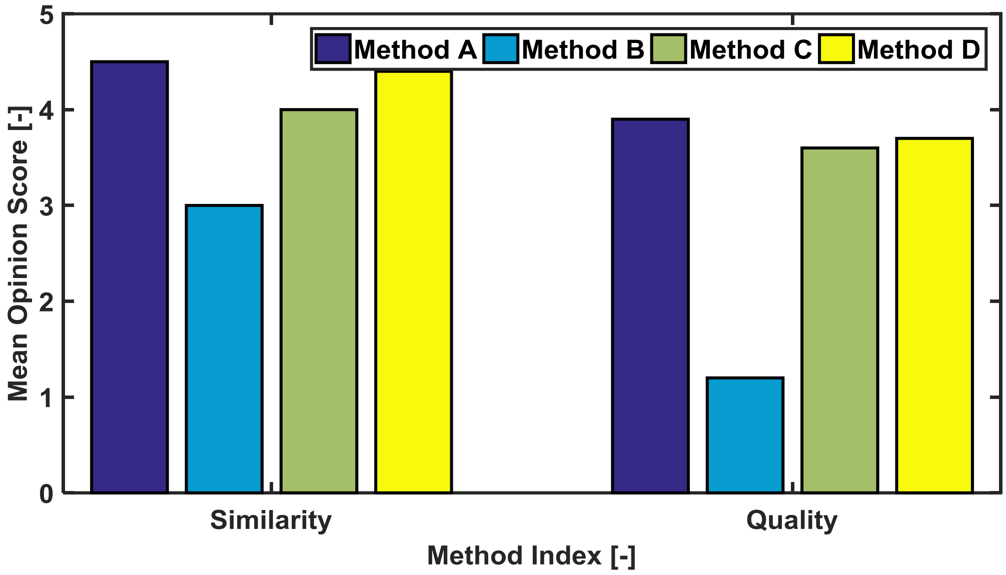


Figure 4-12 Voice Gender Conversion (from Female to Male)

The results show that both the methods A and D achieve the much higher perceptual quality than the method B and the method C. The rationale is that the scaled coefficients (e.g. LPC and LSF) do not always have meaningful interpolations in the frequency domain, where the conversion causes the unpredicted outputs and consequently degenerating the final quality.

Different from the method D that only scale the formant via a linear function, the proposed method A enables a much more flexible warping on the spectral envelope, which means it can be extended for various voice change applications. From another perspective of the phoneme-independent voice gender conversion, the results show that a piecewise frequency warping function does not bring significant benefits.

4.5 Summary and Conclusions

This chapter develops a real-time voice change system based on the proposed frequency scaling algorithm, i.e. PAOLA. By manipulating LPC coefficients in the frequency domain, the proposed real-time voice change system achieves reasonably good performances in terms of speech quality and perceptual identity. In addition, the proposed voice change system does not require buffering long speech segments, which facilitate it to be used in real-time scenarios.

Chapter 5

Literature Review on Voice Conversion

This chapter focuses on analysing the source-to-target spectrum mapping techniques for the voice conversion system, which also presents a discussion in regards to the valuable combinations and variations. In general, the mapping techniques in voice conversion systems aim to capture the non-linear relationships between the spectral characteristic of the source speaker and that of the target speaker. As has been shown in Section 2.3.1 and Section 2.3.2, via a VOCODER analysis (i.e. decompose speech signals to parameters) procedure and a feature extraction procedure, the spectral information is assumed to be converted into formant envelope or the spectral features (e.g. LPC, LSF and MFCC).

For the parallel corpus, where the source and target speakers say the same sentences, a time-alignment step is usually applied to align the phonemes of the source and target speech signals (Section 2.3.3) before training the mapping function. After alignment, the training set is a sequence of paired feature vectors,

$$\mathbf{T} = \left\{ \left(x^{(n)}, y^{(n)} \right) | \, n = 1 \ldots N \right\} \tag{5-1}$$

Here, $x^{(n)}$ and $y^{(n)}$ are *M*-dimension source and target vectors at frame *n*,

$$x^{(n)} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_M \end{bmatrix}, y^{(n)} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{bmatrix} \tag{5-2}$$

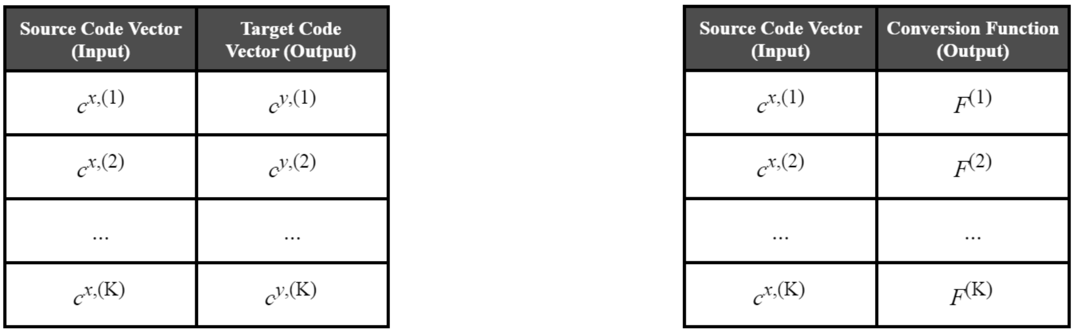
The goal is to find out a suitable mapping function *F*, so that converted feature vectors are similar to the target feature vectors (e.g. with minimum loss *ε* or maximum likelihood *L*). For instance, the Euclidean Distance can be a criterion in which the squared errors are compared.

$$\varepsilon = \sum_{n=1}^{N} d\left(F\left(x^{(n)}\right), y^{(n)}\right)$$
$$d\left(F\left(x^{(n)}\right), y^{(n)}\right) = \sum_{m=1}^{M}\left(\hat{y}_m^{(n)} - y_m^{(n)}\right)^2 \qquad \textbf{(5-3)}$$

Here, $F\left(x^{(n)}\right)$ denotes the converted feature vector at frame n and $\hat{y}_m^{(n)}$ denotes the m^{th} element of the converted feature vector. In addition to the frame-by-frame conversion, the recent approaches also try to model the long-term context information, such as the temporal trajectory [24]. According to the different modelling scheme, the spectral mapping technique can be further classified into six sub-categories, namely, 1) Codebook Mapping, 2) Mixture of Linear Models, 3) Kernel Regression, 4) Frequency Warping, 5) Exemplar-based Mapping and 6) Neural Network Mapping.

5.1 Codebook Mapping

The early works date back to 1988 when vector quantization (VQ) techniques were applied to voice conversion systems to realize a mapping between spectrum features. VQ was developed to compress a vector space into a small number of sparse code vectors, which was valuable in the era when computing capability was extremely limited. In general, the training stage of the VQ-based conversion is to formulate the source-to-target codebook (e.g. vector-to-vector or vector-to-function database) from the corpus, as shown in Figure 5-1.



Source Code Vector (Input)	Target Code Vector (Output)
$c^{x,(1)}$	$c^{y,(1)}$
$c^{x,(2)}$	$c^{y,(2)}$
...	...
$c^{x,(K)}$	$c^{y,(K)}$

(a) Vector-to-vector codebook

Source Code Vector (Input)	Conversion Function (Output)
$c^{x,(1)}$	$F^{(1)}$
$c^{x,(2)}$	$F^{(2)}$
...	...
$c^{x,(K)}$	$F^{(K)}$

(b) Vector-to-function codebook

Figure 5-1 Codebooks

The runtime conversion stage first tries to find a code vector from the source codebook that is closest to the source feature vector, then, the corresponding code vector (e.g. a target feature vector or a source-to-target conversion function) from the target (mapping) codebook is taken to convert or replace the source feature vector.

5.1.1 Hard Vector Quantization

The first VQ-type voice conversion approach [4] creates separated codebooks $\mathbf{C}^x = \begin{bmatrix} c_1^x & \dots & c_K^x \end{bmatrix}$ and $\mathbf{C}^y = \begin{bmatrix} c_1^y & \dots & c_K^y \end{bmatrix}$ to store $k = \{1, 2 \dots K\}$ different code vectors for two speakers, where each codebook is generated independently. During the training stage, each input feature vector $x^{(n)}$ or $y^{(n)}$ is approximated by the nearest code vector in the codebook and uses the index of the code vector as the new representation.

$$\begin{cases} x^{(n)} \xrightarrow{\mathbf{C}^x} u^{x,(n)} \\ y^{(n)} \xrightarrow{\mathbf{C}^y} u^{y,(n)} \end{cases} \qquad \textbf{(5-4)}$$

Here, $u^{x,(n)}$ and $u^{y,(n)}$ denotes the source and target codes of the n^{th} frame.

For examples, the coding procedure (with a codebook $\mathbf{C}^x = \begin{bmatrix} c_1^x & c_2^x & c_3^x \end{bmatrix}$) of the source feature vector $x^{(n)}$ is shown in equation (5-5),

$$x^{(n)} = \begin{bmatrix} 1 \\ 2 \\ 7 \end{bmatrix} \xrightarrow{\begin{cases} c_1^x = [2 \;\; 6 \;\; 4]^{\mathrm{T}} \\ c_2^x = [1 \;\; 2 \;\; 4]^{\mathrm{T}} \\ c_3^x = [5 \;\; 3 \;\; 9]^{\mathrm{T}} \end{cases}} \varepsilon = \begin{bmatrix} 5.099 \\ 3 \\ 4.583 \end{bmatrix} \to u^{x,(n)} = 2 \qquad \textbf{(5-5)}$$

As the nearest code vector of $x(n)$ is c_2^x (based on Euclidean Distance), $x^{(n)}$ is quantized into code $u^{x,(n)} = 2$.

Note that the codes that are generated from the coding procedure are always scalar. This facilitates constructing a 2-dimension histogram $\mathbf{H} = \{h_{p,q}\}$ to record the number of events that the source code happens to be $u^{x,(n)} = p$ and the target code happens to be $u^{y,(n)} = q$, as shown in Figure 5-2.

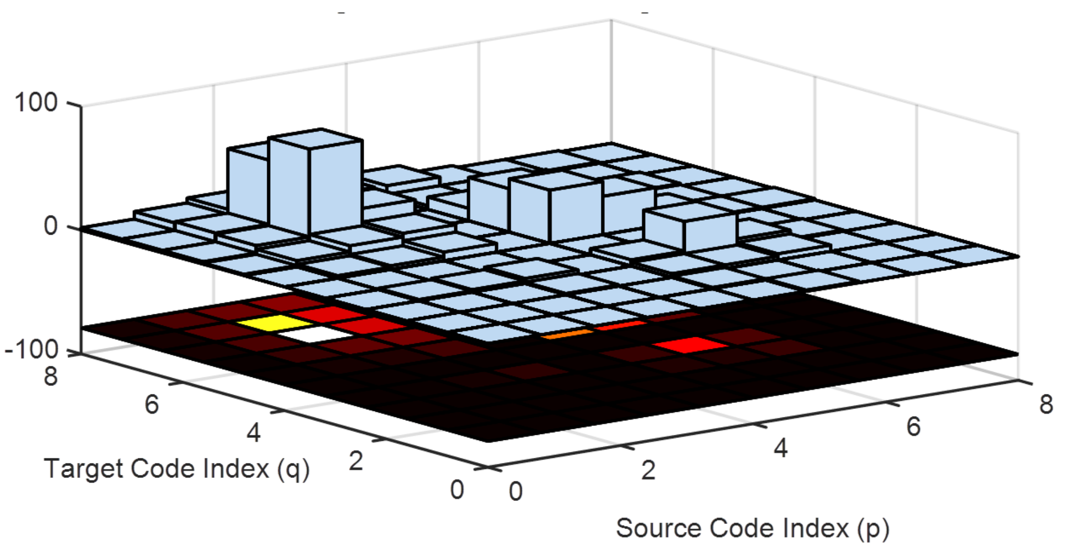


Figure 5-2 Histogram of Source and Target Code Index

Based on the histogram and the target codebook, a mapping codebook can be obtained,

$$c_p^{\hat{y}} = \sum_{q=1}^{K} w_{p,q} c_p^{y}$$
$$w_{p,q} = \frac{h_{p,q}}{\sum_{l=1}^{K} h_{l,q}} \qquad \textbf{(5-6)}$$

Here, $C_p^{\hat{y}}$ denotes the p^{th} mapping codebook and $w_{p,q}$ denotes the normalized weight.

In the runtime conversion stage, the source features vectors are encoded using the source codebook (into index) and decoded using the mapping codebook. The key problem of this approach is that the finite amount of mapping code vectors causes dramatic quantization errors, which consequently degenerates the quality of the output speech.

5.1.2 Fuzzy Vector Quantization

In consideration of the inevitable quantization error caused by hard vector quantization, fuzzy Vector Quantization (FVQ) [38] was proposed to represent source feature vectors as the linear combination of learned code vectors.

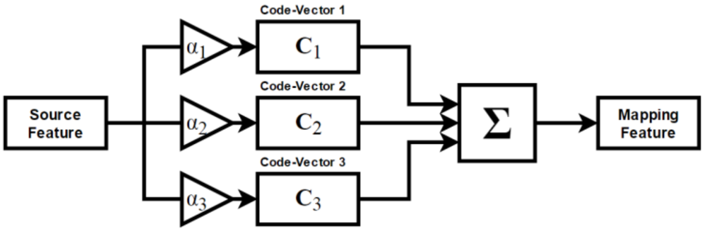


Figure 5-3 Spectrum Conversion using Fuzzy Vector Quantization

Note that the FVQ's training step (to calculate the mapping code vectors) is identical to that of hard vector quantization. Differently, during the conversion step, the source feature vector is first converted into different codes labelled with independent weights.

For example, one source feature vector is given as $x^{(n)}$ and three code vectors are $\mathbf{C}^x = \left[c_1^x \quad c_2^x \quad c_3^x \right]$, the feature vector is encoded as shown below

$$x^{(n)} = \begin{bmatrix} 1 \\ 2 \\ 7 \end{bmatrix} \xrightarrow{\begin{cases} c_1^x = [2 \quad 6 \quad 4]^{\mathrm{T}} \\ c_2^x = [1 \quad 2 \quad 4]^{\mathrm{T}} \\ c_3^x = [5 \quad 3 \quad 9]^{\mathrm{T}} \end{cases}} \varepsilon = \begin{bmatrix} 5.099 \\ 3 \\ 4.583 \end{bmatrix} \rightarrow \begin{cases} \alpha_1^{x,(n)} = 0.2623 \\ \alpha_2^{x,(n)} = 0.4459 \\ \alpha_3^{x,(n)} = 0.2918 \end{cases} \quad \textbf{(5-7)}$$

Herer $\alpha_k^{x,(n)}$ is the weight that reflects the similarity between the input source vector and each source code vector. Instead of searching for the index of the nearest code vector and replacing the source vector directly with a single mapping vector, based on the weight, FVQ achieves spectral mapping by weighted combining of the all mapping code vectors $\mathbf{C}^{\hat{y}}$.

$$y^{(n)} = \sum_{k=1}^{K} \alpha_k^{x,(n)} c_k^{\hat{y}} \quad \textbf{(5-8)}$$

The weight function can be calculated using different methods, including Euclidian distance, phonetic information, exponential decay and vector field smoothing. The hard VQ can be considered as a special case of the Fuzzy VQ, in which only the weight of the most similar code vector is set to **ONE**, while others are set to **ZERO**.

5.1.3 Difference Vector Quantization

Although FVQ reduces the quantization errors by weighted combining codes, it still suffers from muffle and unnatural quality issues [39]. The rationale is that the mapping output is the combination of a finite number of code vectors. If the new input source feature vector contains new acoustic information that the codebooks do not contain, this output is unpredictable.

To allow the system to preserve reasonable acoustic information during the runtime conversion stage, a difference vector is introduced to calculate the difference between the source feature vectors and the target feature vectors. The vector quantization (e.g. codebook) is trained based on the joint feature vectors $z^{(n)} = \left[x^{(n)} \quad y^{(n)} \right]^{\mathrm{T}}$. After the source and target code vectors of the k^{th} cluster (i.e. $\left[\mathbf{C}_k^x \quad \mathbf{C}_k^y \right]$) is obtained, the corresponding difference vector Δc_k is calculated as below,

$$\Delta c_k = c_k^x - c_k^y \tag{5-9}$$

During the runtime conversion stage, the new source features $x^{(n)}$ are first encoded into the code *k* (according to the procedure in hard vector quantization), and the corresponding k^{th} difference vector Δc_k is found. The final mapping output is calculated by adding the difference vector Δc_k to the source feature vector.

$$y^{(n)} = x^{(n)} + \Delta c_k = x^{(n)} - c_k^x + c_k^y \tag{5-10}$$

The key principle of the difference vector Δc_k is to remove the averaged acoustic information of the source speaker c_k^x and add the average acoustic information of the target speaker c_k^y.

#Remark

It is worth to note that the Different Vector Quantization approach can be further extended to the fuzzy version, which weighted combining the difference vectors $\Delta c_{k=1,2\ldots,K}$.

$$y^{(n)} = x^{(n)} + \sum_{k=1}^{K} \alpha_k^{(n)} \Delta c_k = x^{(n)} + \sum_{k=1}^{K} \alpha_k^{(n)} \left(c_k^y - c_k^x \right) \tag{5-11}$$

Here, $\alpha_k^{(n)}$ is the fuzzy weights obtained according to the equation (5-7).

5.1.4 Speaker Transformation Algorithm using Segmental Codebooks (STASC)

Speaker Transformation Algorithm using Segmental Codebooks (STASC) proposed by Arslan et al. is a typical and compressive voice conversion system during the early stage. It generates codebook based on *K* different HMM states. After sequential labelling, the joint (source and target) LSF features $z^{(n)} = \left[x^{(n)} \quad y^{(n)} \right]$ that belong to the state *k* are averaged into the joint code vectors $\mathbf{C}_k^z = \left[\mathbf{C}_k^x \quad \mathbf{C}_k^y \right]$. For each new frame of source signal *x*, it is approximated by a set of source code vectors $\left\{ \mathbf{C}_k^x \mid k = 1 \ldots K \right\}$ with the corresponding weights α_k, where the parameter α_k is calculated by minimizing the error between the optimal approximation and the real source feature $x^{(n)}$ via a gradient descent algorithm. Then, the mapping output $\tilde{y}^{(n)}$ is estimated as shown below,

$$\tilde{y}^{(n)} = \sum_{k=1}^{K} \alpha_k \times \left(h_k \left(x^{(n)} \right) \right) \qquad \textbf{(5-12)}$$

Here, h_k denotes the k^{th} conversion function to manipulate the spectrum. The equation indicates that the mapping output $\tilde{y}^{(n)}$ is the weighted combination of the manipulated source vectors $h_k \left(x^{(n)} \right)$ converted by different transfer functions.

As STASC relies on the source-filter theory of the speech production mechanism and the reversibility/multiplicativity of the transfer function in the frequency domain, the transfer function h^k for conversion is shown below,

$$h_k \left(z^{-1} \right) = \frac{g_k^x \left(z^{-1} \right)}{g_k^y \left(z^{-1} \right)} \qquad \textbf{(5-13)}$$

Here, $g_k^x \left(z^{-1} \right)$ and $g_k^y \left(z^{-1} \right)$ denotes the k^{th} LPC filters (converted from the LSF code vectors $\mathbf{C}_k^x$ and $\mathbf{C}_k^y$) of the source speaker and the target speaker. The flow chart of the STASC system is given in Figure 5-4. The example contains three code vectors (each one corresponds to a transfer function).

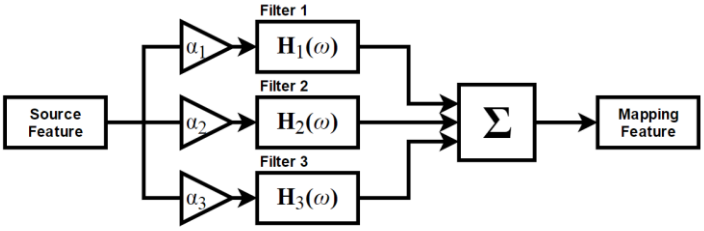


Figure 5-4 Spectrum Conversion using STASC System

Similar to Fuzzy Difference Vector VQ, STASC is a weighted combination approach. Differently, STASC converts the source feature using different *Z*-domain transfer functions $h_k\left(z^{-1}\right)$ but does not compensate the source feature vector by the difference vectors.

5.2 Mixture of Linear Models

Mixture of linear models is a further step that extends the VQ-based spectrum mapping to multi-variable linear regression. Similar to VQ, the mixture of linear models use the clustering techniques (usually soft clustering model, e.g. Gaussian Mixture Model) to divide the scattered samples into clusters (e.g. probability density function in GMM). Then, a set of linear regression functions are trained to minimize the total errors between the converted feature vectors and the target feature vectors.

5.2.1 Source-Only Gaussian Mixture Model

The first Gaussian Mixture Model (GMM) based voice conversion was proposed by Stylianou et al. [40]. This approach realizes a continuous probabilistic transformation of spectrum features, which was shown to be reliable and valid by the existing studies. The methodology of this approach is somewhat different from the approaches that are based on VQ, where it estimates the posterior probability of each source vector that belongs to the k^{th} Gaussian probability density function, and estimates a joint regression function for all clusters, but not for each cluster.

As this approach solely models the distribution of source features, it is also named as Source-Only GMM (SO-GMM). In particular, SO-GMM assumes the distribution of the source feature vector $x^{(n)}$ can be approximated by summing *K* different weighted Gaussian distributions, which are known and used as components in the following content.

$$\begin{cases} P\left(x^{(n)} \mid \lambda\right) = \sum_{k=1}^{K} \phi_k N\left(x^{(n)}; \boldsymbol{\mu}_k^x, \boldsymbol{\Sigma}_k^x\right) \\ \sum_{k=1}^{K} \phi_k = 1 \end{cases} \quad \textbf{(5-14)}$$

Here, *N* denotes Gaussian distribution, *k* denotes the index of a component, ϕ_k denotes the weight of the *k*th component, $\boldsymbol{\Sigma}_k^x$ and $\boldsymbol{\mu}_k^x$ denotes the covariance matrix and mean vector of the *k*th component, and *λ* is the parameter settings (i.e. $\boldsymbol{\Sigma}_k^x$, $\boldsymbol{\mu}_k^x$ and ϕ_k) of GMM. Three types of covariance matrixes for regression are considered in the original approach [40], namely 1) full, 2) diagonal and 3) zero. Theoretically, a full covariance is necessary to model the feature that maintains strong correlations (e.g. spectrum, LSF, LPC) between coefficients, while a diagonal covariance can be used if the correlations between coefficients are removed (e.g. MFCC, MCC). The zero covariance is rarely used as it degenerates GMM to the VQ-Type clustering.

The SO-GMM-based conversion aims to find a function *F* that transforms the source feature vector $x^{(n)}$ into the target feature vector $y^{(n)}$. The parametric form as shown in equation (5-16) is assumed to achieve minimum mean square error [41].

$$P\left(k \mid x^{(n)}; \lambda\right) = \frac{\phi_k N\left(x^{(n)}; \boldsymbol{\mu}_k^x, \boldsymbol{\Sigma}_k^x\right)}{\sum_{l=1}^{L} \phi_l N\left(x^{(n)}; \boldsymbol{\mu}_l^x, \boldsymbol{\Sigma}_l^x\right)} \quad \textbf{(5-15)}$$

$$F\left(x^{(n)} \mid \lambda\right) = \sum_{l=1}^{L} P\left(k \mid x^{(n)}; \lambda\right)\left[\boldsymbol{\mu}_l^x + \boldsymbol{\Gamma}_l^{x,y}\left(\boldsymbol{\Sigma}_l^x\right)^{-1}\left(x^{(n)} - \boldsymbol{\mu}_l^y\right)\right] \quad \textbf{(5-16)}$$

As SO-GMM models the distribution of source features, $\boldsymbol{\Sigma}_k^x$, $\boldsymbol{\mu}_k^x$ and ϕ_l are already known from the GMM training. However, the cross-covariance $\boldsymbol{\Gamma}_l^{x,y}$ (between the source and target vectors) and the target mean $\mathbf{v}_l$ are under-determined because the distribution of the target feature vectors is not modelled, which is estimated by minimizing the squared error *ε*, as equation (5-17), via a least squares optimization.

$$\varepsilon = \sum_{n=1}^{N} \left\| y^{(n)} - F\left(x^{(n)}\right) \right\| \quad \textbf{(5-17)}$$

Although SO-GMM initially enables the continuous probability transformation for voice conversion, it brought a well-known problem – "over-smoothing" [24]. As the distribution of the target features is ignored, the system cannot determine an accurate function, if the source features are very similar but the target features are totally different. For example, two words "hello" and "steady" have the same phoneme /ə/. The source speaker might always pronounce the phonemes /ə/ of the two words with very similar timbre while the target speaker pronounces /ə/ with totally different timbres, namely one-to-many conversion.

Particularly, if the source features follow a Gaussian distribution and the corresponding target features follow two different Gaussian distributions as shown in Figure 5-5, the least square optimization will face a severe problem.

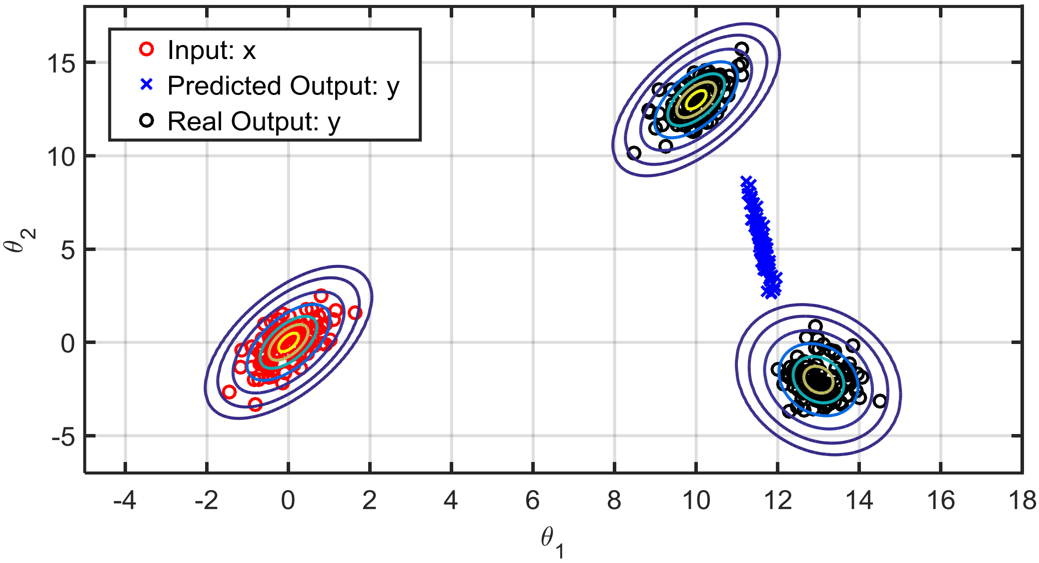


Figure 5-5 One Gaussian samples to Two Gaussian samples Regression

As the optimization is executed to minimize the mean square error, the predicted outputs (i.e. blue crosses) will not fall on the area of the real outputs (i.e. black circles) with high probability. As the physical interpretation of the fallen area may be illogical for the speech generation, the quality of the converted speech signal becomes poor.

5.2.2 Joint-Density Gaussian Mixture Model

In order to tackle the problem shown in Figure 5-5, an alternative solution is to model the joint probability density of both the source and target features, which is named as

Joint-Density GMM (JD-GMM) [42]. JD-GMM concatenates the paired source and target features into the joint vectors $z^{(n)} = \left[x^{(n)} \quad y^{(n)} \right]$ for clustering, where $x^{(n)}$ and $y^{(n)}$ are spectrum features of the source speaker and the target speaker. As the joint vector $z^{(n)}$ contains both the source feature and the target feature, two joint vectors with similar source features but different target features can be easily distinguished by the target features as shown below,

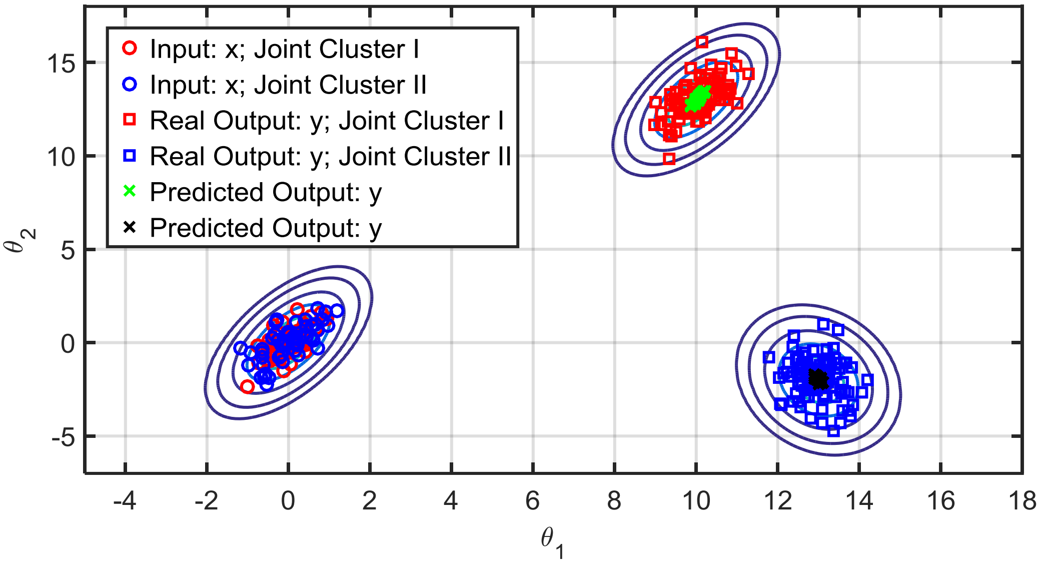


Figure 5-6 Two Gaussian samples to Two Gaussian samples Regression

In order to achieve this, the probability density function of the joint vector $z^{(n)}$ is modelled by a GMM,

$$\begin{aligned} P\left(z^{(n)}\right) &= \sum_{k=1}^{K} P\left(k \mid z^{(n)}; \lambda_z\right) N\left(z^{(n)}; \boldsymbol{\mu}_k^z, \boldsymbol{\Sigma}_k^z\right) \\ &= \sum_{k=1}^{K} w_k N\left(z^{(n)}; \boldsymbol{\mu}_k^z, \boldsymbol{\Sigma}_k^z\right) \end{aligned} \qquad \textbf{(5-18)}$$

Similar to the SO-GMM, JD-GMM assumes the distribution of the joint vectors $z^{(n)}$ is the weighted sum of M different Gaussian distributions. N denotes Gaussian distributions, k denotes the index of a component, w_k denotes the weight of the k^{th} component, and λ is the parameter settings (i.e. $\boldsymbol{\Sigma}_k^z$, $\boldsymbol{\mu}_k^z$ and w_k) of GMM.

Note that the joint covariance matrix $\boldsymbol{\Sigma}_k^z$ contains 4 sub covariance matrixes,

$$\mathbf{\Sigma}_k^z = \begin{bmatrix} \mathbf{\Sigma}_k^{xx} & \mathbf{\Sigma}_k^{xy} \\ \mathbf{\Sigma}_k^{yx} & \mathbf{\Sigma}_k^{yy} \end{bmatrix} \tag{5-19}$$

Here, $\mathbf{\Sigma}_k^{xx}$ denotes the auto-covariance of the source feature vectors, $\mathbf{\Sigma}_k^{yy}$ denotes the auto-covariance of the target feature vectors, $\mathbf{\Sigma}_k^{xy}$ and $\mathbf{\Sigma}_k^{yx}$ denote the cross-covariance of the source and target feature vectors. Similar to SO-GMM, the four matrixes are all set to diagonal if the correlations between coefficients are low (e.g. MCC is used), otherwise, the four covariance matrixes are set to full. In addition, the joint mean vector $\boldsymbol{\mu}_k^z$ consists of 2 subvectors,

$$\boldsymbol{\mu}_k^z = \begin{bmatrix} \boldsymbol{\mu}_k^x \\ \boldsymbol{\mu}_k^y \end{bmatrix} \tag{5-20}$$

Here, $\boldsymbol{\mu}_k^x$ and $\boldsymbol{\mu}_k^y$ denote the mean of the k^{th} component of the source and target features. Similar to SO-GMM, JD-GMM aims to find an optimal function F that can transform the source feature vector $x^{(n)}$ into its counterpart in the target data set $y^{(n)}$ with minimum error. The following form is used to achieve the optimal conversion:

$$P\left(q \mid x^{(n)}; \lambda_z\right) = \frac{w_q N\left(x^{(n)}; \boldsymbol{\mu}_q^x, \mathbf{\Sigma}_q^{xx}\right)}{\sum_{k=1}^{K} w_k N\left(x^{(n)}; \boldsymbol{\mu}_k^x, \mathbf{\Sigma}_k^{xx}\right)} \tag{5-21}$$

$$F\left(x^{(n)}\right) = \sum_{k=1}^{K} P\left(k \mid x^{(n)}; \lambda_z\right)\left[\boldsymbol{\mu}_k^y + \mathbf{\Sigma}_k^{xy}\left(\mathbf{\Sigma}_k^{xx}\right)^{-1}\left(x^{(n)} - \boldsymbol{\mu}_k^x\right)\right] \tag{5-22}$$

As the Kain's approach models the distribution of joint features, $\mathbf{\Sigma}_k^{xx}$, $\mathbf{\Sigma}_k^{xy}$, $\boldsymbol{\mu}_k^x$, $\boldsymbol{\mu}_k^y$ and w_k are already the known parameters from the GMM clustering procedure, which largely reduces the computation costs compared to SO-GMM. During the runtime stage, the posterior probability $P\left(k \mid x^{(n)}; \lambda_z\right)$ is calculated, and the input source feature vector is converted using the equation (5-22).

#Remark

It is worthy to note that only the source feature vectors are available in the runtime stage. The frame–level classifications (e.g. using JD-GMM) cannot guarantee the adjacent frames that belong to the same phoneme to be categorized as the same class because the target feature vector is unknown. This discontinuous classification will cause applying different regression functions on adjacent frames, which consequently degenerates the converted speech quality.

5.3 Kernel Regression

Kernel regression is an alternative approach to the mixture of linear models, which firstly maps the source feature vectors from its original space to another kernel space via a non-linear function. Then, the linear regression is applied to the new kernel space. The basic assumption of kernel regression is that, after nonlinear mapping, the vectors in the kernel space can be easily converted into the target feature vectors via a simple (i.e. linear) function.

5.3.1 Dynamic Kernel Partial Least Square

In theory, the nonlinear mapping process that uses the radial basis function (RBF) kernel (e.g. Gaussian kernel) enables fitting an arbitrary non-linear function, which meets the requirement of spectrum conversion. In order to capture the nonlinear relationships and the temporal information of the source features and the target features, Dynamic Kernel Partial Least Square (DKPLS) was introduced by Helander et al. [16]. The training procedure of DKPLS aims to 1) search for the suitable parameters of the kernel functions, 2) estimate the optimal regression model between the kernel vectors (i.e. converted from the source feature vectors) and the target vectors, and 3) model the temporal information. In particular, DKPLS uses Gaussian kernels (equation (5-23)),

$$\kappa\left(x^{(n)},k\right)=e^{\frac{-\left\|x^{(n)}-\boldsymbol{\mu}_k\right\|^2}{2\sigma^2}} \qquad \textbf{(5-23)}$$

Here, the centroid $\boldsymbol{\mu}_k$ (i.e. mean) of the k^{th} Gaussian kernel is estimated via the k-means clustering, and the variance parameters σ is set to $\sqrt{0.5}$ according to empirical studies.

Via the obtained kernel functions, the source feature vectors are converted to the kernel vectors $\boldsymbol{\Lambda}=\left[\kappa^{(1)} \quad \kappa^{(2)} \quad \dots \quad \kappa^{(N)}\right]^{\mathrm{T}}$.

$$\begin{aligned}\mathbf{\Lambda} &= \begin{bmatrix} \kappa\left(x^{(1)},k=1\right) & \kappa\left(x^{(2)},k=1\right) & \cdots & \kappa\left(x^{(N)},k=1\right) \\ \kappa\left(x^{(1)},k=2\right) & \kappa\left(x^{(2)},k=2\right) & \cdots & \kappa\left(x^{(N)},k=2\right) \\ \vdots & \vdots & \ddots & \vdots \\ \kappa\left(x^{(1)},k=K\right) & \kappa\left(x^{(2)},k=K\right) & \cdots & \kappa\left(x^{(N)},k=K\right) \end{bmatrix}^{\mathrm{T}} \\ &= \begin{bmatrix} \kappa^{(1)} & \kappa^{(2)} & \cdots & \kappa^{(N)} \end{bmatrix}^{\mathrm{T}} \end{aligned} \qquad \textbf{(5-24)}$$

In order to capture the temporal information, the dynamic kernel vector is constructed as equation (5-25).

$$d^{(n)} = \begin{bmatrix} p^{(n-1)} \\ p^{(n)} \\ p^{(n+1)} \end{bmatrix} \qquad \textbf{(5-25)}$$

Based on the dynamic kernel vectors, a regression function *F* can be found by minimizing the sum of squared conversion errors, equation (5-26),

$$\varepsilon = \sum_{n=1}^{N} \left\| g\left(y^{(n)}\right) - F\left(d^{(n)}\right) \right\|^2 \qquad \textbf{(5-26)}$$

Here, $y^{(n)}$ denotes the *n^th^* target feature vector, *F* denotes the linear regression function, and *g* denotes the dimension reduction function. Due to the nature that the dynamic kernel vectors can be collinear (i.e. the kernel vectors of the adjacent frames are identical, and the normal matrix may be not invertible), the parameters of a regression function *F* is determined by a partial least square method [16].

5.3.2 Mixed Kernel Support Vector Regression

As Support Vector Regression (SVR) is a feasible method to model the nonlinear relationship between source and target speakers effectively and be less prone to over-fitting, it was imposed to spectrum mapping by Song et al. [43], namely Mixed Kernel Support Vector Regression (MKSVR). Another remarkable novelty of Song’s approach is that it combines two kernel functions for nonlinear mapping, such that the system achieved better generalization. In particular, the RBF kernel is used for better interpolations, and the polynomial kernel is used for reasonable extrapolations. The results of two kernels are mixed via a weighted combination. The results demonstrate that this method outperforms the original JD-GMM approach, as it achieves a high similarity between converted and target speakers, without sacrificing quality and naturalness.

5.4 Frequency Warping

Both the mixture of the linear model and the kernel regression can support the nonlinear spectrum mapping theoretically. However, the outputs may be "unbounded" and lack generalizability. For instance, if the runtime source features fall outside the area that only covers the training source features, the converted features may not have good physical interpretations from the perspective of speech signal which potentially degenerates the quality of the converted speech signal. Moreover, the statistical over-smoothing [24] which removes the perceptual details from the converted spectrum may also cause the converted speech signals to be muffled and unnatural.

These problems led to the proposal of a frequency warping based approaches. Different from the regression-based algorithm, the frequency warping is physically motivated, which guarantees a "bounded" conversion and prevents removing the perceptual details. The basic assumption of frequency warping is that the formant frequencies and bandwidths are different for different speakers. Therefore, the conversion procedure should be constrained to manipulate the formant frequencies/bandwidths, which prevents the unreasonable output. In addition to the frequency warping, the amplitude scaling is another necessary component for compensating the spectral energy, which enhances the perceptual individuality.

5.4.1 Piecewise-Linear Warping Function

As the complex roots of the LPC coefficients reflect the formant peaks in the frequency domain, an intuitive solution is to warp the spectral envelope such that the formant peaks are aligned. The proposed method aims to find the optimal peak-to-peak alignment as shown in Figure 5-7.

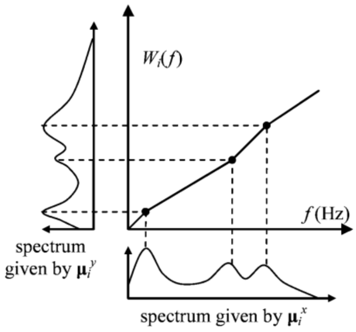

$W_i(f)$
f(Hz)
spectrum given by $\mathbf{\mu}_i^y$
spectrum given by $\mathbf{\mu}_i^x$

Figure 5-7 Piece-wise Linear Warping Function

To this end, the frequencies of the formant peaks (from source and target) are paired in an optimal way to determine the desired piecewise-linear warping function, so that the spectrum distortion is minimized after warping [5]. A noticeable problem of this method is the intensive computation cost, as pairing the suitable formant pairs requires enumerating all combinations. For instance, if the number of poles located at the positive semi-spectrum is $c = 11$ and 5 of them ($b = 5$) are assumed to be the effective formant peaks which are used for pairing, the complexity is derived in equation (5-27),

$$\mathbf{O}\left(\frac{c!}{b!(c-b)!}\right)_{c=11,b=5} = \left({}^{11}\mathrm{C}_5\right)^2 = \left(\frac{11!}{(5!)((11-5)!)}\right)^2 = (462)^2 = 213444 \tag{5-27}$$

Here, ${}^{11}\mathrm{C}_5$ denotes randomly selecting 5 formants from 11 candidates for pairing, and $\left({}^{11}\mathrm{C}_5\right)^2$ denotes the cross-comparison of ${}^{11}\mathrm{C}_5$ sets of formants from source to target, which results totally 213444 times comparison.

After the effective formant peaks (i.e. for pairing) are extracted, the piece-wise linear warping function is obtained based on them, as shown in equation (5-28).

$$W(\omega) = \begin{cases} \alpha_0\omega + \beta_0 & \omega_0 < \omega < \omega_1 \\ \alpha_1\omega + \beta_1 & \omega_2 < \omega < \omega_3 \\ \vdots & \vdots \\ \alpha_Q\omega + \beta_Q & \omega_Q < \omega < \omega_{Q+1} \end{cases} \tag{5-28}$$

Here, the parameters α_q and β_q are defined as in (5-29),

$$\alpha_q = \frac{\varpi_{q+1} - \varpi_q}{w_{q+1} - w_q}, \beta_q = \varpi_q - \alpha_i w_q \\ w_0 = \varpi_0 = 0, w_{Q+1} = \varpi_{Q+1} = \pi \qquad \textbf{(5-29)}$$

Here, w_q denotes the q^{th} formant frequency of the source speech and ϖ_q denotes the q^{th} formant frequency of the target speech.

During the runtime stage, as the pairing functionality is only calculated during the training stage, this solution becomes efficient, which does not require a high computation capability.

5.4.2 Dynamic Warping Function

Instead of calculating the warping function by paring the effective formant peaks, another feasible solution is to directly align the formant envelope. A famous non-parametric method to align the formant envelope is dynamic warping which calculates the sample-by-sample warping function to minimize the difference between the target and the converted envelope.

Note that the dynamic warping function can be not only applied to the original spectral envelope but can be also applied to the formant density function. Compared to the spectral envelope, the formant density function is not affected by formant energy or spectral tilt, which is more robust for formant comparison. Specifically, the formant density function can be obtained by histogram and Parzen window, where the Parzen window provides a finer and smoother estimation of the formant density than the histogram.

The optimization of dynamic warping function utilizes the powerful searching tools such as greedy search algorithms and breadth-first searching algorithms [22]. Usually, if the spectral envelope is clean and smooth (e.g. obtained by the high-quality VOCODER), the estimation of the greedy algorithm and the BFS algorithm are very similar. However, if the spectral envelope is noising and it contains small incorrect formant peaks, the BFS is a more accurate solution compared to the greedy algorithm.

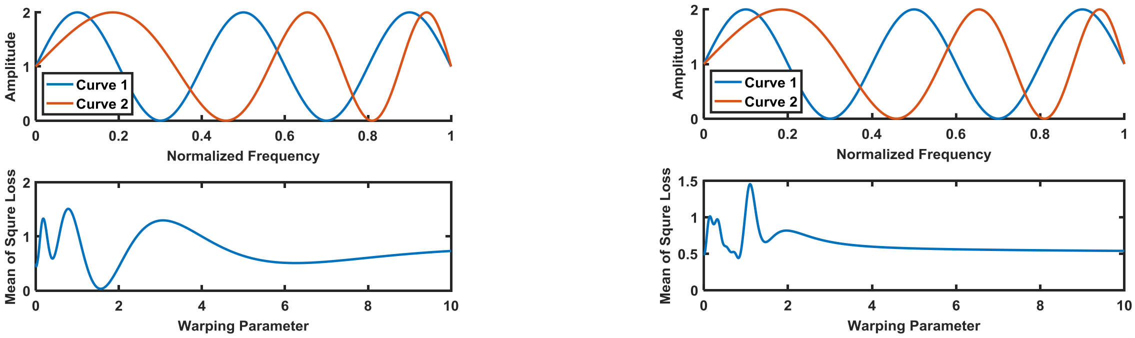
The computation complexity of greedy search is linear with time, and the computation complexity of the BFS algorithm is quadratic with time. However, as the BFS is only required in the training stage, the efficiency of the runtime conversion stage is relatively small and acceptable, as it is not related to BFS.

#Remark

The piece-wise linear warping function can be conveniently integrated with the dynamic warping. As the spectral envelope (e.g. formant bandwidth) between two adjacent formant peaks can be more accurately converted using a nonlinear function.

5.4.3 Other Approaches

Most of the warping techniques are developed according to the principle of Vocal Tract Length Normalization (VTLN) which is used in Automatic Speech Recognition (ASR). Particularly, Sunderman et al. [44] studied various vocal tract length normalization approaches for voice conversion, which includes piecewise, power, quadratic and bilinear VTLN functions. Although some warping functions only need one parameter (such as symmetric piecewise linear function, quadratic function), they can hardly achieve the global optima as the loss function in regard to the warping parameter is not convex as plotted in Figure 5-8.



(a) Quadratic Function

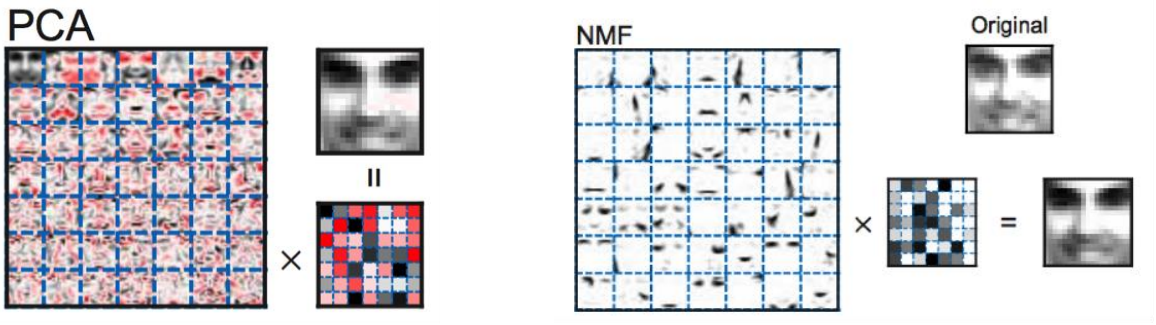
(b) Linear Function

Figure 5-8 Quadratic and Linear Warping Functions and the Loss Functions

A possible approximation of these warping functions can be realized by estimating the optimal warping curve using dynamic warping method and then applying linear/non-linear regression analysis to obtain the warping parameter. However, it becomes superfluous as the warping curve obtained from dynamic warping is usually more accurate. Another feasible method is to apply the global searching/optimization algorithms (e.g. branch and bound [45] or a meta-heuristic algorithm) to find the suitable parameters, which is a big fish in a small pond. In addition, a parametric bilinear warping method is proposed by Erro et al. [46], and an iterative algorithm is used to estimate the optimal parameter in the cepstral domain.

5.5 Exemplar-based Mapping

Exemplar-based mapping is a further step of VQ-based mapping towards a different research direction, which approximates the spectral features using a set of bases. Different from VQ that the spectral feature is approximated by a single quantized (feature) vector, an exemplar method uses multiple vectors to approximate the input features. Using an image processing application (i.e. facial feature extraction) as an example (Figure 5-9),



(a) Principle Component Analysis

(b) Non-negative Matrix Factorization

Figure 5-9 Linear Matrix Factorization (Exemplar) Examples

The exemplar methods decompose the signal into components (i.e. bases) and weights. The small components can be used to recover the original signal by weighted adding together.

Particularly, the general formulization of the **linear** exemplar method with minimizing the square loss is shown in equation (5-30),

$$\hat{\mathbf{x}}_n = \sum_{k=1}^{K} w_{k,n}\mathbf{b}_k = \mathbf{w}_n^{\mathrm{T}}\mathbf{B}$$
$$\underset{\mathbf{w},\mathbf{B}}{\arg\min} \sum_{n=1}^{N} \left\| \mathbf{x}_n - \hat{\mathbf{x}}_n \right\|_2^2 \qquad \textbf{(5-30)}$$

Here, $w_{k,n}$ is the n^{th} sample's weight for the k^{th} basis $\mathbf{b}_k$. The number of bases can be either smaller or larger than the dimension of the features. Specifically, if the dimension of the features is smaller, the bases are the overcomplete representations of the original feature. As there are infinite possible ways to construct the overcomplete representations without limitation, the L1 regularization is usually added to the objective function (equation (5-31)) to calculate the unique sparse weights.

$$\begin{aligned}
&\hat{\mathbf{x}}_n = \sum\nolimits_{k=1}^{K} w_{k,n}\mathbf{b}_k = \mathbf{w}_n^{\mathrm{T}}\mathbf{B} \\
&\underset{\mathbf{w},\mathbf{B}}{\arg\min} \sum_{n=1}^{N} \left(\left\| \mathbf{x}_n - \hat{\mathbf{x}}_n \right\|_2^2 + \lambda \left\| \mathbf{w}_n \right\|_1 \right)
\end{aligned} \qquad \textbf{(5-31)}$$

In addition to the regularization terms, the feasible region of the weights and bases can also be limited. As the module of the spectral envelope in the frequency domain is always positive, a reasonable assumption is that both the weights and the bases must be non-negative, which formulates the Non-negative Matrix Factorization (NMF), as shown in equation (5-32).

$$\begin{aligned}
&\hat{\mathbf{x}}_n = \sum\nolimits_{k=1}^{K} w_{k,n}\mathbf{b}_k = \mathbf{w}_n^{\mathrm{T}}\mathbf{B} \\
&\underset{\mathbf{w},\mathbf{B}}{\arg\min} \sum_{n=1}^{N} \left(\left\| \mathbf{x}_n - \hat{\mathbf{x}}_n \right\|_2^2 \right) \\
&s.t. \quad w_{k,n} \geq 0 \\
&s.t. \quad b_{k,m} \geq 0
\end{aligned} \qquad \textbf{(5-32)}$$

5.5.1 Conventional Exemplar-based Mapping using NMF for Spectrum conversion

Under the context of spectrum mapping, a feasible approach is to concatenate the spectral features of the source and target speakers into the joint vectors $z^{(i)} = \left[x^{(i)}, y^{(i)} \right]$. Then, the NMF can be applied to the joint vectors to obtain a set of joint bases $\mathbf{B}^z = \left[\mathbf{B}^x, \mathbf{B}^y \right]$ and a set of shared weights **w** (equation (5-33))

$$\begin{aligned}
&\hat{\mathbf{z}}_n = \begin{bmatrix} \hat{\mathbf{x}}_n \\ \hat{\mathbf{y}}_n \end{bmatrix} = \sum\nolimits_{k=1}^{K} w_{k,n} \begin{bmatrix} \mathbf{b}_k^x \\ \mathbf{b}_k^y \end{bmatrix} = \mathbf{w}_n^{\mathrm{T}} \begin{bmatrix} \mathbf{B}^x \\ \mathbf{B}^y \end{bmatrix} \\
&\underset{\mathbf{w},\mathbf{B}}{\arg\min} \sum_{n=1}^{N} \left(\left\| \hat{\mathbf{z}}_n - \mathbf{z}_n \right\|_2^2 \right) \\
&s.t. \quad w_{k,n} \geq 0 \\
&s.t. \quad b_{k,m}^x, b_{k,m}^y \geq 0
\end{aligned} \qquad \textbf{(5-33)}$$

In other words, the joint vectors are the weighted sum (by **w**) of the joint bases, which is equivalent to the source (target) features are the weighted sum of the source (target) bases.

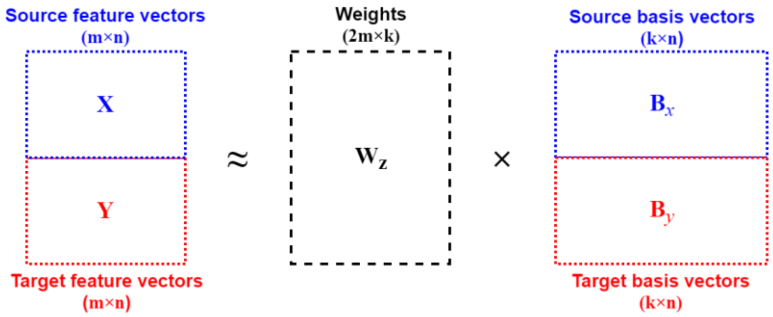


Figure 5-10 Joint Feature Vectors, Weights and Joint Bases in NMF

Following the assumption that a spectral envelope is the result of the resonance from the vocal tract, it is intuitive to consider that a spectrum is a weighted combination of the non-negative formant peaks.

Training:

As the objective function of the conventional NMF is non-convex [47] to **w** and **B**, various methods were introduced to calculate the weights and bases. A popular algorithm – Multiplicative Update Rule [48] can be applied to estimate the parameters via an orthogonal descent optimization that updates the weights and bases iteratively. Additionally, it is also feasible to introduce L1 regularization with the non-negative constraints, which converts NMF into a non-negative sparse coder, as shown in equation (5-34)

$$\begin{aligned}
&\hat{\mathbf{z}}_n = \begin{bmatrix} \hat{\mathbf{x}}_n \\ \hat{\mathbf{y}}_n \end{bmatrix} = \sum\nolimits_{k=1}^{K} w_{k,n} \begin{bmatrix} \mathbf{b}_k^x \\ \mathbf{b}_k^y \end{bmatrix} = \mathbf{w}_n^{\mathrm{T}} \begin{bmatrix} \mathbf{B}^x \\ \mathbf{B}^y \end{bmatrix} \\
&\underset{\mathbf{w},\mathbf{B}}{\arg\min} \sum_{n=1}^{N} \left(\left\| \hat{\mathbf{z}}_n - \mathbf{z}_n \right\|_2^2 + \lambda \left\| \mathbf{w}_n \right\|_1 \right) \\
&s.t. \quad w_{k,n} \geq 0 \\
&s.t. \quad b_{k,m}^x, b_{k,m}^y \geq 0
\end{aligned} \qquad \textbf{(5-34)}$$

Other popular derivations of the conventional NMF include 1) replacing the loss function (e.g. different divergence functions [49]–[52]), and 2) adding convex constraints to the bases [53].

Runtime:

During the runtime stage, the new source feature vector is converted to the optimal weights using the source bases $\mathbf{B}^x$, which estimate the weight by optimizing the following problem (equation (5-35)).

$$\begin{aligned}
&\hat{\mathbf{x}}_n = \sum\nolimits_{k=1}^{K} w_{k,n}\mathbf{b}_k^x = \mathbf{w}_n^{\mathrm{T}}\mathbf{B}^x \\
&\arg\min_{\mathbf{w}} \sum_{n=1}^{N}\left(\left\|\mathbf{x}_n - \hat{\mathbf{x}}_n\right\|_2^2\right) \\
&s.t. \quad w_{k,n} \geq 0
\end{aligned} \qquad \textbf{(5-35)}$$

In this case, the bases are already given, and the objective function becomes a convex function, which is easier to optimize. Finally, the optimal target estimation is obtained by weighted combining the target bases $\mathbf{B}^y$.

#Remark

Note that if the types of feature vectors (e.g. MCC), which does not have non-negative constraints, are involved, they are possible to be factorized using other matrix factorization methods (e.g. PCA). A main benefit of NMF is its relative sparseness [54]. Comparing to the other weighted sum solutions, many elements in the activation matrix provided by NMF are about equal to 0, which reduces the over-smoothing issue.

5.5.2 Non-negative Matrix Deconvolution

Non-negative Matrix Deconvolution (NMD) is an extension of NMF, which utilizes features of multiple frames to calculate the multi-frame basis vectors [55]. As a non-recursive model, an example optimization objective and constraints of NMD is shown in equation (5-36),

$$\begin{aligned}
&\hat{\mathbf{v}}_n = \begin{bmatrix} \hat{\mathbf{z}}_{n-1} \\ \hat{\mathbf{z}}_n \\ \hat{\mathbf{z}}_{n+1} \end{bmatrix} = \sum\nolimits_{k=1}^{K} w_{k,n}\mathbf{b}_k^v = \mathbf{w}_n^{\mathrm{T}}\mathbf{B}^v \\
&\arg\min_{\mathbf{w},\mathbf{B}} \sum_{n=1}^{N}\left(\left\|\hat{\mathbf{v}}_n - \mathbf{v}_n\right\|_2^2 + \lambda\left\|\mathbf{w}_n\right\|_1\right) \\
&s.t. \quad w_{k,n} \geq 0 \\
&s.t. \quad b_{k,m}^x, b_{k,m}^y \geq 0
\end{aligned} \qquad \textbf{(5-36)}$$

Here, $\mathbf{v}_n$ is the dynamic vector which contains the static features (e.g. joint features

$\mathbf{z}_n = [\mathbf{x}_n, \mathbf{y}_n]^{\mathrm{T}}$) of multiple adjacent frames (e.g. the previous one, the current one and the next one). Thus, the size of the bases increases, as they need to capture the information of three frames instead of one frame. The training stage of NMD is almost identical that of NMF. Differently, the runtime stage in NMD will generate multiple frames the step length of one frame. In practice, the overlapping frames are averaged to obtain the output.

The straightforward benefit provided by NMD is that the model will maintain some (i.e. finite length) temporal information of the speech signal. However, two main challenges are also imposed by NMD. Firstly, NMD increases the dimension of feature space (i.e. model complexity), which requires more training data to prevent the over-fitting problem. Secondly, NMD requires much more computation costs for matrix factorization, as multiple feature vectors are concatenated into high-dimension vectors for training and conversion.

5.5.3 Semi Non-negative Matrix Factorization

From a different perspective, the basic assumption of NMF is that both the bases **B** and the weights **w** are non-negative, which is too strong and not suitable for some features, e.g. dynamic features and cepstral coefficients. To this end, the Semi Non-negative Matrix Factorization (Semi-NMF) is proposed to only regulate the activation matrix **w** to be non-negative, but not limit the value range of basis vectors **B**. The experimental results show that the performance of semi-NMF is similar to NMF but is able to process the negative valued features, which extends the flexibility of using different features. Another promising advantage of Semi-NMF is the fast convergence in comparison with NMF, as the Alternating Direction Method of Multipliers (ADMM) [56] can be used for optimization.

5.5.4 NMF with Activation Matrix Mapping

The basic approach of NMF-based mapping considers that the weights **w** are shared by source and target speakers, which is inaccurate for most practical scenarios [57]. To solve this problem, an activity mapping matrix **M** is integrated into the basic NMF model. Instead of learning the joint bases and the weights from the joint vectors **z**, this approach separates the matrix factorization procedure for source and target feature vectors, where two sets of independent NMF bases and weights are learned. Then, a matrix **M** is learned to map the source weights $\mathbf{w}^x$ to the target weights $\mathbf{w}^y$ with minimum error, which can be also considered as applying the linear regression (using **M**) to convert $\mathbf{w}^x$ to $\mathbf{w}^y$. According to the experimental evaluations, the results demonstrate significant improvements in the quality score and the identity score.

5.5.5 NMF using Phoneme-Categorized Dictionary

It is argued that [58], the exemplars estimated by the conventional NMF-based approaches may cause phoneme mismatching between the input feature vector and the selected exemplars. A phoneme-categorized dictionary (i.e. bases) based NMF was proposed to proactively split the original (full size) dictionary into $K = 10$ sub-dictionaries according to Japanese phoneme categories as shown in Figure 5-11.

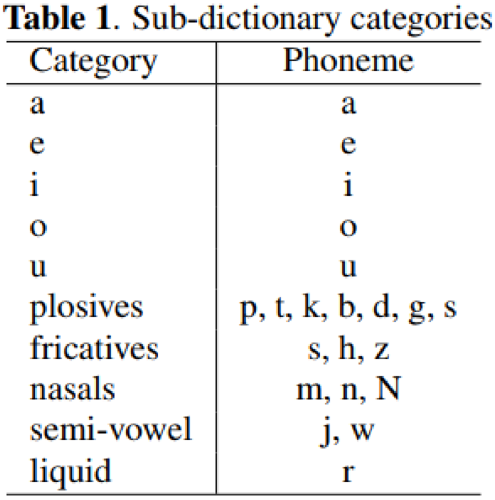


Table 1. Sub-dictionary categories

Category	Phoneme
a	a
e	e
i	i
o	o
u	u
plosives	p, t, k, b, d, g, s
fricatives	s, h, z
nasals	m, n, N
semi-vowel	j, w
liquid	r

Figure 5-11 Categorized Phonemes for Japanese

During the runtime conversion stage, the weight vector **w** is calculated as the conventional NMF using the full-size dictionary. Then, the system selects one of the sub-dictionaries, which maintains the largest activation weights. Based on this sub-dictionary, the new weight vector $\boldsymbol{\omega}$ is re-estimated, and the mapping target feature is reconstructed according to the new weight vector and the sub-dictionary.

5.5.6 NMF-based Frequency Warping and Residue Compensation

According to the concept that the weights can be view as the categorizing index, NMF-based frequency warping and residue compensation is proposed. This method generates exemplar dictionary as the conventional NMF and calculates the frequency warping functions for each exemplar via dynamic warping. In addition, the residue signals are achieved by subtracting the original target exemplars and the mapping exemplar converted by frequency warping.

During the runtime conversion stage, the input source feature vectors are first transformed into weights based on the source bases. Then, the frequency warping functions are regenerated according to weights. Similarly, the compensating residues are constructed using the residue exemplars. For conversion, the source feature vector (i.e. spectral envelope) is first converted via the selected warping function, and the constructed residue is then added to the warped signal.

5.6 Artificial Neural Network

As voice conversion can be treated as a non-linear process to map the source spectrums to the target spectrums, a powerful framework – Artificial Neural Networks (ANN) meets the requirements of voice conversion. In the literature, ANN has been widely studied [59] for voice conversion from various perspectives. As the combination of different ANN technologies is approximately infinite, it is meaningless to list all the possible combinations. To this end, the organization of this sub-section will review the ANN-based voice conversion from three perspectives: 1) operating units, 2) network structures, and 3) timing models. Note that as the multi-layer neural network can extract the higher-order feature from the original input feature, this section distinguishes the two types of features by different words. Specifically, the word “feature” names the original input feature, while the higher-order (from the neural network) is replaced by the word “representation”.

5.6.1 Operating Units

The basic operating unit in ANN denotes the minimal functional component at each layer. There are typically three types of units used for spectrum conversion: 1) perceptron, 2) Autoencoder and 3) Restricted Boltzmann Machine. Some other popular operating units (e.g. the neurons in a self-organizing map) will not be reviewed in this section as they are rarely used in voice conversion applications. The timely units (e.g. Convolution Kernel and LSTM) will be discussed in the later section.

5.6.1.1 Perceptron and Multi-layer Perceptron

Perceptron is the minimal unit in feedforward Multi-layer Perceptron (MLP). It is a function that accepts univariate/multivariate input and produces univariate output as shown in Figure 5-12.

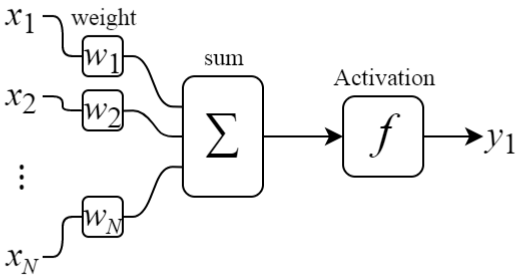


Figure 5-12 Perceptron with the univariate output

Here, *f* is the activation function (e.g. a linear function for regression, a sigmoid function for 0/1 decision, a radial basis function for non-linear transformation), *w* is the weight and $\sum$ is the summation function that treats x_n as input. For multivariate output, it is possible to merge perceptrons as shown in Figure 5-13.

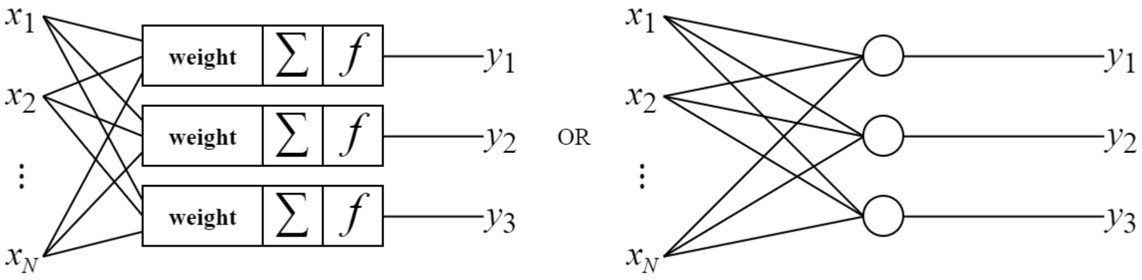


Figure 5-13 Perceptron with the multivariate output

Based on the perceptron, MLP is defined as a model that stacks layers of multi-output perceptrons with at least one (hidden) layer of hidden units. A hidden unit is defined as a perceptron whose output is connected to the inputs of another perceptron [60]. The typical example of 4-layer MLP is shown in Figure 5-14.

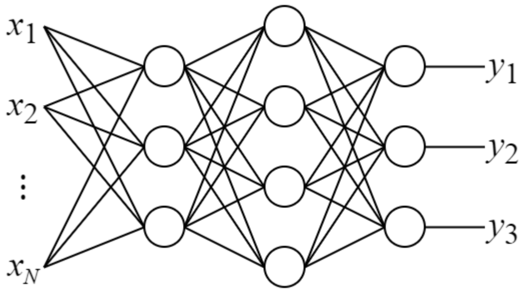


Figure 5-14 Multi-layer Perceptron

The neural network aims at learning the latent representations from the inputs, such that the latent representations can accurately generate the outputs. The mapping function is shown below.

$$g_k(\mathbf{x}) = f(\mathbf{W}_k\mathbf{x} + \mathbf{b}_k) \tag{5-37}$$

$$\hat{\mathbf{y}} = \bigodot_{p=1}^{P} g_p(\mathbf{x}) = g_P \circ g_{P-1} \circ \cdots \circ g_1(\mathbf{x}) = g_P\left(g_{P-1}\left(\cdots g_1(\mathbf{x})\cdots\right)\right) \tag{5-38}$$

Here, P is the total number of layers, $\odot$ denotes iteratively applying the function g to the input. In order to minimize the squared loss function, a backpropagation algorithm [59] can be applied to estimate the parameters $\mathbf{W}_k$ and $\mathbf{b}_k$ for each layer.

$$\underset{\mathbf{W},\mathbf{b}}{\arg\min}\, \varepsilon, \varepsilon = \sum_{n=1}^{N} \left\| \hat{\mathbf{y}}^{(n)} - \mathbf{y}^{(n)} \right\|^2 \tag{5-39}$$

The approach using multi-layer perception was thoroughly evaluated by Desai et al. [59]. The proposed work is compared to GMM-based approaches, which shows improvements in terms of output quality and speaker identity when various acoustic features (e.g. F_0, MCC and difference of adjacent frames) are considered together.

5.6.1.2 Autoencoder and Stacked Autoencoder

Autoencoder can be treated as a special type of perceptron, where it takes a multivariate input **x** and first maps it to a latent representation **h** through an encoding procedure as shown below,

$$\mathbf{h} = f(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{5-40}$$

Here, *f* is the activation function. Then, the latent representation **h** is expected to reconstruct the input **x**, via a decoding procedure,

$$\hat{\mathbf{x}} = f\left(\mathbf{W}'\mathbf{h} + \mathbf{b}'\right) \tag{5-41}$$

Here, $\hat{\mathbf{x}}$ is the prediction of the original input $\mathbf{x}$. For example, the weighting matrix **W** and the bias **b** can be estimated via a conventional least square criteria as equation(5-42),

$$\underset{\mathbf{W},\mathbf{b}}{\arg\min} \left\| \hat{\mathbf{x}} - \mathbf{x} \right\|^2, \hat{\mathbf{x}} = f\left(\mathbf{W}'\left(f\left(\mathbf{W}\mathbf{x} + \mathbf{b}\right)\right) + \mathbf{b}'\right) \tag{5-42}$$

The graph representation of a single layer autoencoder is shown in Figure 5-15.

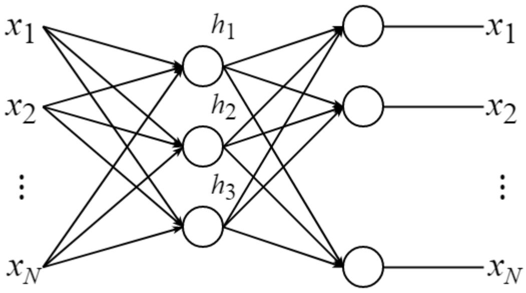


Figure 5-15 Single Layer Autoencoder

Notably, the first difference between the autoencoder and a three-layer perceptron is that the autoencoder is unsupervised training and the perceptron is supervised training. Particularly, the autoencoder set the output as the input to obtain the latent representation of the input feature without any target label, while the perceptron tries to find an optimal function to map the input to the output. The second significant difference is the regulation of weights. In MLP, the weighting matrixes for mappings $\mathbf{W}_1 : \mathbf{x} \mapsto \mathbf{h}$ and $\mathbf{W}_2 : \mathbf{h} \mapsto \mathbf{y}$ are not limited. In the autoencoder, the weighting matrixes for mapping $\mathbf{W} : \mathbf{x} \mapsto \mathbf{h}$and $\mathbf{W}' : \mathbf{h} \mapsto \mathbf{x}$must satisfy a pre-defined rule, such that they are mutually transposed. The training of the parameters can be realized via a conventional gradient descent method.

There are numerous variants of autoencoder, e.g. sparse autoencoder and denoise autoencoder. The original autoencoder targets to learn the under-complete representations (i.e. the dimension of the hidden representation is lower than that of the input vector) from the inputs. The sparse autoencoder targets to extract the overcomplete and sparse representations (i.e. the dimension of the hidden layer is higher than that of the input vector with an L_1 regulation) representations from the inputs. In order to enhance the robustness to noise, the denoise autoencoder adds noise to the inputs and expect the latent representations to generate clean inputs. Similar to perceptron, it is also feasible to stack autoencoders, which enables extracting deep representations from the original input as shown in Figure 5-16.

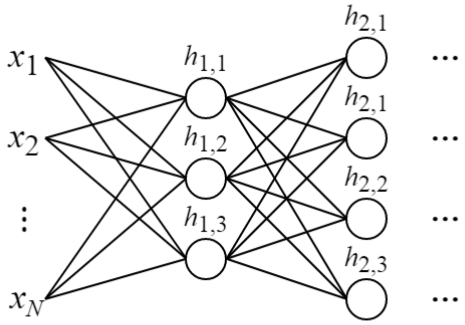


Figure 5-16 Multi-layer Autoencoder

5.6.1.3 Restricted Boltzmann Machine

Sharing the similar encoding/decoding idea as an autoencoder, Restricted Boltzmann Machine [61] (RBM) is an energy-based model that uses a stochastic approach to capture the latent presentations from the input features. The graph representation of RBM is shown in Figure 5-17.

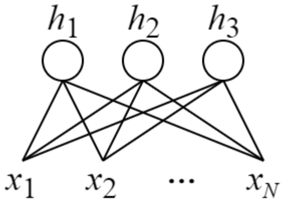


Figure 5-17 Graph for Restrict Boltzmann Machine

An energy-based model computes the probability of being a particular state by normalization. Particularly, the joint energy of a configuration (i.e. input-visible $\mathbf{v}$, output-hidden $\mathbf{h}$ and parameters $\boldsymbol{\theta} = \{\mathbf{a}, \mathbf{b}, \mathbf{W}\}$) is defined as a Boltzmann energy function,

$$\begin{aligned} E(\mathbf{v}, \mathbf{h}) &= -\mathbf{a}^{\mathrm{T}}\mathbf{v} - \mathbf{b}^{\mathrm{T}}\mathbf{h} - \mathbf{v}^{\mathrm{T}}\mathbf{W}\mathbf{h} \\ &= -\sum_i a_i v_i - \sum_j b_j h_j - \sum_i \sum_i v_i w_{i,j} h_j \end{aligned} \tag{5-43}$$

5.6.2 Network Structures

Due to the development of the effective layer-wise pre-training algorithms for deep neural networks, especially the algorithm used in autoencoder and RBM, it is possible to train a reliable deep neural network effectively. In particular, the layer-wise pre-training algorithm makes the optimization procedure “visible” and “controllable” [62] to the researcher, which facilitate investigating suitable approaches for different applications. From another perspective, the layer-wise pre-training can be considered as an automatic representation learning, where the representations that are extracted via an layer-wise unsupervised training achieves significant improvement in terms of accuracy and generalizability compared to the conventional backpropagation algorithm [60].

In addition to the basic operating unit, another important part is how to structure the operating unit to realize a better performance. In other words, the network structure determines the way to connect the operating units and also the conversion accuracy. There are three distinctive network structures that are mainly used for spectrum conversion: 1) Speaker Dependent Encoding, 2) Joint Speaker Encoding, and 3) Joint Speaker Regression.

5.6.2.1 Speaker-Dependent Coding

The basic principle of speaker-dependent coding (SDC) is to train a model that is able to extract the speaker-dependent representations from each speaker. In SDC, the model is bi-directional, which means the speaker-dependent (hidden) representations and the speaker’s inputs can be converted to each other mutually. Under the context of spectrum conversion, two SDC models, one for the source speaker and another one for the target speaker, are trained. The extracted hidden representations are further utilized for training a source-to-target mapping model (e.g. a regression model). The basic structure is shown in Figure 5-18. Note that it is always possible to stack operating units to enable the deep structure. In particular, the hidden representation can be further decomposed into the next-layer hidden representation and the next-layer model to extract the deep nonlinear representations from the features.

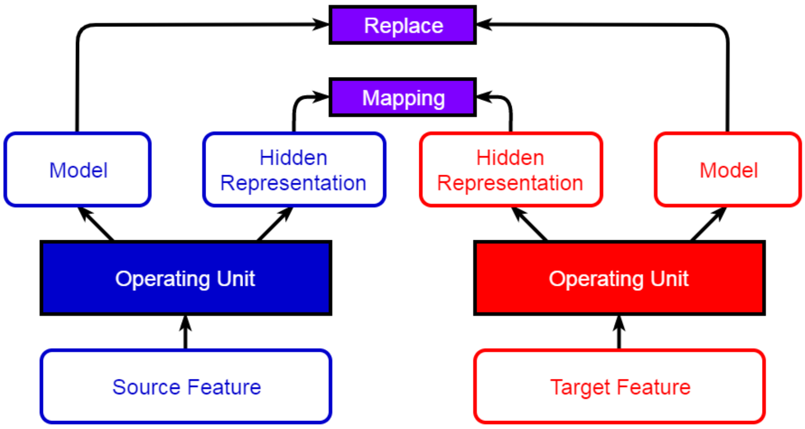


Figure 5-18 Speaker-Dependent Coding Framework

Training:

The target of an SDC model is to extract the latent representations from the original input spectral features (e.g. spectral envelope) for a speaker without any supervised output. To this end, both the autoencoder and RBM are feasible for SDC. In particular, it decomposes the spectral features into an encoding/decoding model and the hidden representations.

For a source-to-target conversion application, a mapping function is learned from the source and target representations via a supervised learning, which means the converted (i.e. from source) hidden representations can be successfully decoded to the target spectral features using the target SDC model.

Runtime:

The spectral features of the source speaker are encoded into latent representations using the source SDC model. Then, the representations are converted using the mapping model. Finally, the converted spectral features of the target speaker are decoded from the converted latent representations using the target SDC model.

5.6.2.2 Joint-Speaker Encoding

Different from SDC that encodes the features of each speaker independently, Joint Speaker Coding (JSC) extracts the shared representations for different speakers, which is very similar to the conventional NMF-based method (Section 5.5.1), where the shared hidden representations (i.e. similar to the weights used in NMF) are extracted from the joint vectors (i.e. by concatenating the source and target feature vectors). In order to enable encoding and decoding between latent representations and spectral features, the model used in JSC must be bi-directional as same as SDC. The basic structure of the JSC mode is shown in Figure 5-19.

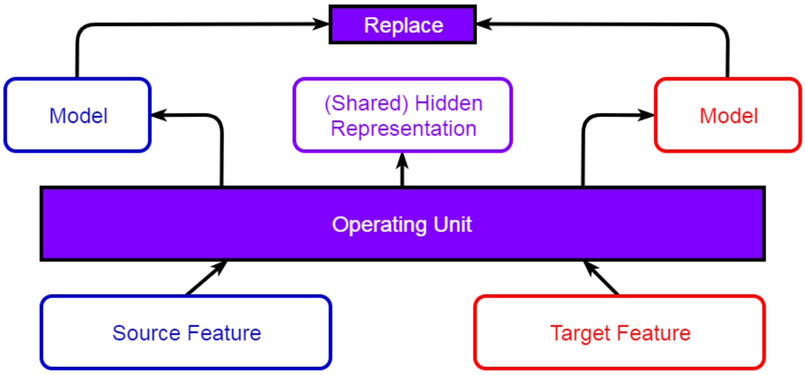


Figure 5-19 Joint-Speaker Encoding Framework

Training:

Under the context of a source-to-target spectrum conversion, a single JSC model (for both the source and target speakers) is trained. The basic assumption of JSC is that the speaker-invariant representations are shared with different speakers. Using JSC, the spectral features of the speaker and target speakers can be decomposed into the speaker-invariant representations (i.e. hidden representations **h**) and a joint model (i.e. weights **W**). The joint model is actually a concatenation of two separated models (i.e. $\mathbf{W} = \left[\mathbf{W_x}, \mathbf{W_y}\right]$), one for the source speaker ($\mathbf{W_x}$) and another one for the target speaker ($\mathbf{W_y}$).

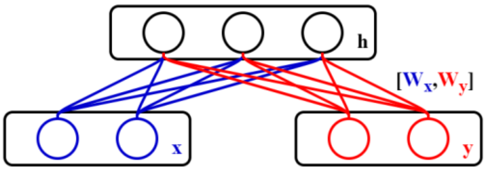


Figure 5-20 Joint-Speaker Encoding Graph

Runtime:

The spectral features of the source speaker are encoded ($\mathbf{x} \mapsto \mathbf{h}$) into speaker-invariant representations (**h**) using the source model ($\mathbf{W_x}$) of the joint JSC. Then the speaker-invariant representations are decoded ($\mathbf{h} \mapsto \mathbf{y}$) into the converted features using the target model ($\mathbf{W_y}$).

JSC only requires one layer to extract the speaker-invariant representations. To realize a deeper network, it is possible to stack SDCs and then use JSC as the top layer as the mapping function. Notably, a special case of JSC is NMF which treats the shared representations (in JSC) as the weights (in NMF) and weights (in JSC) as bases (in NMF).

5.6.2.3 Joint Speaker Regression

In contrast to the encode/decode structure, a joint speaker regression (JSR) structure is feasible for the nonlinear spectral mapping between the source and the target speakers' spectrums. JSR is significantly different from the previous two structures (i.e. SDC and JSC). In JSC and SDC, the operating units need to be bi-directional for encoding/decoding, which is relaxed in JSR. In other words, it is valid to use either the uni-directional discriminant operating units (e.g. perceptrons) or the bi-directional operating units (e.g. Autoencoder or RBM) in JSR, where the bi-directional operating units enable layer-wise pre-trainings.

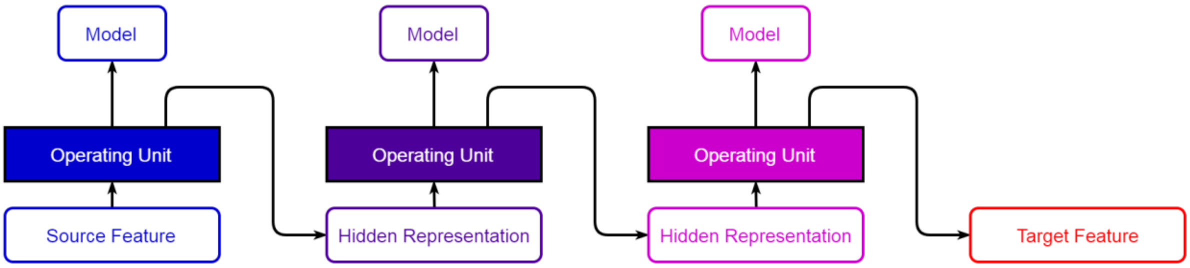


Figure 5-21 Joint Speaker Regression Framework

The initial neural network based voice conversion does use JSR (multi-layer perceptrons [59]). However, as MLP does not natively support layer-wise pre-training and, some noticeable problems are caused by the original backpropagation-based optimization, such as exploding/vanishing gradients and overfitting. With RBM or Autoencoder, JSR can enable layer-wise pre-training to achieve better performance.

Training:

A significant difference between JSR and SDC is the order of training units. Specifically, SDC trains speaker-dependent network for each speaker independently, then the mapping model is learned to convert the latent representations. In contrast, JSR trains the network from the input (source feature vectors) to the output (target feature vectors) layer-by-layer, and a back propagation algorithm is required to fine-tune the parameters.

Runtime:

In JSR, the runtime process is very straightforward, the source features are fed into the input layer, the features/representations are transformed layer-by-layer, and the output layer will generate the converted representations.

5.6.3 Timing Models

Due to the inherent timing characteristics of speech signals, the timing models seem to be more suitable for modelling. In particular, the convolutional neural network (CNN) and recurrent neural network (RNN) which can capture the timing information is effective for spectral mapping.

5.6.3.1 Convolutional Neural Network

In general, CNN enables extracting spatial representations using convolution kernels. The design principle can be diversely applied to various signal processing tasks (e.g. image processing and speech processing). For spectrum conversion, an example is CNN operates on the time-frequency domain to collect the representations using a two-dimension FIR filter (i.e. moving weighted average) as shown below,

$$\begin{aligned}\mathrm{conv}\left(\mathbf{X},\mathbf{W}\right)_{u,v} &= \underbrace{\left\langle \begin{bmatrix} x_{u-1,v-1} & x_{u-1,v} & x_{u-1,v+1} \\ x_{u,v-1} & x_{u,v} & x_{u,v+1} \\ x_{u+1,v-1} & x_{u+1,v} & x_{u+1,v+1} \end{bmatrix}, \begin{bmatrix} w_{0,0} & w_{0,1} & w_{0,2} \\ w_{1,0} & w_{1,1} & w_{1,2} \\ w_{2,0} & w_{2,1} & w_{2,2} \end{bmatrix} \right\rangle}_{\text{inner product}} \\ &= \sum_{i=0}^{2}\sum_{j=0}^{2} x_{u+i-1,v+j-1} w_{ij}\end{aligned} \qquad \textbf{(5-44)}$$

Here, u is the time index and v is the frequency index. Additionally, CNN can also be extended to process the dynamic information of speech. For instance, convolution can be applied to the 1st order difference vectors between adjacent frames as shown below.

$$\mathrm{conv}(\mathbf{X},\mathbf{W})_{u,v} = \underbrace{\left\langle \begin{bmatrix} \overbrace{x_{u-1,v}-x_{u-1,v-1}}^{\Delta\mathbf{x}_{\mathbf{v}}} & \overset{\mathbf{x}_{\mathbf{v}}}{x_{u-1,v}} & \overbrace{x_{u-1,v+1}-x_{u-1,v}}^{\Delta\mathbf{x}_{\mathbf{v+1}}} \\ x_{u,v}-x_{u,v-1} & x_{u,v} & x_{u,v+1}-x_{u,v} \\ x_{u+1,v}-x_{u+1,v-1} & x_{u+1,v} & x_{u+1,v+1}-x_{u+1,v} \end{bmatrix}, \begin{bmatrix} w_{0,0} & w_{0,1} & w_{0,2} \\ w_{1,0} & w_{1,1} & w_{1,2} \\ w_{2,0} & w_{2,1} & w_{2,2} \end{bmatrix} \right\rangle}_{\text{inner product}} \tag{5-45}$$

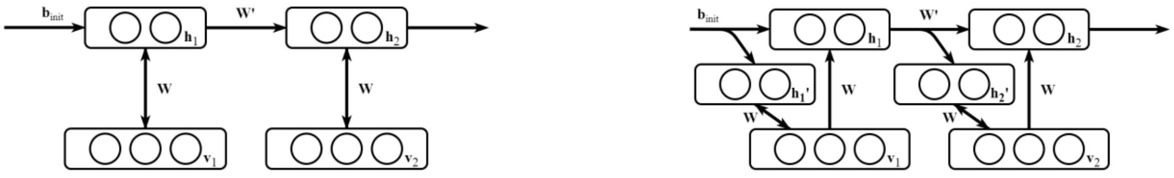
A significant problem of CNN is the finite memory. For speech signals, the long-term historical frames may still affect the processing of the current frame. However, CNN lacks the ability to dynamically adjust the size of convolution kernels, which motivates the development of the RNN based voice conversion.

5.6.3.2 Recurrent Neural Network

Different from CNN that utilizes finite historical samples (a frame in a time slot), RNN supports storing infinite historical information [63], which theoretically enables more powerful modelling than CNN. The early-stage RNN models suffer from the problem of vanishing gradient. To this end, RNNs (e.g. Long Short-Term Memory and Temporal Restricted Boltzmann Machines) were utilized to improve the performance.

Recurrent Temporal Restricted Boltzmann Machine based Spectrum conversion:

Temporal Restricted Boltzmann Machine (TRBM) is an unsupervised model that extends Restricted Boltzmann Machine (RBM) to model the timing information (e.g. state or memory) as shown in Figure 5-22. Aiming at tackling the intractable inference problem [64] in TRBM, Recurrent Temporal Restricted Boltzmann Machine (RTRBM) [64] were proposed to simplify the inference procedure (from hidden representations to visible inputs) of TRBM and achieves the remarkable accuracy and efficiency. The structure of a typical RTRBM is shown in Figure 5-22.



(a) Temporal RBM

(b) Recurrent Temporal RBM

Figure 5-22 Two Timing RBMs

Under the context of spectrum conversion, a model that combines RTRBM and Speaker Dependent Coding was proposed, namely Speaker Dependent RTRBM (SD-RTRBM). SD-RTRBM targets to model the spectral and timing information for the source and target speakers independently. After the deep representations ($\mathbf{h}_x$ and $\mathbf{h}_y$) of two speakers (i.e. source *x* and target *y*) are obtained, a supervised regression neural network is trained to map the source feature $\mathbf{h}_x$ to the target feature $\mathbf{h}_y$. Finally, the parameters of two SD-RTRBM models and the regression model are further fine-tuned by a backpropagation algorithm. Compared to the GMM-based approach that uses an additive model of piecewise linear functions, SD-RTRBM first extracts deep timing representations and then utilizes a non-linear mapping function to process spectrums. The experimental results show a significant improvement in comparison with GMM based solutions.

Bidirectional Long Short-Term Memory Network based Spectrum conversion:

Long Short-Term Memory Network (LSTM) network is a special kind of RNN that is capable of capturing the long-term timing information and avoid the gradient vanishing problem [64], [65]. Instead of pre-defining the period that a memory should be kept, it is the default behaviour of uni-directional LSTM (Uni-LSTM) to dynamically determine the memory length by four interacting functions as shown in Figure 5-23,

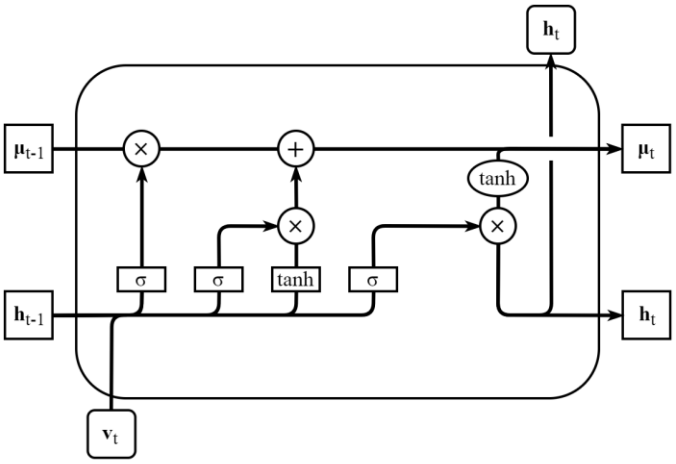


Figure 5-23 Uni-directional LSTM Operating Unit

Here, $\boldsymbol{\mu}$ is the memory weighting factor from the (in time) training of the previous sample, **h** is the hidden representations generated by the current input x and the timing information. The latent representations are also fed to the training of the next sample as memory.

Theoretically, the unidirectional RNNs (e.g. uni-LSTM) can only use the previous timing information but not the futures, while a bi-directional RNN can extract the latent information using the context information in both forward and backward directions elegantly. According to the literature, the bi-directional RNN significantly outperforms uni-directional RNNs on numerous tasks of sequence modelling (e.g. Natural Language Learning, Automatic Speech Recognition and Text-to-Speech Synthesis).

Under this context, Sun et al. [65] apply an alternative LSTM architecture – Bidirectional Long Short-Term Memory (Bi-LSTM) for spectrum conversion. The network structure is shown in Figure 5-24,

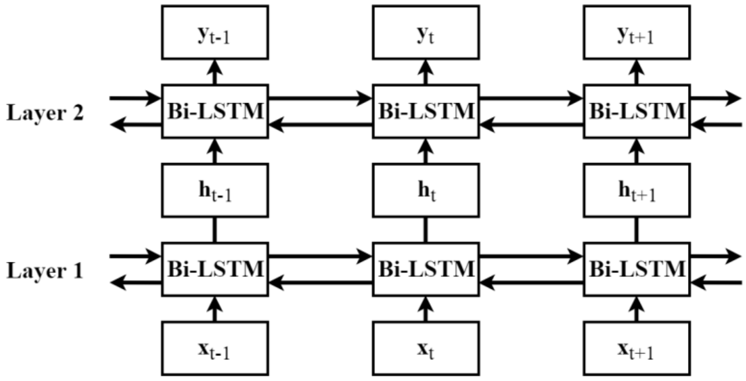


Figure 5-24 Spectrum conversion using Bi-LSTM

The parameters of the Bi-LSTM network are trained via backpropagation through time (BPTT) without layer-wise pre-training. For practical consideration, the weight gradients are computed for one sentence at a time. The evaluations show remarkable improvements in comparison with stacked Speaker Dependent Restrict Boltzmann Machine (SD-RBM) without historical information. Additionally, the results demonstrate that if Bi-LSTM is used, the dynamic modelling becomes negligible, and hence reduces the computational cost. However, the key weak aspect of this approach is that the system requires a full sentence in the runtime conversion stage. Otherwise, the reverse direction LSTM will not work.

In addition to LSTM, Gate Recurrent Unit (GRU) is a famous variation of LSTM. It controls the flow of information like the LSTM unit, but without having to use a memory unit. It just exposes the full hidden content without any control. A comparative study [66] has been carried out for LSTM and GRU, the experimental results show that LSTM achieves less MCD and higher identity than GRU.

5.6.4 Remarks

ANN gives nearly infinite possibilities for voice conversion applications by combining the different existing techniques. For instance, it is feasible to model the spectral features using a timing model or a non-timing model; the mapping function between the extracted latent representations (from the source and target speakers) can be a timing model or a non-timing model; and it is possible to utilize the different off-the-shelf derivations of the conventional operating units to construct the network.

The core idea of ANN is to extract the valuable representations from the inputs. In fact, the spectral features (as the inputs) of speech are usually very sparse in a specific domain. MCC is a convincing example that utilizes twenty a few coefficients to encode any full-size spectral envelope with a low distortion, which means that MCC is already an effective/compact representation of the spectral envelope. It is questionable to use the non-timing ANN model to extract the deep representations from the spectral envelope or even from the MCC. To this end, a study should be carried out on comparing the performance of using MCC and using ANN for representation learning.

5.7 Summary and Conclusions

This chapter presents a comprehensive review of the spectrum mapping technique which is the most important part of a voice conversion system. In addition, the pros and cons of different approaches are discussed. From a practical point of view, the frequency warping method guarantees high perceptual qualities but degenerates perceptual identifies. This motivates the subsequent research work, i.e. Continuous Frequency Warping and Magnitude Scaling (CFWMS), which is presented in the next chapter.

Chapter 6

Low-latency Voice Conversion System

Voice conversion morphs the utterance of a source speaker, such that the converted speech is perceived as been spoken by a **specific** target speaker [4]. In other words, it keeps the linguistic information and modifies the non-linguistic information such as pitch, formants, breathes and behaviour. The scope of this chapter is to enable the low-latency spectral and prosody conversion for real-time applications by proposing a novel method – Continuous Frequency Warping and Magnitude Scaling (CFWMS) under the Joint Density Gaussian Mixture Model (JD-GMM) framework. Particularly, CFWMS enables a continuous spectral manipulation in the frequency domain through both directions (i.e. vertical and horizontal).

The proposed approach (i.e. CFWMS) estimates an optimal warping function based on the formant-peak density functions without relying on the weak frame-level relationship between source and target [67]. Then, the magnitude scaling method corrects the spectral power in each sub-band, which guarantees the converted signals to have the similar sub-band energy distribution as the target signals. In addition, a novel strategy is proposed to limit the classification trajectory which prevents irrational discontinuities between frames. As the proposed strategy only requires the current and the historical information, it is suitable for real-time applications. The performance of the proposed CFWMS is compared with the conventional solutions – JD-GMM (Joint Density Estimation) [42] and DKPLS (Dynamic Kernel Partial Least Square) [16], the experimental results show an improving performance in terms of speech quality and perceptual identities.

The low-latency voice conversion is an independent chapter that is not highly related to the previous two chapters (i.e. Chapter 2 and Chapter 3), as it requires a recent proposed high-quality VOCODER – WORLD [68]. Section 6.1 will show the data pre-processing and clustering procedures before training the conversion function. The proposed conversion scheme – CFWMS will be presented in Section 6.2. Section 6.3 illustrates the novel trajectory limitation strategy. The evaluation results are shown in Section 6.4.

6.1 Data Pre-processing and Clustering

As mentioned in the previous section (Section 2.3), a conventional voice conversion system contains two stages: 1) the training stage and 2) runtime conversion stage. Figure 6-1 presents the proposed voice conversion procedures.

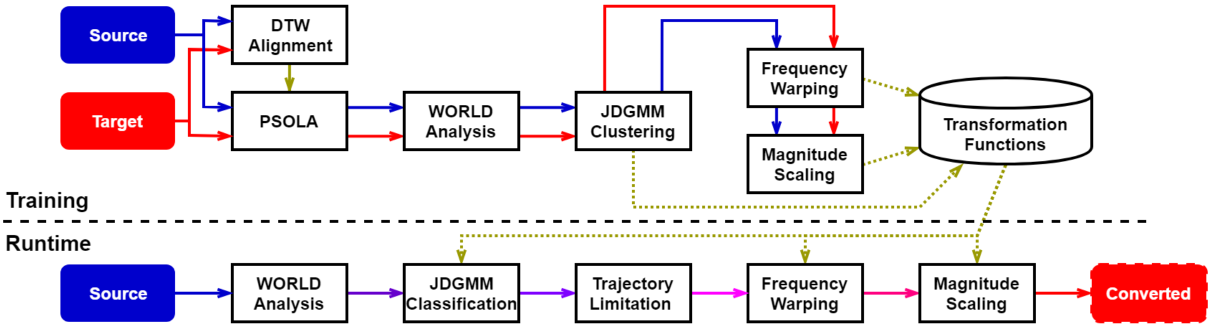


Figure 6-1 Overview of the proposed voice conversion

6.1.1 Time Alignment

The speech signals from both the source speaker and the target speaker are first aligned using Dynamic Time Warping (DTW) [22]. Based on the optimal alignment function, the speech signals are time-scaled via Pitch Synchronous Overlap-and-Add (PSOLA) technology [9] to guarantee the proper match of phonemes.

6.1.2 Feature Extraction

The speech signals from both the source and target speakers are analysed at frame-level to extract the spectral features (e.g. spectral energy, pitch/F_0, formant and aperiodicity [11]) via the WORLD vocoder [68]. WORLD is the computationally efficient version of STRAIGHT [11] which is widely used in speech synthesis applications. Specifically, the extracted formant envelope is encoded into Mel-Cepstral Coefficients (MCC) [21] to minimize the correlation between coefficients. This leads to significant benefits which simplify the covariance estimation in GMM clustering [69] and avoid the over-fitting problem.

6.1.3 Joint Density Clustering

Let $x^{(n)}$ and $y^{(n)}$ be the *M*-dimension MCC vectors (*x*: source, *y*: target) at the frame *n*. The distribution of the joint MCC vectors $z^{(n)} = \left[x^{(n)}, y^{(n)}\right]$ can be modelled via JD-GMM as in equation (6-1),

$$P\left(z^{(n)}\right) = \sum_{k=1}^{K} w_k N\left(z^{(n)}; \boldsymbol{\mu}_k^z, \boldsymbol{\Sigma}_k^z\right) \tag{6-1}$$

GMM assumes the distribution of $z^{(n)}$ is the weighted sum of *K* different multivariable Gaussian distributions. The index of each Gaussian component is *k*. The weight of the *k*th component is w_k. The mean $\boldsymbol{\mu}_k^z$ and variance $\boldsymbol{\Sigma}_k^z$ of each component *k* is composed as in equation (6-2).

$$\boldsymbol{\mu}_k^z = \begin{bmatrix} \boldsymbol{\mu}_k^x \\ \boldsymbol{\mu}_k^y \end{bmatrix} \quad \boldsymbol{\Sigma}_k^z = \begin{bmatrix} \boldsymbol{\Sigma}_k^{xx} & \boldsymbol{\Sigma}_k^{xy} \\ \boldsymbol{\Sigma}_k^{yx} & \boldsymbol{\Sigma}_k^{yy} \end{bmatrix} \tag{6-2}$$

Here, $\boldsymbol{\mu}_k^x$ and $\boldsymbol{\Sigma}_k^{xx}$ denotes the mean and auto-covariance of the *k*th component of the source speakers. $\boldsymbol{\mu}_k^y$ and $\boldsymbol{\Sigma}_k^{yy}$ denotes the mean and auto-covariance of the *k*th component of the target speaker. The matrixes $\boldsymbol{\Sigma}_k^{xy}$ and $\boldsymbol{\Sigma}_k^{yx}$ denote the cross-covariance between the source and target speakers of the *k*th component. As MCC minimizes the correlation between coefficients, the covariance matrixes $\boldsymbol{\Sigma}_k^{xx}$, $\boldsymbol{\Sigma}_k^{xy}$, $\boldsymbol{\Sigma}_k^{yx}$ and $\boldsymbol{\Sigma}_k^{yy}$ for clustering are assumed to be diagonal.

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1^2 & & 0 & \\ & \sigma_2^2 & & \\ & & \ddots & \\ 0 & & & \sigma_M^2 \end{bmatrix} \tag{6-3}$$

After the parameters of JD-GMM are estimated via an Expectation Maximization (EM) algorithm, the posterior probability of any source sample belong to the cluster *q* is calculated as follows:

$$P\left(q \mid x^{(n)};\lambda_z\right) = \frac{w_q N\left(x^{(n)};\mathbf{\mu}_q^x,\mathbf{\Sigma}_q^{xx}\right)}{\sum_{k=1}^{K} w_k N\left(x^{(n)};\mathbf{\mu}_k^x,\mathbf{\Sigma}_k^{xx}\right)} \tag{6-4}$$

Here, the posterior probability $P\left(q \mid x^{(n)};\lambda_z\right)$ is further utilized by frequency warping and magnitude scaling as the weights of applying the *k*^th^ conversion function to the input source feature vector $x^{(n)}$.

6.2 Frequency Warping with Magnitude Scaling

6.2.1 Overview

Frequency Warping and Magnitude Scaling operates on the formant envelope (extracted by the WORLD vocoder) directly. Note that, the spectral features (i.e. the formant envelope) are weighted, i.e. by multiplying the posterior probability $P\left(q \mid x^{(n)};\lambda_z\right)$, before training the k^{th} mapping function.

6.2.2 Frequency Warping

This procedure aims to find out the optimal spectral warping function, such that the source formant peaks can be moved to the target location. Note magnitude information is removed because it does not contribute to peak alignments.

6.2.2.1 Formant Peak Density

The formant peak density of the source and target speaker are estimated from the phase of the complex roots of the LPC coefficients (as a polynomial) estimated from the formant envelope as shown in equation (6-5).

$$\begin{aligned} &\varphi = \left| \angle\left(\arg\left(\overbrace{\prod_i \left(a_i x^2 + b_i x + c_i\right)}^{\text{LPC Polynomial}} = 0 \right)\right)\right| \\ &s.t. \quad b_i^2 - 4a_i c_i < 0 \end{aligned} \tag{6-5}$$

Here, *i* denotes the formant index. The rationale is that LPC is an all-pole model and the phase of each pair of conjugate poles reflects the normalized frequency of formant peaks (resonant frequency) [35]. Then, the formant peak density $Q[l]$ in the l^{th} frequency bin is estimated via Parzen window estimation. In practice, $l=512$ bins are used here.

$$Q[l]=\frac{1}{512}\sum_{n=1}^{G}\kappa\left(g^{(n)};\mu_l,\sigma\right)$$
$$\mu_l=\frac{l}{512},\sigma=\frac{1}{160}. \qquad \textbf{(6-6)}$$

Here, G denotes the total number of formant peak frequencies $\mathbf{G}=\left\{g^{(1)},\ldots,g^{(G)}\right\}$, κ is the Gaussian kernel function with the (normalized) equidistant centres μ_l and the (normalized) same width $\sigma=\frac{1}{160}$,

6.2.2.2 Loss Function

The dynamic warping aims to estimate an optimal function that maps the formant peak density from source to target with minimum accumulated cost *C*.

$$\begin{aligned}
&C\left(\mathbf{Q}^x,\mathbf{Q}^y\right)=\underset{\mathbf{r},\mathbf{v}}{\arg\min}\left(\sum_{j=1}^{J}\Delta\left(Q^y\left(r_j\right),Q^x\left(v_j\right)\right)\right)\\
&s.t.\quad \begin{matrix}\forall r_j\in[1,2,\ldots,512]\\ \forall v_j\in[1,2,\ldots,512]\end{matrix}\\
&s.t.\quad r_{j+1}\geq r_j,v_{j+1}\geq v_j
\end{aligned} \qquad \textbf{(6-7)}$$

The first constraint assures that the warping function $\left[r_j,v_j\right]$ starts at $[1,1]$ and ends at $[512,512]$. The second constraint guarantees the monotonicity of the warping function. $Q^x\left(v_j\right)$ and $Q^y\left(r_j\right)$ denotes the density at the frequency bin $\left[r_j,v_j\right]$ for the source and the target speakers. The optimal path is defined as the directed path (inside a graph with a set of vertices $\left\{\left[r_j,v_j\right]|\,\forall r_j\in\mathbf{r}\wedge\forall v_j\in\mathbf{v}\right\}$) with minimum accumulated cost *C*, which indicates the maximized similarity between the source and the target probability densities. The Chi-square test statistic is used here for interpretable measurement between non-parametric density functions. In particular, Δ denotes the chi-square test statistic between two probability densities at the bin *l* of the target and the bin *k* of the source.

$$\Delta\left(Q_{tgt}(l),Q_{src}(k)\right)=\frac{\left(Q_{tgt}(l)-Q_{src}(k)\right)^2}{Q_{tgt}(l)+Q_{src}(k)} \qquad \textbf{(6-8)}$$

The all eligible warping path constructs a directed graph (as shown in Figure 6-2, with a set of vertices) with minimum accumulated cost (as the red parts in Figure 6-2), which indicates the maximized similarity (minimize the accumulated Δ) between the source and the target formant density functions.

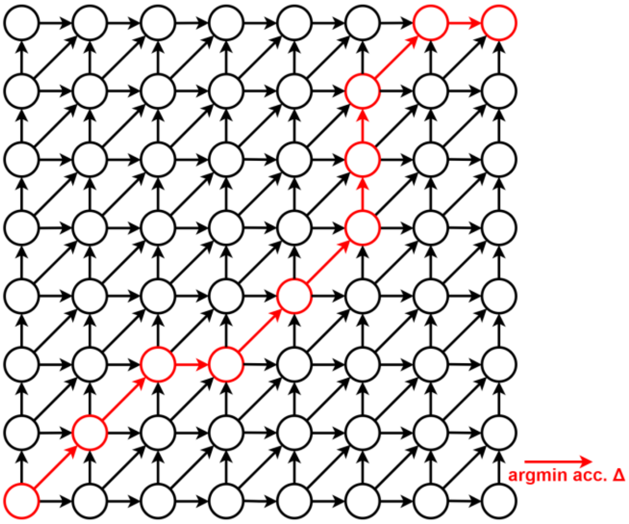


Figure 6-2 Optimal Path of Dynamic Frequency Warping

6.2.2.3 Optimal Path

Estimating the optimal warping function to minimize the accumulated cost is a shortest-path problem, which was usually solved by a greedy algorithm in DTW [70]. However, a more accurate solution can be achieved via the Breadth-First Searching (BFS, e.g. Dijkstra [71]) algorithm. Two types of algorithms can be used to estimate the warping function, which includes greedy search and BFS. The greedy search is cost-efficient but does not guarantee the global optima. In contrast, BFS guarantees the global optima but is computationally expensive. The complexity of BFS in the worst situation is $O(E+V)$; which is equivalent to $512\times3+512=2048$, which is actually affordable for an offline training. For further information and different implementations, it is suggested that the readers can find more details in [71]. In addition, as the searching algorithm is no longer needed after training, it is not against the principle of low computational costs for real-time applications.

6.2.3 Magnitude Scaling

The speaker individuality of the speech is insufficiently converted only by frequency wrapping, as the spectral energy in the different sub-bands is inaccurate. This section describes a solution that compensates the magnitude horizontally in the frequency domain, such that the difference is minimized. Note that the magnitude scaling function is trained from the spectrum modified by frequency warping.

6.2.3.1 Sub-band Energy

The magnitude scaling is not performed on each single frequency point but on each sub-band. The Mel-frequency filter bank [72] splits the spectrum into sub-band components which guarantees a perfect reconstruction. Particularly, 16 bins (filters) are used for analyzing and constructing the spectrum (Figure 6-3).

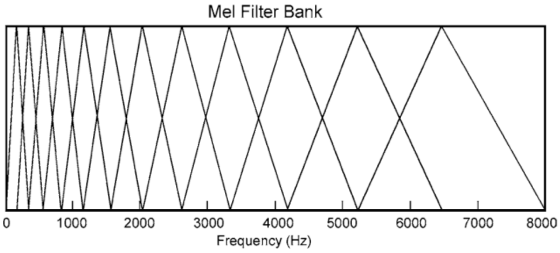


Figure 6-3 Mel Scale Filter Banks

For each sample *n*, each sub-band *β* and each cluster *k*, the energy $u_{\beta,k}^{x,(n)}$ and $u_{\beta,k}^{y,(n)}$ of source *x* and target *y* samples are estimated as in (6-9),

$$\begin{aligned} u_{\beta,k}^{x,(n)} &= \int v_{\beta}(\omega)\rho_k^{(n)} x^{(n)}(\omega)\, d\omega \\ u_{\beta,k}^{y,(n)} &= \int v_{\beta}(\omega)\rho_k^{(n)} y^{(n)}(\omega)\, d\omega \end{aligned} \qquad \textbf{(6-9)}$$

Here, $v_{\beta}(\omega)$ is the β^{th} triangular window function shown in Figure 6-3, $\rho_k^{(n)}$ denotes the posterior probability (i.e. $P\left(k \mid x^{(n)}; \lambda_z\right)$, equation (6-4)) that the *n*th sample belongs to the *k*th GMM cluster and *ω* denotes the normalized frequency index. The objective of the magnitude scaling is to modify the source spectrum vertically, such that the spectral energy is converted. This is realized by splitting the spectrum into sub-bands and modifying the energy of each sub-band by the linear scaling function.

6.2.3.2 Linear Scaling

The explicit relationship between the source and target weighted sum magnitude $u_{\beta,k}^{x,(n)}$ and

$u_{\beta,k}^{y,(n)}$ is estimated via a regression model. The regression is applied to minimize the square error given as in equation (6-10),

$$\underset{d,b}{\arg\min}\left(\sum\nolimits_{n=1}^{N}\left\|d_{\beta,k}u_{\beta,k}^{x,(n)}-u_{\beta,k}^{y,(n)}+b_{\beta,k}\right\|_2\right) \qquad \textbf{(6-10)}$$

The regression is applied to each sub-band *β* and to each cluster *k* independently. The optimal estimation of the energy can be achieved via the Normal equations and the optimal estimation is given as in equation (6-11),

$$\mathbf{u}_k^{\hat{y},(n)}=\left(\mathbf{d}_k\mathbf{I}\right)\mathbf{u}_k^{x,(n)}+\mathbf{B}=\begin{bmatrix} d_{\beta=1,k} & 0 & 0 & 0 \\ 0 & d_{\beta=2,k} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & d_{\beta=B,k} \end{bmatrix}\begin{bmatrix} u_{\beta=1,k}^{x,(n)} \\ u_{\beta=2,k}^{x,(n)} \\ \vdots \\ u_{\beta=B,k}^{x,(n)} \end{bmatrix}+\begin{bmatrix} b_{\beta=1,k} \\ b_{\beta=2,k} \\ \vdots \\ b_{\beta=B,k} \end{bmatrix} \qquad \textbf{(6-11)}$$

According to the equation (6-10), the objective of the energy compensation function is to manipulate the spectrum in each sub-band, such that the energy of each sub-band is converted to the target one. It results in the amplitude mapping function as shown in equation (6-12)

$$\hat{y}^{(n)}=\sum\nolimits_k\rho_k^{(n)}\left(\sum\nolimits_\beta\frac{v_\beta(\omega)x^{(n)}(\omega)u_{\beta,k}^{y,(n)}}{u_{\beta,k}^{x,(n)}}\right) \qquad \textbf{(6-12)}$$

The complexity to calculate the equation is quadratic with time, in order to further reduce the complexity, the equation (6-12) needs to be simplified. Obviously, the source vector $x^{(n)}$ is unrelated to the summations, and $u_{\beta,k}^{\hat{y},(n)}=d_\beta u_{\beta,k}^{x,(n)}$, the equation (6-12) is converted to the equation (6-13).

$$\begin{aligned}\hat{y}^{(n)}&=x^{(n)}(\omega)\left(\sum\nolimits_k\rho_k^{(n)}\sum\nolimits_\beta v_\beta(\omega)\left(\frac{d_{\beta,k}u_{\beta,k}^{x,(n)}}{u_{\beta,k}^{x,(n)}}\right)\right)+\sum\nolimits_k\rho_k^{(n)}\left(\sum\nolimits_\beta v_\beta(\omega)b_{\beta,k}\right)\\&=x^{(n)}(\omega)\sum\nolimits_k\left(\rho_k^{(n)}\sum\nolimits_\beta v_\beta(\omega)d_{\beta,k}\right)+\sum\nolimits_k\left(\rho_k^{(n)}\sum\nolimits_\beta v_\beta(\omega)b_{\beta,k}\right)\end{aligned} \qquad \textbf{(6-13)}$$

Note that the two summations $\sum_k\left(\rho_k^{(n)}\ldots\right)$ can be pre-computed in an offline manner, which largely reduces the computation load of runtime conversion.

6.3 Trajectory Limitation

The proposed CFWMS solution is a frame-level mapping scheme. However, this design ignores the relationship between adjacent frames and potentially causes discontinuity [24], [28] in output speech. In order to tackle this problem, two types of methods have been studied, namely, 1) trajectory [24] and 2) dynamic feature [16], [24].

6.3.1 Spectral Trajectory

To tackle the discontinuity issue, Toda *et al.* [24] proposed a global model that targets to generate the optimal spectral trajectories using sentence-level maximum likelihood estimation. Related studies [73] have shown the effectiveness of the trajectory-based model. However, this model requires the global information of a full sentence. Moreover, the computation is enormous even in the runtime conversion stage, which limits its feasibility in real-time applications.

6.3.2 Dynamic Feature

From the perspective that the global information is not available, another type of model introduces the dynamic feature in addition to the static feature. A typical example is the first–order difference vector, as shown in (6-14)

$$\Delta x^{(i)} = x^{(i)} - x^{(i-1)} \tag{6-14}$$

Here, $x^{(i)}$ and $x^{(i-1)}$ denotes two adjacent feature vectors. The dynamic feature aims to maintain a smoother transition between frames. In fact, the dynamic feature still suffers from noticeable distortion, if the classifications of adjacent frames are not related.

6.3.3 Problem Statement

According to the previous study [24], the significant culprit of trajectory discontinuity is the one-to-many mapping under the clustering framework. For example, the one-to-many mapping can be easily observed between the source and target speech signals. For instance, two words “hello” and “steady” maintain the same phoneme /ə/. The source speaker can pronounce this phoneme with very similar timbre and the target speaker pronounces this phoneme with two different timbres.

As the joint-density clustering (JD-GMM) concatenates the source features and target features into joint features and generates an independent conversion function for each cluster (i.e. two clusters may contain the same source pronunciation and different target pronunciations) at the training stage. During the runtime stage, as only the feature vectors of the source speaker are available, the frame–level classifications cannot guarantee adjacent frames with the same pronunciation to be categorized into the same class with the same mapping function. The discontinuous classification is shown in Figure 6-4.

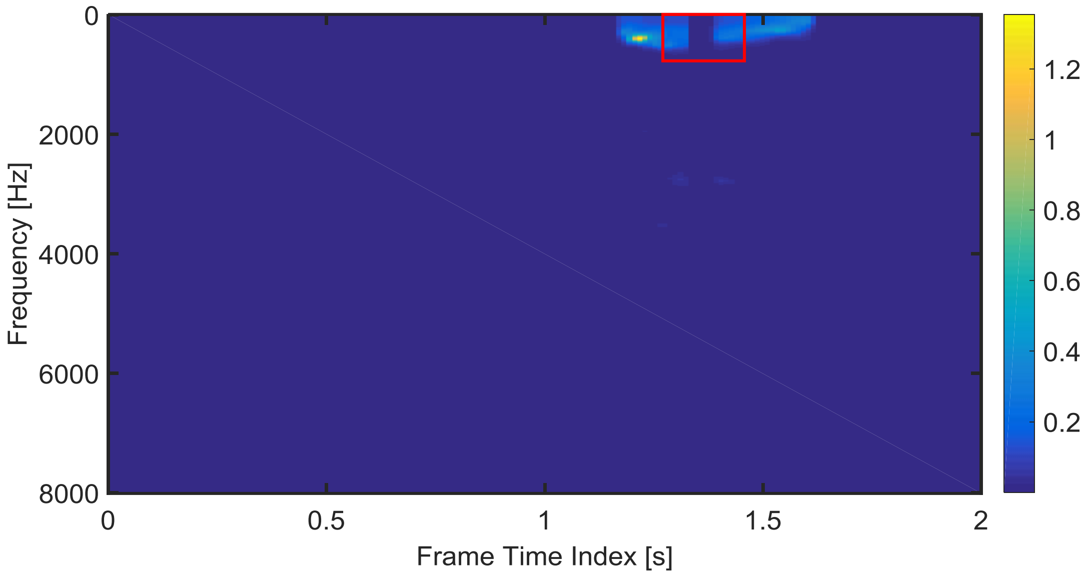


Figure 6-4 Example of Trajectory Discontinuity

This figure shows the frames that are most likely to belong to the 20th cluster. It is significant that some of the frames (i.e. dark gap) within the red rectangular are **NOT** classified into the 20th cluster. As the result, these frames may be converted via an improper warping and scaling function, which consequently causes noticeable quality degenerations. Although dynamic features are able to capture a little more sequential information (i.e. by using the previous and the next frames), it still cannot immune from the misclassification and degenerate speech quality.

6.3.4 Classification Trajectory Limitation

Following the principle of an offline spectral trajectory estimation method [28], this section further investigates an efficient online trajectory limitation method – Classification Trajectory Limitation (CTL) to prevent unreasonable transition between frames using only the historical information.

The principle of the CTL is to decide whether the current trajectory point needs to be changed. Correspondingly, there is a visible factor that can highly impact to the decision:

“The similarity between two adjacent frames”

In particular, if the adjacent frames are very similar, the trajectory point is unlikely to be changed. Otherwise, the trajectory point is likely to be changed. Based on this principle, CTL justifies if the trajectory is reasonable by monitoring the formant difference between the current frame and the previous one frame. If the adjacent frames belong to the same phoneme, the difference will not exceed a pre-defined threshold H. Otherwise, the two frames are considered as belonging to two different phonemes.

The trajectory change (i.e. a different frame-level classification result) is valid only if the current frame belongs to a different cluster compared to the previous frame. Otherwise, the current frame should maintain the similar classification as the previous frame, which regulates the trajectory to be smooth and continuous.

It is worth to note that, CTL does not always guarantee the optimal trajectory especially when the first frame of a phoneme segment is wrongly classified. This will result the following frames being misclassified as well. However, this will be corrected at the next phoneme, when the trajectory changes. The detailed enhancement of CTL is considered as one of the future work.

6.4 Performance Evaluation on Voice Gender Conversion

The proposed solution was compared with two existing approaches, namely, JD-GMM and DKPLS. JD-GMM [42] is the conventional baseline widely used for comparison. DKPLS (Dynamic Kernel Partial Least Square) [16] is a recent state-of-the-art method that employs Kernel PLS Regression for dynamic features. The CMU ARCTIC databases [74] were used to evaluate the performance. 30 sentences (for source and target speakers) are used for training and 10 sentences are used for testing.

6.4.1 Objective Evaluation

For the objective evaluation, the performance is measured via the frame-level Mel-Cepstral Distortion (MCD) [26] between the converted and the target signals.

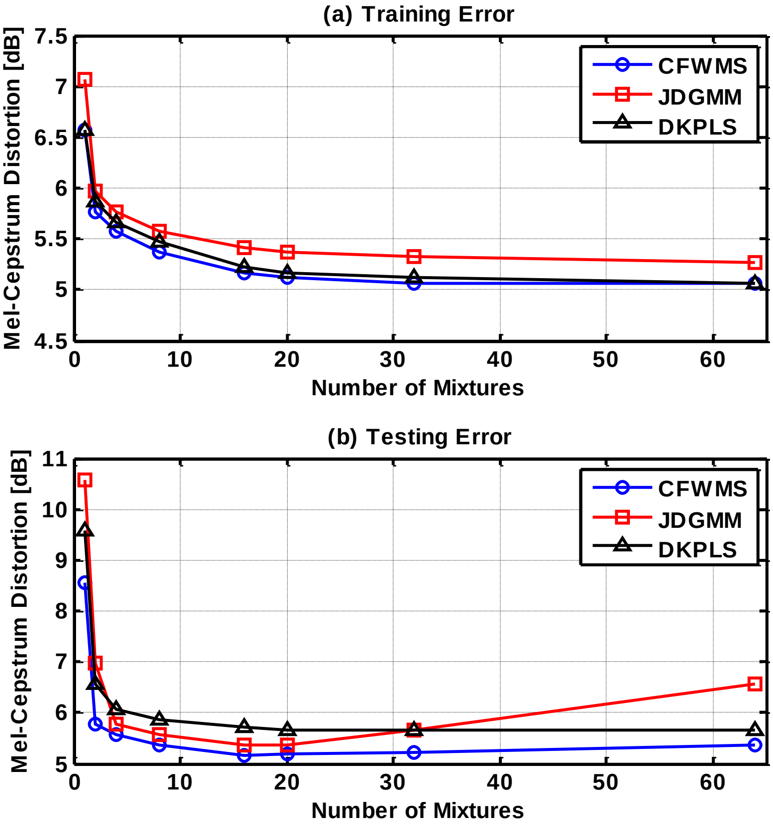


Figure 6-5 MCD between the converted signals and the target signals

Figure 6-5 shows the training error (a) and testing error (b) along with the number of mixtures in GMM. As expected, the training error decreases with the increasing number of mixtures. The increasing trend of the testing error in JD-GMM confirms Kain’s observation [42]. Note that DKPLS can “somehow” prevent over-fitting by reducing the dimension of latent factors [16]. For CFWMS, the testing error is not significantly increased with the number of mixtures. This is because the proposed CTL method regulates the irrational trajectory mapping and FCWMS provides more reasonable predictions, even though CFWMS is developed under the

JD-GMM framework.

6.4.2 Subjective Evaluation

A MOS test was employed for subjective evaluation, in which converted speech signals using 4 methods were played to 5 listeners blindly. Listeners were asked to decide which of the utterances is closer to the target speech signals or higher quality. The identity comparative results are provided in Figure 6-6 (a). It can be observed that the speech signals transformed by CFWMS are most likely to be perceived as it is spoken by the target speaker. The quality results are given in Figure 6-6 (b). Obviously, the voice quality of the speech converted via CFWMS is significantly better than those converted by the other spectral methods.

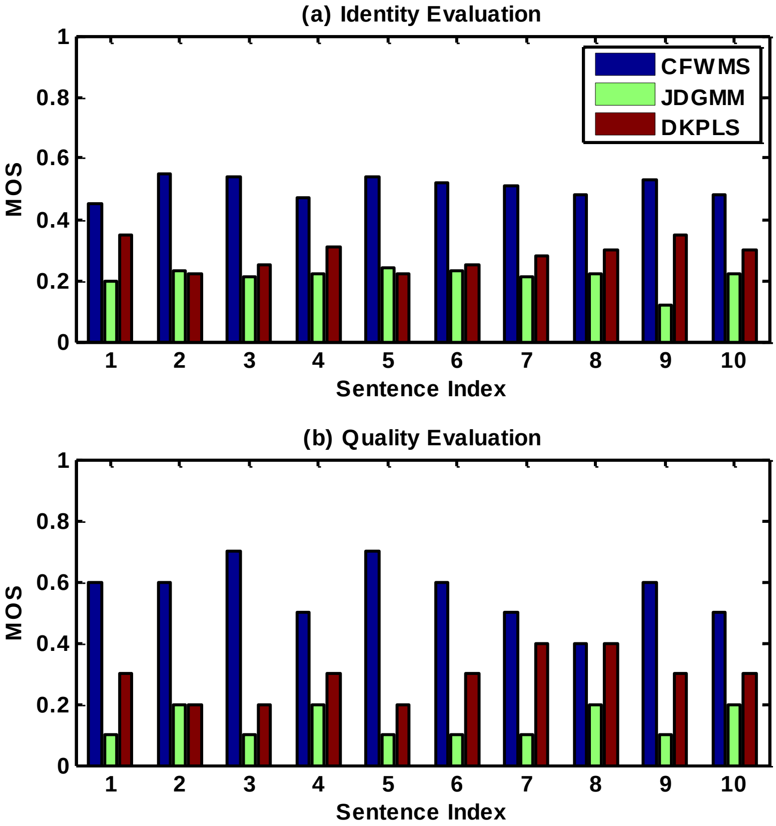


Figure 6-6 Subjective Evaluations

6.5 Summary and Conclusions

This chapter presents Continuous Frequency Warping and Magnitude Scaling (CFWMS), a low-latency (i.e. frame-level), high quality and high-efficiency spectrum mapping solution.

Additionally, a low-latency trajectory limitation algorithm, i.e. Classification Trajectory Limitation (CTL), is presented to further improve the quality of the output speech which is converted using CFWMS. The experimental results show that the proposed approach achieves significant improvements in terms of speech quality and perceptual similarity, by comparing it with a baseline solution (JDGMM) and a state-of-the-art voice conversion solution (DKPLS).

Chapter 7

Summary and Future Work

This chapter reviews and summarises the finished work and the obtained results. Following this, the suggestions for future work are presented. Finally, the conclusion is given.

7.1 Summary

The first goal of this research project is to develop a real-time frequency scale modification algorithm to “colour” the speaker voice with low latency and high quality, which corresponds to the content in Chapter 3. The second goal is to design an efficient pitch and formant modification solution to enable the flexible and real-time voice change application, where the designs are presented in Chapter 4. Notably, the proposed algorithm has been selected and used in the commercialized products. The third goal is to review the existing voice conversion frameworks and solutions, which aims to identify the unsolved issue in voice conversion and give some suggestions to future designs. The related works are presented in Chapter 5. Finally, Chapter 6 achieves the fourth goal which is to design an efficient and high-quality solution for real-time voice conversion application.

Limited by the contract with the commercialization company, the algorithms and implementations of the real-time frequency scale modification and the real-time voice change were not achieved into proceedings or articles. The research work that is isolated from the commercialization company was submitted to the 28th Irish Signals & Systems Conference (ISSC 2017, Killarney, Ireland, 20-21, June 2017), the manuscript was accepted and presented using the title: “ Voice Conversion based on Continuous Frequency Warping and Magnitude Scaling”.

From the theoretical analysis and experimental results in Chapter 3, the developed real-time frequency scale modification achieves the same quality as the conventional solution –

WSOLA but requires much less computing load. The proposed voice change framework and the applications (i.e. Voice Gender Conversion) developed based on this framework achieves remarkable perceptual identity. In addition, the comparative studies in Chapter 4 also show that the proposed frequency warping scheme overweighs other warping solutions (i.e. scaling the LSF or modifying the roots of LPC).

Chapter 6 covers the majority of the state-of-the-art framework/approaches in voice conversion and analyses the characteristics of different frameworks. Based on the issues that are found in the existing approaches, a voice conversion solution, i.e. Continuous Frequency Warping and Magnitude Scaling, is designed and implemented. Both the subjective and the objective test results show that the proposed solution outperforms the baseline solutions (i.e. JD-GMM and DKPLS) in terms of the perceptual identity and quality.

7.2 Future Work

According to the progress to-date, some future works can be considered to further contribute to the voice conversion research. For instance, a significant problem in voice conversion is that frames are not perfectly "**aligned**" during the training stage. Although DTW can provide a reasonable time alignment for source and target speakers, the relationship between formant/pitch of source and target speakers may still be uncorrelated. For example, the word "Hello" can be either an exalting tone or an eliminating tone, if the source speaker and target speaker are saying the same word using different tones, the relationship between pitch becomes uncorrelated. Moreover, according to the author's observations, formant and pitch are not entirely unrelated, which exacerbates the statistical smoothing and consequently degenerates speech quality. Although a frequency warping based algorithm somehow mitigates statistical smoothing, it still suffers from the uncorrelated formants, i.e. causes inaccurate warping functions. To this end, a practical solution is to normalize the formant features by the pitch values, such that the formant can be better aligned.

7.3 Conclusion

This research project aims to develop efficient voice change and voice conversion solutions for real-time applications. Corresponding to the voice change application, a real-time frequency scale modification algorithm and a real-time pitch/formant scale modification framework has been developed. The rest part contributes to the voice conversion application, the related works have been comprehensively reviewed and a low complexity solution is proposed. The designs, details and evaluations are documented in this thesis.

Reference

[1] B. P. Nguyen and M. Akagi, ‘Spectral modification for voice gender conversion using temporal decomposition’, *J. Signal Process.*, 2007.

[2] T. Kinnunen, Z.-Z. Wu, K. A. Lee, F. Sedlak, E. S. Chng, and H. Li, ‘Vulnerability of speaker verification systems against voice conversion spoofing attacks: The case of telephone speech’, in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, 2012, pp. 4401–4404.

[3] ‘Talking Tom – Talking Tom and Friends’. [Online]. Available: https://talkingtomandfriends.com/tom/. [Accessed: 24-Jan-2018].

[4] M. Abe, S. Nakamura, K. Shikano, and H. Kuwabara, ‘Voice conversion through vector quantization’, *J. Acoust. Soc. Jpn. E*, vol. 11, no. 2, pp. 71–76, 1990.

[5] D. Erro, A. Moreno, and A. Bonafonte, ‘Voice conversion based on weighted frequency warping’, *IEEE Trans. Audio Speech Lang. Process.*, vol. 18, no. 5, pp. 922–931, 2010.

[6] ‘Connector Helps Kids Talk to Santa This Year With Cool New App’, *Connector*, 11-Dec-2014. [Online]. Available: http://connector.ie/blog/connector-helps-kids-talk-santa-year-cool-new-app/. [Accessed: 24-Jan-2018].

[7] D. A. Reynolds, ‘An overview of automatic speaker recognition technology’, in *Acoustics, speech, and signal processing (ICASSP), 2002 IEEE international conference on*, 2002, vol. 4, pp. IV–4072.

[8] A. V. McCree and T. P. Barnwell, ‘A mixed excitation LPC vocoder model for low bit rate speech coding’, *IEEE Trans. Speech Audio Process.*, vol. 3, no. 4, pp. 242–250, 1995.

[9] E. Moulines and F. Charpentier, ‘Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones’, *Speech Commun.*, vol. 9, no. 5–6, pp. 453–467, 1990.

[10] D. O’Shaughnessy, ‘Linear predictive coding’, *IEEE Potentials*, vol. 7, no. 1, pp. 29–32, 1988.

[11] H. Kawahara, ‘Speech representation and transformation using adaptive interpolation of weighted spectrum: vocoder revisited’, in *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, 1997, vol. 2, pp. 1303–1306.

[12] Y. Stylianou, T. Dutoit, and J. Schroeter, ‘Diphone Concatenation using a Harmonic plus Noise Model of Speech’, in *Proc. EUROSPEECH*, 1997.

[13] D. Erro, A. Moreno, and A. Bonafonte, ‘Flexible harmonic/stochastic speech synthesis.’, in *SSW*, 2007, pp. 194–199.

[14] J. L. Flanagan and R. M. Golden, ‘Phase vocoder’, *Bell Labs Tech. J.*, vol. 45, no. 9, pp. 1493–1509, 1966.

[15] M. Slaney, *Lyon’s cochlear model*, vol. 13. Apple Computer, Advanced Technology Group, 1988.

[16] E. Helander, H. Silén, T. Virtanen, and M. Gabbouj, ‘Voice conversion using dynamic kernel partial least squares regression’, *IEEE Trans. Audio Speech Lang. Process.*, vol. 20, no. 3, pp. 806–817, 2012.

[17] E. Helander, T. Virtanen, J. Nurminen, and M. Gabbouj, ‘Voice conversion using partial least squares regression’, *IEEE Trans. Audio Speech Lang. Process.*, vol. 18, no. 5, pp. 912–921, 2010.

[18] M. Airaksinen, ‘Analysis/Synthesis Comparison of Vocoders Utilized in Statistical Parametric Speech Synthesis’, Master of Science in Technology, Aalto University, Aalto University, 2012.

[19] J. F. Gemmeke, T. Virtanen, and A. Hurmalainen, ‘Exemplar-based sparse representations for noise robust automatic speech recognition’, *IEEE Trans. Audio Speech Lang. Process.*, vol. 19, no. 7, pp. 2067–2080, 2011.

[20] K. K. Paliwal, ‘On the use of line spectral frequency parameters for speech recognition’, *Digit. Signal Process.*, vol. 2, no. 2, pp. 80–87, 1992.

[21] K. Tokuda, T. Kobayashi, T. Masuko, and S. Imai, ‘Mel-generalized cepstral analysis-a unified approach to speech spectral estimation.’, in *ICSLP*, 1994, vol. 94, pp. 18–22.

[22] A. Bundy and L. Wallen, ‘Dynamic Time Warping’, in *Catalogue of Artificial Intelligence Tools*, Springer, 1984, pp. 32–33.

[23] D. J. Berndt and J. Clifford, ‘Using dynamic time warping to find patterns in time series.’, in *KDD workshop*, 1994, vol. 10, pp. 359–370.

[24] T. Toda, A. W. Black, and K. Tokuda, ‘Voice conversion based on maximum-likelihood estimation of spectral parameter trajectory’, *IEEE Trans. Audio Speech Lang. Process.*, vol. 15, no. 8, pp. 2222–2235, 2007.

[25] C.-H. Wu, C.-C. Hsia, T.-H. Liu, and J.-F. Wang, ‘Voice conversion using duration-embedded bi-HMMs for expressive speech synthesis’, *IEEE Trans. Audio Speech Lang. Process.*, vol. 14, no. 4, pp. 1109–1116, 2006.

[26] R. Kubichek, ‘Mel-cepstral distance measure for objective speech quality assessment’, in *Communications, Computers and Signal Processing, 1993., IEEE Pacific Rim Conference on*, 1993, vol. 1, pp. 125–128.

[27] M. Viswanathan and M. Viswanathan, ‘Measuring speech quality for text-to-speech systems: development and assessment of a modified mean opinion score (MOS) scale’, *Comput. Speech Lang.*, vol. 19, no. 1, pp. 55–83, 2005.

[28] Y. Chen, M. Chu, E. Chang, J. Liu, and R. Liu, ‘Voice conversion with smoothed GMM and MAP adaptation.’, in *INTERSPEECH*, 2003.

[29] ‘[1510.07020] An Efficient Polyphase Filter Based Resampling Method for Unifying the PRFs in SAR Data’. [Online]. Available: https://arxiv.org/abs/1510.07020. [Accessed: 24-Jan-2018].

[30] S. Roucos and A. Wilgus, ‘High quality time-scale modification for speech’, in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP’85.*, 1985, vol. 10, pp. 493–496.

[31] G. John, L. Lori, F. William, F. Jonathan, P. David, and D. Nacy, ‘DARPA, TIMIT Acoustic-Phonetic Continuous Speech Corpus CD-ROM. National Institute of Standards and Technology’. NIST, 1993.

[32] H. Ba, N. Yang, I. Demirkol, and W. Heinzelman, ‘BaNa: A hybrid approach for noise resilient pitch detection’, in *Statistical Signal Processing Workshop (SSP), 2012 IEEE*, 2012, pp. 369–372.

[33] W. Verhelst and M. Roelands, ‘An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech’, in *Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference on*, 1993, vol. 2, pp. 554–557.

[34] B. S. Atal and S. L. Hanauer, ‘Speech analysis and synthesis by linear prediction of the speech wave’, *J. Acoust. Soc. Am.*, vol. 50, no. 2B, pp. 637–655, 1971.

[35] S. K. Mitra and J. F. Kaiser, *Handbook for digital signal processing*. John Wiley & Sons, Inc., 1993.

[36] H. Mizuno and M. Abe, ‘Voice conversion algorithm based on piecewise linear conversion rules of formant frequency and spectrum tilt’, *Speech Commun.*, vol. 16, no. 2, pp. 153–164, 1995.

[37] D. G. Childers and K. Wu, ‘Gender recognition from speech. Part II: Fine analysis’, *J. Acoust. Soc. Am.*, vol. 90, no. 4, pp. 1841–1856, 1991.

[38] S. Nakamura and K. Shikano, ‘Speaker adaptation applied to HMM and neural networks’, in *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on*, 1989, pp. 89–92.

[39] H. Matsumoto and Y. Yamashita, ‘Unsupervised speaker adaptation from short utterances based on a minimized fuzzy objective function’, *J. Acoust. Soc. Jpn. E*, vol. 14, no. 5, pp. 353–361, 1993.

[40] Y. Stylianou, O. Cappé, and E. Moulines, ‘Continuous probabilistic transform for voice conversion’, *IEEE Trans. Speech Audio Process.*, vol. 6, no. 2, pp. 131–142, 1998.

[41] H. G. Sung, ‘Gaussian mixture regression and classification’, PhD Thesis, Rice University, 2004.

[42] A. Kain and M. W. Macon, ‘Spectral voice conversion for text-to-speech synthesis’, in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, 1998, vol. 1, pp. 285–288.

[43] P. Song, Y. Q. Bao, L. Zhao, and C. R. Zou, ‘Voice conversion using support vector regression’, *Electron. Lett.*, vol. 47, no. 18, pp. 1045–1046, 2011.

[44] D. Sundermann and H. Ney, ‘VTLN-based voice conversion’, in *Signal Processing and Information Technology, 2003. ISSPIT 2003. Proceedings of the 3rd IEEE International Symposium on*, 2003, pp. 556–559.

[45] E. L. Lawler and D. E. Wood, ‘Branch-and-bound methods: A survey’, *Oper. Res.*, vol. 14, no. 4, pp. 699–719, 1966.

[46] D. Erro, E. Navas, and I. Hernaez, ‘Parametric voice conversion based on bilinear frequency warping plus amplitude scaling’, *IEEE Trans. Audio Speech Lang. Process.*, vol. 21, no. 3, pp. 556–566, 2013.

[47] D. D. Lee and H. S. Seung, ‘Algorithms for non-negative matrix factorization’, in *Advances in neural information processing systems*, 2001, pp. 556–562.

[48] C.-J. Lin, ‘On the convergence of multiplicative update algorithms for nonnegative matrix factorization’, *IEEE Trans. Neural Netw.*, vol. 18, no. 6, pp. 1589–1596, 2007.

[49] A. Cichocki, H. Lee, Y.-D. Kim, and S. Choi, ‘Non-negative matrix factorization with α-divergence’, *Pattern Recognit. Lett.*, vol. 29, no. 9, pp. 1433–1440, 2008.

[50] A. Cichocki, R. Zdunek, and S. Amari, ‘Csiszar’s divergences for non-negative matrix factorization: Family of new algorithms’, in *International Conference on Independent Component Analysis and Signal Separation*, 2006, pp. 32–39.

[51] C. Févotte and J. Idier, ‘Algorithms for nonnegative matrix factorization with the β-divergence’, *Neural Comput.*, vol. 23, no. 9, pp. 2421–2456, 2011.

[52] C. Févotte, N. Bertin, and J.-L. Durrieu, ‘Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis’, *Neural Comput.*, vol. 21, no. 3, pp. 793–830, 2009.

[53] E. Esser, M. Moller, S. Osher, G. Sapiro, and J. Xin, ‘A convex model for nonnegative matrix factorization and dimensionality reduction on physical space’, *IEEE Trans. Image Process.*, vol. 21, no. 7, pp. 3239–3252, 2012.

[54] P. O. Hoyer, ‘Non-negative matrix factorization with sparseness constraints’, *J. Mach. Learn. Res.*, vol. 5, no. Nov, pp. 1457–1469, 2004.

[55] Z. Wu, T. Virtanen, T. Kinnunen, E. S. Chng, and H. Li, ‘Exemplar-based voice conversion using non-negative spectrogram deconvolution’, in *Eighth ISCA Workshop on Speech Synthesis*, 2013.

[56] S. Boyd, ‘Alternating direction method of multipliers’, in *Talk at NIPS Workshop on Optimization and Machine Learning*, 2011.

[57] R. Aihara, T. Takiguchi, and Y. Ariki, ‘Activity-mapping non-negative matrix factorization for exemplar-based voice conversion’, in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, 2015, pp. 4899–4903.

[58] R. Aihara, T. Nakashika, T. Takiguchi, and Y. Ariki, ‘Voice conversion based on Non-negative matrix factorization using phoneme-categorized dictionary’, in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 7894–7898.

[59] S. Desai, A. W. Black, B. Yegnanarayana, and K. Prahallad, ‘Spectral mapping using artificial neural networks for voice conversion’, *IEEE Trans. Audio Speech Lang. Process.*, vol. 18, no. 5, pp. 954–964, 2010.

[60] R. J. Schalkoff, *Artificial neural networks*, vol. 1. McGraw-Hill New York, 1997.

[61] R. Salakhutdinov, A. Mnih, and G. Hinton, ‘Restricted Boltzmann machines for collaborative filtering’, in *Proceedings of the 24th international conference on Machine learning*, 2007, pp. 791–798.

[62] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, ‘Greedy layer-wise training of deep networks’, in *Advances in neural information processing systems*, 2007, pp. 153–160.

[63] B. A. Pearlmutter, ‘Gradient calculations for dynamic recurrent neural networks: A survey’, *IEEE Trans. Neural Netw.*, vol. 6, no. 5, pp. 1212–1228, 1995.

[64] I. Sutskever, G. E. Hinton, and G. W. Taylor, ‘The recurrent temporal restricted boltzmann machine’, in *Advances in Neural Information Processing Systems*, 2009, pp. 1601–1608.

[65] L. Sun, S. Kang, K. Li, and H. Meng, ‘Voice conversion using deep bidirectional long short-term memory based recurrent neural networks’, in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, 2015, pp. 4869–4873.

[66] M. Ramos, ‘Voice Conversion with Deep Learning’, Master of Science Degree, TECNICO LISBOA, TECNICO LISBOA, 2016.

[67] E. Godoy, O. Rosec, and T. Chonavel, ‘Voice conversion using dynamic frequency warping with amplitude scaling, for parallel or nonparallel corpora’, *IEEE Trans. Audio Speech Lang. Process.*, vol. 20, no. 4, pp. 1313–1323, 2012.

[68] M. Morise, F. Yokomori, and K. Ozawa, ‘WORLD: a vocoder-based high-quality speech synthesis system for real-time applications’, *IEICE Trans. Inf. Syst.*, vol. 99, no. 7, pp. 1877–1884, 2016.

[69] R. Adams, ‘Active queue management: a survey’, *IEEE Commun. Surv. Tutor.*, vol. 15, no. 3, pp. 1425–1476, 2013.

[70] C. A. Ratanamahatana and E. Keogh, ‘Everything you know about dynamic time warping is wrong’, in *Third Workshop on Mining Temporal and Sequential Data*, 2004.

[71] S. Skiena, ‘Dijkstra’s algorithm’, *Implement. Discrete Math. Comb. Graph Theory Math. Read. MA Addison-Wesley*, pp. 225–227, 1990.

[72] L. Muda, M. Begam, and I. Elamvazuthi, ‘Voice recognition algorithms using mel frequency cepstral coefficient (MFCC) and dynamic time warping (DTW) techniques’, *ArXiv Prepr. ArXiv10034083*, 2010.

[73] Y. Stylianou, ‘Voice transformation: a survey’, in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, 2009, pp. 3585–3588.

[74] J. Kominek, A. W. Black, and V. Ver, ‘CMU ARCTIC databases for speech synthesis’, 2003.