NTNU

Norwegian University of
Science and Technology

Voice Transformation based on Gaussian mixture models

Terje Gundersen

Master of Science in Communication Technology
Submission date: June 2010
Supervisor: Torbjørn Svendsen, IET

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

Problem Description

Voice Transformation refers to the process of modifying a voice from a speaker so that it sounds like it has been produced by a different speaker. The main speaker-specific characteristics are determined by the short time frequency spectrum and the fundamental frequency of the vocal cords. This work will be based on a project that implemented a system for filter transformation based on the Gaussian mixture model. Variants and improvements of the basic system of the spectral envelope transformation will be studied, and a selected solution should be implemented and evaluated. Methods for excitation transformation will also be studied and experimentally evaluated.

Assignment given: 18. January 2010
Supervisor: Torbjørn Svendsen, IET

ABSTRACT

In this thesis, a probabilistic model for transforming a voice to sound like another specific voice is tested. The model is fully automatic and only requires some 100 training sentences from both speakers with the same acoustic content. The classical source-filter decomposition allows prosodic and spectral transformation to be performed independently. The transformations are based on a Gaussian mixture model and a transformation function suggested by Y. Stylianou [1]. Feature vectors of the same content from the source and target speaker, aligned in time by dynamic time warping, are fitted to a GMM. The short time spectra, represented as cepstral coefficients and derived from LPC [2], and the pitch periods, represented as fundamental frequency estimated from the RAPT algorithm [3], are transformed with the same probabilistic transformation function.

Several techniques of spectrum and pitch transformation were assessed in addition to some novel smoothing techniques of the fundamental frequency contour. The pitch transform was implemented on the excitation signal from the inverse LP filtering by time domain PSOLA. The transformed spectrum parameters were used in the synthesis filter with the transformed excitation as input to yield the transformed voice.

A listening test was performed with the best setup from objective tests and the results indicate that it is possible to recognise the transformed voice as the target speaker with a 72 % probability. However, the synthesised voice was affected by a muffling effect due to incorrect frequency transformation and the prosody sounded somewhat robotic.

PREFACE

The following report is the result of my Master's thesis work for the Norwegian University of Science and Technology (NTNU) at the Department of Electronics and Telecommunications. This thesis is a continuation of a project I worked on last semester on the same topic. In the project I got familiarised with voice transformation and implemented some fundamental algorithms used in my Master's thesis. Every code line was made from scratch, which entailed many hours of debugging in both semesters. Even though, it has been inspiring and joyful to work on such a new and experimental research topic.

I would like to express my special thanks to my advisor, Torbjørn Svendsen, for great talks and guidance in our weekly meetings.

Terje Gundersen
June 2010

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

CC ..... Cepstral Coefficients
DTW ..... Dynamic Time Warping
EM ..... Expectation Maximisation
FT ..... Filter Transform
GMM ..... Gaussian Mixture Model
LP ..... Linear Prediction
LPC ..... Linear Prediction Coding
NCCF ..... Normalised Cross-Correlation Function
NCD ..... Normalised Cepstral Distance
NPD ..... Normalised Pitch Distance
PSOLA ..... Pitch Synchronous Overlap-and-Add
PT ..... Pitch Transform
RAPT ..... Robust Algorithm for Pitch Tracking
RC ..... Reflection Coefficients
TTS ..... Text-To-Tpeech
VQ ..... Vector Quantisation

CHAPTER 1
INTRODUCTION

1.1 Definition

Voice transformation, or conversion, refers to the various modifications that can be applied to the sound produced by a person, either speaking or singing [5]. More specific, in this thesis voice transformation refers to the process of modifying the speech signal from a person so that it sounds like a certain other person has produced it.

1.2 Motivation

Users of a text-to-speech (TTS) system often want to have a choice between several synthetic voices, if they are not satisfied with the default voice. To create a new voice for a TTS system from scratch requires a lot of work and money. If it was possible to create only one synthetic voice and then transform that voice to other voices, with only a few minutes of training data, the system provider could produce any number of voices with ease.

Another application of voice transformation is speech prostheses. If a person has limited ability of producing speech or is going to loose his ability to speak, he can record some utterances of his voice and create a speech synthesis with his own voice. A similar application is synthesising the voice of celebrities, e.g. movie stars, who has passed away or lost their voice.

1.3 Modification Methods

The production of speech can be modelled as a source of glottal excitation passing through the vocal tract which acts as a filter, called the source-filter model [6]. The filter is a linear time-varying filter that can be assumed to be stationary for short

time intervals, e.g. 10 ms. The signal can be separated into a source and a filter by conducting a linear prediction (LP) analysis and subtracting the predicted signal from the real signal, yielding a filter described as LP coefficients and the source as the excitation from inverse LP filtering.

One of the earliest implementations of speech transformation by M. Abe utilised vector quantization (VQ) for mapping spectral properties with discrete source and target classes [7]. By training codebooks to represent a mapping of corresponding feature vectors from the source and target speaker, the transformation is simply to look up in the codebook and map the feature vectors. In contrast, Y. Stylianou *et al.* suggested a soft-decision model by using a Gaussian mixture model (GMM) to compute a continuous probability density mapping for spectral transformation [1]. Traditional pitch transformation used a simple fundamental frequency scaling while A. Kain *et al.* suggested using the GMM model for pitch transformation as well [8] which is state-of-the-art today.

Source modifications can modify the pace, pitch and intensity of a voice. Time- and pitch-scale modifications can be done with the pitch synchronous overlap and add (PSOLA) procedure. PSOLA splits the signals into two pitch periods long windowed parts and re-synthesises the signal with transformed pitch information with the overlap-add procedure, by duplicating or deleting frames to alter the pace of the voice and adjusting the spacing of windows to alter the pitch. Filter modifications can modify the magnitude spectrum of the frequency response of the vocal tract, which carries information of speaker individuality, by mapping filter coefficients [9, 10]. The phase is not altered in this frequency transform approach, which is unfortunate for the quality of the transformation.

For the purpose of this thesis, voice transformation is a combination of pitch and filter modifications to transform the voice characteristics of a source speaker to match a specific target speaker.

1.4 Structure and Goal of This Thesis

The goal of this thesis is to assess different implementations methods to achieve the best possible voice transformation, i.e. synthesis of a new natural voice which sounds like the target voice. Some necessary pre- and post-processing like endpoint detection, voicing detection, fundamental frequency smoothing are implemented as well as some experimental techniques; pre-emphasis, exclusion of cepstral energy coefficient c_0 in spectral transformation and two novel smoothing techniques.

This thesis is structured as follows. The background theory and concepts for carrying out the experiments are explained in Chapter 2. A more detailed explanation of how the system can be implemented is given in Chapter 3. Objective and subjective results are presented with comments in Chapter 4. The final Chapter 5 summarises the system implementation and suggests future work.

CHAPTER 2

THEORY

The basic idea of the voice transformation is to make a linear prediction (LP) analysis of the signal and perform an inverse filtering to yield a source-filter separation, where the excitation represents the source and the LP coefficients represent the filter. The frequency spectra of the voice can be transformed by altering the filter coefficients to match the new voice, and the pitch can be transformed by transforming the fundamental frequency in small speech segments and altering the pitch by the PSOLA technique on the excitation signal. The transformed excitation and LP coefficients are used in the inverse filter to synthesis the transformed voice, as depicted in Figure 2.1.

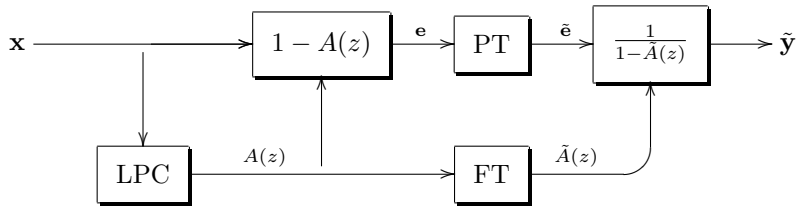


Figure 2.1: Voice transformation system with source-filter separation where a pitch transform (PT) is performed on the excitation and a filter transform (FT) modifies the filter parameters.

The challenge in this procedure is to find global context-independent transformation functions explained in Section 2.4. The transformation function is trained with known corresponding source and target vectors to output the correct target vector with a new source vector as input. In the training process the source and target speaker signals are time-aligned with dynamic time warping, Section 2.3. The corresponding vectors are represented as a Gaussian mixture model (GMM), Section 2.2.

2.1 Signal Representation

Representing a signal in the time domain requires a huge amount of storage, e.g. representing 10 ms with a sampling frequency of 8 kHz yields 80 values, and a 5 seconds sentence would need approximately 40 000 values. Linear predictive coding, Section 2.1.1, is a fast and simple signal representation that only requires e.g. 10 values to represent a signal segment of 10 ms, as opposed to 80. The LPC coefficients, $A(z)$, are the coefficients in an IIR filter which approximates the signal. By applying an inverse filtering of such a filter we get a source-filter separation where the source is the prediction error, as shown in Figure 2.1.

LPC has a number of equivalent representations with different properties used in different tasks. The representations used in the filter transformation are presented in Section 2.1.2 and 2.1.3. Fundamental frequency is used in the pitch transformation, explained in Section 2.1.4.

2.1.1 Linear Predictive Coding

Linear predictive coding, LPC, was first applied to speech signals by B. Atal *et al.* in 1968 [2]. LPC approximates the signal as a p-th order all-pole filter by predicting the current sample as a linear combination of its previous p samples [11]

$$\tilde{x}(n) = \sum_{k=1}^{p} \alpha_k x(n-k) \tag{2.1}$$

The prediction error by this representation is

$$\begin{aligned} e(n) =& x(n) - \tilde{x}(n) \\ =& x(n) - \sum_{k=1}^{p} \alpha_k x(n-k) \end{aligned} \tag{2.2}$$

The prediction coefficients α_k can be estimated by the minimum mean squared error technique, which estimates the coefficients that minimise the total prediction error E.

$$\begin{aligned} E =& \sum_n e^2(n) \\ =& \sum_n \left(x(n) - \sum_{k=1}^{p} \alpha_k x(n-k) \right)^2 \end{aligned} \tag{2.3}$$

The coefficients can be obtained by taking the derivative of (2.3) with respect to α_i, and equating to 0. By setting $\partial E / \partial \alpha_i = 0$ we obtain

$$\sum_n x(n-i)x(n) = \sum_k \alpha_k \sum_n x(n-i)x(n-k), \qquad 1 \leq i \leq p \tag{2.4}$$

By introducing autocorrelation defined as

$$R(k) = \sum_n x(n)x(n+k) \tag{2.5}$$

Equation (2.4) becomes

$$R(i) = \sum_{k=1}^{p} \alpha_k R(|i-k|), \qquad 1 \leq i \leq p \tag{2.6}$$

Which yields a set of p linear equations with p unknown parameters, which easily can be solved by e.g. the Levinson-Durbin algorithm [11, 12].

$$\begin{bmatrix} R(0) & R(1) & \dots & R(p-1) \\ R(1) & R(0) & \dots & R(p-2) \\ \vdots & \vdots & \ddots & \vdots \\ R(p-1) & R(p-2) & \dots & R(0) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_p \end{bmatrix} = \begin{bmatrix} R_1 \\ R_2 \\ \vdots \\ R_p \end{bmatrix} \tag{2.7}$$

2.1.2 Cepstrum

The cepstrum was first introduced by B. P. Bogert *et al.* [13] in 1963. The cepstrum of a signal is a homomorphic transformation to the *quefrency* domain [6]. Cepstral coefficients (CC) have the property of low intra-frame correlation.

The complex cepstrum is defined as the inverse Fourier transform of the log Fourier transform

$$c(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \ln X(e^{j\omega}) e^{j\omega n} d\omega \tag{2.8}$$

However, the complex cepstrum $c(n)$ parameters can be converted from LP coefficients α_n by the following recursion:

$$c(n) = \begin{cases} \alpha_n + \sum_{k=1}^{n-1} \left(\frac{k}{n}\right) c(k)\alpha_{n-k}, & 0 < n \leq p \\ \sum_{k=n-p}^{n-1} \left(\frac{k}{n}\right) c(k)\alpha_{n-k}, & n > p \end{cases} \tag{2.9}$$

where p is the number of LP coefficients. The cepstrum representation is defined with infinite number of coefficients, anything less would be an approximation. The usual choice of accuracy is in the vicinity of 20 coefficients.

The cepstrum parameters can be converted back from LPC by rearranging the equation,

$$c_n - \sum_{k=1}^{n-1} \left(\frac{k}{n}\right) c_k \alpha_{n-k} = \begin{cases} \alpha_n, & 0 < n \leq p \\ 0, & n > p \end{cases} \tag{2.10}$$

However, the recursion assumes that the cepstrum parameters represent an all-pole model with poles inside the unit circle, minimum-phase, which might not be the case. Which means that if the cepstrum parameters have been altered, for instance by a transformation function, the resulting LP coefficients might be unstable and the recursion will not yield 0 for $n > p$.

Another approach to map CC to LP is to use equation (2.8). By taking the Fourier transform of the complex cepstrum and the inverse logarithm, we get the

Fourier transform of the signal. The magnitude of the Fourier transform yields the power spectrum of the signal. From the power spectrum we can get the autocorrelation R_{xx} of the signal by applying the inverse Fourier transform. From the

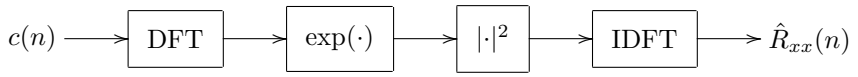


Figure 2.2: Conversion of complex cepstrum to autocorrelation.

autocorrelation representation we have the same situation as in (2.7), which means that we can get the LPC parameters by the Levinson-Durbin algorithm [12]. The number of DFT coefficients can be chosen arbitrary to get the desired resolution.

2.1.3 Reflection Coefficients

Reflection coefficients (RC) is another equivalent representation of LPC which could easily be checked for stability. The reflection coefficients have a dynamic range of $[-1, 1]$ if they are stable.

The reflection coefficients k can be obtained from the linear prediction coefficients α by the following recursion

$$\begin{aligned} k_i &= \alpha_i^i, \quad i = p, \ldots, 1 \\ \alpha_j^{i-1} &= \frac{\alpha_j^i + \alpha_j^i \alpha_{i-j}^i}{1 - k_i^2}, \quad 1 \leq j < i \end{aligned} \tag{2.11}$$

where $\alpha_i^p = \alpha_i$.

The coefficients can be converted back to LP by [6]

$$\begin{aligned} \alpha_i^i &= k_i, \quad i = 1, \ldots, p \\ \alpha_j^i &= \alpha_j^{i-1} - k_i \alpha_{i-j}^{i-1}, \quad 1 \leq j < i \end{aligned} \tag{2.12}$$

where $\alpha_i = \alpha_i^p$.

2.1.4 Pitch Detection

Pitch is the perceptual "tone" of the voice and cannot directly be measured from the signal. However, the pitch period can be simplified to label the smallest true period of the signal. The inverse of the smallest period of a small analysis interval yields the fundamental frequency $f_0(t)$. If there is no distinct period in the analysis window, it is detected as unvoiced.

There are many methods of prediction the pitch and detecting voicing in a speech signal; in fact, it is written books on the subject [14]. They all, however, use some correlation algorithm, with some pre and post processing, to find the smallest distinctive period in the signal.

D. Talkin suggested a method using cross-correlation, in contrast to autocorrelation as used in many other techniques, called robust algorithm for pitch tracking

(RAPT) [3]. The cross-correlation in this matter is much the same as autocorrelation, but considers a correlation window of more than just the current analysis frame. Let x be a windowed signal segment, the cross-correlation χ of x is defined as

$$\chi_{i,k} = \sum_{j=m}^{m+n-1} x_j x_{j+k}, \quad k = [0, K-1], m = iz, i = [0, M-1] \tag{2.13}$$

where i is the frame index for M frames, z is the number of sample to advance for each frame, k is the lag index and n is the number of samples in the correlation window.

The RAPT algorithm uses two versions of the speech signal, one at the original sample rate and one with significantly reduced sample rate. The reduced sample rate version is passed into a normalised cross-correlation function (NCCF) and the local maximum correlation distance is recorded. The results of the first iteration are used as a initialisation of the same procedure with the original signal, where the local maxima of the NCCF are restricted to be in the vicinity of the results of the first iteration. The second iteration yields candidate f_0 values that are used in a dynamic programming algorithm which utilises certain properties of the NCCF and voiced speech. Such properties are *a*) if there are more than one local maximum, the shortest period is the correct choice, *b*) f_0 is slowly varying over time so local maxima in adjacent frames should be located at comparable lags, *c*) if the change in adjacent frames are significant it should be a doubling or a halving of lags, *d*) a change in voicing in adjacent frames will occur at low frequencies, *e*) the short-time spectra of voiced and unvoiced frames are significantly different and *f*) amplitude tend to increase at the onset of voicing and to decrease at offset [3].

To find the pitch peak of an analysis frame is simply to search for the maximum value in the vicinity of where it is supposed to be according to the fundamental frequency results.

2.2 Gaussian Mixture Model

The GMM is a classical parametric model used in many pattern recognition techniques [1]. Because of the wide variety of speech realisations, representing all of them would require enormous complexity of both training data and the transformation function. A cluster classification would be a better choice. The first implementation of voice transformation by M. Abe [7] used vector quantisation (VQ) [15] where a large number of vectors are clustered into a smaller number of classes represented by its mean value. A vector is then represented by the mean of the closest class, called hard decision, which could result in large errors.

The Gaussian mixture model is a soft decision classifier where each class has a Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. The GMM assumes that the probability distribution of the observed parameters takes the following parametric form

$$p(\mathbf{z}) = \sum_{i=1}^{m} \alpha_i \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \tag{2.14}$$

where m is the number of mixture models and $\mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ denotes the p-dimensional normal distribution [16] with the mean vector $\boldsymbol{\mu}_i$ and covariance matrix $\boldsymbol{\Sigma}_i$ defined by

$$\mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^p |\boldsymbol{\Sigma}|}} \exp\left[-\frac{1}{2}(\mathbf{z}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{z}-\boldsymbol{\mu})\right] \tag{2.15}$$

The weighting factor α_i in (2.14) is the prior probability of class i with constraints $\sum_{i=1}^{M} \alpha_i = 1$ and $\alpha_i \geq 0$. The input vectors $\mathbf{z}_i$ are assumed to be independent [1].

The conditional probability that a given observation vector $\mathbf{z}$ belongs to the component C_i of the GMM, is given by Bayes' rule [16]

$$P(C_i|\mathbf{z}) = \frac{\alpha_i \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\sum_{j=1}^{m} \alpha_j \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{2.16}$$

The parameters $\boldsymbol{\alpha}, \boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ can be estimated with the expectation maximisation (EM) algorithm. The EM algorithm is an iterative algorithm for unsupervised learning in which component information is unavailable. The EM algorithm tries to find the parameters in the GMM that give the maximum likelihood of the observed data $\mathbf{Z}$. The initial values for the parameters of the GMM can be computed from the VQ representation of the vectors by the *k-means* algorithm [17]. The next step is to find the expectation of the log-likelihood of the latent data given the current estimate of the parameters. A new estimate of the parameters is found by maximising the expected log-likelihood and the process is repeated until convergence [6].

2.3 Dynamic Time Warping

Feature comparison of two signals gives more sense if they are aligned in time. If the only difference in two signals is a time shift, they will not appear as equal if they are not aligned in time. Dynamic time warping (DTW) is a dynamic programming concept to warp two sequences such that the difference between the signals is minimised. The signals are segmented into small parts that are compared and given a similarity-score. For LP parameters a good similarity-score is the *Itakura distance* [18],

Definition 2.1. Given two vectors $\mathbf{x}$ and $\mathbf{y}$ of LP coefficients from a LPC analysis of speech signals $\mathbf{a}$ and $\mathbf{b}$, and the autocorrelation of $\mathbf{a}$, R_{aa}. The Itakura distance between the two LP vectors is

$$\begin{aligned} d_I(\mathbf{x}, \mathbf{y}) &= \log\left(\frac{\mathbf{y}^T \mathcal{T}(R_{aa})\mathbf{y}}{\mathbf{x}^T \mathcal{T}(R_{aa})\mathbf{x}}\right) \\ &= \log\left(\mathbf{y}^T \mathcal{T}(R_{aa})\mathbf{y}\right), \quad x_0 = 1 \end{aligned} \tag{2.17}$$

where $\mathcal{T}$ is the Toeplitz-matrix defined as

$$\mathcal{T}(R) = \begin{bmatrix} R_0 & R_1 & \cdots & R_{n-1} \\ R_1 & R_0 & \cdots & R_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ R_{n-1} & R_{n-2} & \cdots & R_0 \end{bmatrix} \tag{2.18}$$

The Itakura distance is not a true metric however, since it is not symmetric [19]. The denominator of (2.17) is the residual, or the power of the error, from the LP analysis of $\mathbf{a}$ with the source coefficients $\mathbf{x}$. For normalized coefficients, $x_0 = 1$, the power is 1. The numerator is the power of the residual from the LP analysis of $\mathbf{a}$ with the coefficients $\mathbf{y}$.

The two signals which are to be aligned in time are represented as sequences of feature vectors, $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]^T$ and $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_M]^T$.

The minimum accumulated distance between the two signals can be found by choosing the shortest path from a finite set of possible paths. The problem can be solved as a recursion [6]

$$\begin{aligned} \mathbf{D}_{min}(i,j) = \min\big[&\mathbf{D}_{min}(i-1,j-1) + \alpha d_I(\mathbf{x}_i, \mathbf{y}_j), \\ &\mathbf{D}_{min}(i-1,j) + \beta d_I(\mathbf{x}_i, \mathbf{y}_j), \\ &\mathbf{D}_{min}(i,j-1) + \beta d_I(\mathbf{x}_i, \mathbf{y}_j), \\ &\mathbf{D}_{min}(i-1,j-2) + \gamma d_I(\mathbf{x}_i, \mathbf{y}_j)\big] \end{aligned} \tag{2.19}$$

where $\mathbf{x}_i$ and $\mathbf{y}_j$ denote the LP coefficients of segment i and j from the two signals, respectively. The problem is broken down into simpler subproblems, hence dynamic programming.

The possible paths are shown in Figure 2.3. The weights α, β and γ, called local

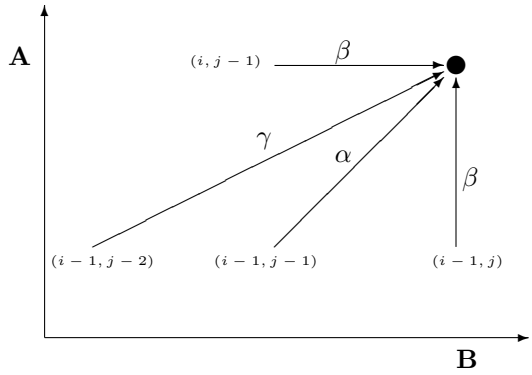


Figure 2.3: Possible paths in DTW algorithm with local constraints α, β and γ.

constraints, are optional and could have a significant effect on the outcome. For instance, the β weights can be assigned a larger value to avoid multiple skipping

or repetition of frames. Global constraints can be applied in addition to the local constraints, which limits the resulting path by borders to guarantee a maximum modification of the signals.

Remark 2.1. There is no option for the path $\mathbf{D}_{min}(i-2, j-1)$ to $\mathbf{D}_{min}(i, j)$ to ensure that all segments of the source speaker are used. Moreover, if a source vector is mapped to two or more target vectors, only one of the target vectors, or the mean of the target vectors, can be used to represent the mapping. This is important to achieve a one-to-one mapping of the source vectors as they are used as input to the transformation function. For the same reason, the global constraints cannot allow open ends for the source vector since this will in practise mean stripping off the edges of the signal.

The indices in the resulting shortest path are used to duplicate and/or delete frames in the target signal to get a time-aligned version of the target matrix, which is the best match to the LP matrix of the source speaker.

2.4 Transformation Function

Given a set of time-aligned LP vectors of the source and target speaker, $\mathbf{x}$ and $\mathbf{y}$, and a GMM fitted to a joint vector $\mathbf{z} = [\mathbf{x}, \mathbf{y}]$, as depicted in Figure 2.4, we want

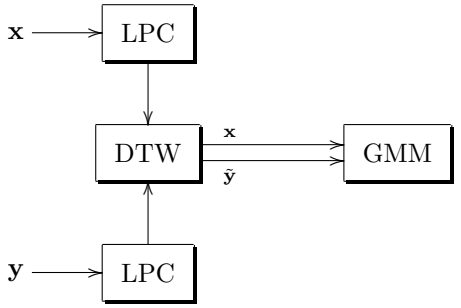


Figure 2.4: Training of GMM used in full covariance matrix transformation.

to define a transformation function which yields the most likely target vector given a new source vector

$$\mathcal{F}(\mathbf{x}) = E[\mathbf{y}|\mathbf{x}] = \tilde{\mathbf{y}} \tag{2.20}$$

With the GMM fitted to the joint vector $\mathbf{z} = [\mathbf{x}, \mathbf{y}]$, the covariance matrix will be

$$\mathbf{\Sigma}^{zz} = \begin{bmatrix} \mathbf{\Sigma}^{xx} & \mathbf{\Sigma}^{xy} \\ \mathbf{\Sigma}^{yx} & \mathbf{\Sigma}^{yy} \end{bmatrix} \tag{2.21}$$

If we assume that the source and target vectors are jointly Gaussian, the likelihood of $\mathbf{y}$ given $\mathbf{x}$ for a single component GMM can be expressed as [20]

$$E\left[\mathbf{y}|\mathbf{x}\right] = \boldsymbol{\mu}^y + \mathbf{\Sigma}^{yx}\left(\mathbf{\Sigma}^{xx}\right)^{-1}\left(\mathbf{x} - \boldsymbol{\mu}^x\right) \tag{2.22}$$

where E denotes the expectation, $\boldsymbol{\mu}$ is the mean vector

$$\boldsymbol{\mu}^y = E[\mathbf{y}] \tag{2.23}$$

and $\boldsymbol{\Sigma}$ is the cross-covariance matrix

$$\boldsymbol{\Sigma}^{yx} = E\left[(\mathbf{y} - \boldsymbol{\mu}^y)(\mathbf{x} - \boldsymbol{\mu}^x)^T\right] \tag{2.24}$$

Y. Stylianou suggested using this formula for a multicomponent GMM yielding the transformation function.

Definition 2.2. Given a Gaussian mixture model trained with a joint vector of time aligned feature vectors $\mathbf{z} = [\mathbf{x}, \mathbf{y}]$ and a new feature vector $\mathbf{x}$ not contained in the training set of the GMM. The most likely corresponding feature vector $\mathbf{y}$ is [21]

$$\begin{aligned}\mathcal{F}(\mathbf{x}) =& E[\mathbf{y}|\mathbf{x}] \\ =& \sum_{i=1}^{m} P(C_i|\mathbf{x}) \left[\boldsymbol{\mu}_i^y + \boldsymbol{\Sigma}_i^{yx} \left(\boldsymbol{\Sigma}_i^{xx}\right)^{-1} \left(\mathbf{x} - \boldsymbol{\mu}_i^x\right)\right]\end{aligned} \tag{2.25}$$

where the likelihoods are weighted with the conditional probabilities of $\mathbf{x}$ belonging to component C_i.

2.4.1 Diagonal Covariance Matrix

To decrease the computational complexity, the covariance matrix in the GMM can be constrained to be diagonal, if it is a good assumption with the signal representation in question. Cepstral coefficients is such a representation. However, by constraining the covariance matrix to be diagonal we have to estimate the cross-covariance σ^{yx}.

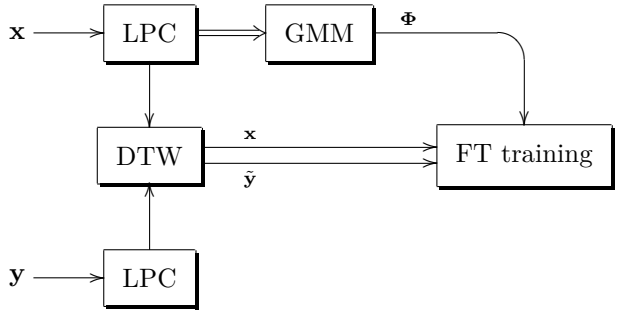


Figure 2.5: Training of filter transformation parameters with diagonal covariance matrix.

If we train the GMM with $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$, the class probabilities $P(\mathbf{C}_i|\mathbf{x}_t)$ and the associated means $\boldsymbol{\mu}_i^x$ and variances of the source vector $\boldsymbol{\sigma}_i^{xx}$ can be derived from the GMM, while the means of the target vectors $\boldsymbol{\mu}_i^y$ and cross-covariance matrices $\boldsymbol{\sigma}_i^{yx}$ are unknown and need to be estimated.

Assuming diagonal covariance matrices, the computation of (2.25) can be done for each individual vector element k

$$\mathcal{F}(x_t^{(k)}) = \sum_{i=1}^{m} P(C_i|\mathbf{x}_t)[\mu_i^{y^{(k)}} + \sigma_i^{yx^{(k)}}(x_t^{(k)} - \mu_i^{x^{(k)}})/\sigma_i^{xx^{(k)}}] \tag{2.26}$$

By estimating the transformation output to be equal to a training set of target vectors $\mathbf{Y}$, (2.26) can be expressed as a set of overdetermined linear equations that can be used to least-square optimise the unknown variables [1]

$$\begin{aligned}\mathcal{F}(\mathbf{x}^{(k)}) = \mathbf{y}^{(k)} =& \mathbf{P}\boldsymbol{\mu}^{y^{(k)}} + \mathbf{D}^{(k)}\boldsymbol{\sigma}^{yx^{(k)}} \\ =& \begin{bmatrix}\mathbf{P} & \vdots & \mathbf{D}^{(k)}\end{bmatrix} \begin{bmatrix}\boldsymbol{\mu}^{y^{(k)}} \\ \cdots \\ \boldsymbol{\sigma}^{yx^{(k)}}\end{bmatrix}\end{aligned} \tag{2.27}$$

where $\mathbf{y}^{(k)}$ has dimensions $n \times 1$, and $\mathbf{P}$ is a $n \times m$ matrix with the posterior probabilities

$$\mathbf{P} = \begin{bmatrix} P(C_1|\mathbf{x}_1) & \cdots & P(C_m|\mathbf{x}_1) \\ P(C_1|\mathbf{x}_2) & \cdots & P(C_m|\mathbf{x}_2) \\ \vdots & & \vdots \\ P(C_1|\mathbf{x}_n) & \cdots & P(C_m|\mathbf{x}_n) \end{bmatrix} \tag{2.28}$$

$\mathbf{D}^{(k)}$ is a $n \times m$ matrix defined as

$$\mathbf{D}^{(k)} = \begin{bmatrix} P(C_1|\mathbf{x}_1)(x_1^{(k)} - \mu_1^{x^{(k)}})/\sigma_1^{xx^{(k)}} & \cdots & P(C_m|\mathbf{x}_1)(x_1^{(k)} - \mu_m^{x^{(k)}})/\sigma_m^{xx^{(k)}} \\ P(C_1|\mathbf{x}_2)(x_2^{(k)} - \mu_1^{x^{(k)}})/\sigma_1^{xx^{(k)}} & \cdots & P(C_m|\mathbf{x}_2)(x_2^{(k)} - \mu_m^{x^{(k)}})/\sigma_m^{xx^{(k)}} \\ \vdots & & \vdots \\ P(C_1|\mathbf{x}_n)(x_n^{(k)} - \mu_1^{x^{(k)}})/\sigma_1^{xx^{(k)}} & \cdots & P(C_m|\mathbf{x}_n)(x_n^{(k)} - \mu_m^{x^{(k)}})/\sigma_m^{xx^{(k)}} \end{bmatrix} \tag{2.29}$$

The two unknown matrices $\boldsymbol{\mu}^{y^{(k)}}$ and $\boldsymbol{\sigma}^{yx^{(k)}}$ will both have dimensions $m \times 1$

$$\boldsymbol{\mu}^{y^{(k)}} = \left[\mu_1^{y^{(k)}}, \mu_2^{y^{(k)}}, \ldots, \mu_m^{y^{(k)}}\right]^T \tag{2.30}$$

$$\boldsymbol{\sigma}^{yx^{(k)}} = \left[\sigma_1^{yx^{(k)}}, \sigma_2^{yx^{(k)}}, \ldots, \sigma_m^{yx^{(k)}}\right]^T \tag{2.31}$$

We want to find the set $\left[\boldsymbol{\mu}^{y^{(k)}}, \boldsymbol{\sigma}^{yx^{(k)}}\right]^T$ that minimises the total squared conversion error

$$\epsilon = \sum_t \|y_t^{(k)} - \mathcal{F}(x_t^{(k)})\|^2 \tag{2.32}$$

This least-square problem can be solved by the normal equations [22]

$$\left(\begin{bmatrix}\mathbf{P}^T\\ \cdots \\ \mathbf{D}^{(k)^T}\end{bmatrix}\begin{bmatrix}\mathbf{P} & \vdots & \mathbf{D}^{(k)}\end{bmatrix}\right)\begin{bmatrix}\boldsymbol{\mu}^{y^{(k)}}\\ \cdots \\ \boldsymbol{\sigma}^{yx^{(k)}}\end{bmatrix} = \begin{bmatrix}\mathbf{P}^T\\ \cdots \\ \mathbf{D}^{(k)^T}\end{bmatrix}\mathbf{y}^{(k)}$$

$$\begin{bmatrix}\boldsymbol{\mu}^{y^{(k)}}\\ \cdots \\ \boldsymbol{\sigma}^{yx^{(k)}}\end{bmatrix} = \left(\begin{bmatrix}\mathbf{P}^T\\ \cdots \\ \mathbf{D}^{(k)^T}\end{bmatrix}\begin{bmatrix}\mathbf{P} & \vdots & \mathbf{D}^{(k)}\end{bmatrix}\right)^{-1}\begin{bmatrix}\mathbf{P}^T\\ \cdots \\ \mathbf{D}^{(k)^T}\end{bmatrix}\mathbf{y}^{(k)} \tag{2.33}$$

2.5 Synthesis

Given a transformed set of pitch and frequency feature parameters, we would like to synthesise the new transformed voice. As depicted in Figure 2.1 the pitch transformation (PT) is applied on the excitation from the inverse LP filtering and the transformed excitation is then passed through a pitch-synchronous LP synthesis filter with the transformed filter coefficients. The PT consists of transforming the fundamental frequency with the presented transformation function and change the pitch in the excitation signal by time-domain PSOLA.

2.5.1 PSOLA

Pitch synchronous overlap-and-add (PSOLA) procedure can modify the pace and/or the pitch of a speech signal. Time domain PSOLA splits the signals into two pitch periods long windowed parts and re-synthesises the parts with the overlap-add procedure, by duplicating or deleting frames to alter the pace of the voice and adjusting the spacing of windows to alter the pitch [6]. The goal is to alter the pitch of the signal without affecting the spectral characteristics while this is taken care of by the LP parameters in the synthesise filter.

The unvoiced parts of the excitation signal are approximately white noise and need no modification. The voiced parts can be represented as a function of pitch cycles

$$e(n) = \sum_{i=-\infty}^{\infty} e_i(n - t_s(i)) \tag{2.34}$$

where $t_s(i)$ are the epochs of the source speaker signal and the pitch period is

$$P_s(i) = t_s(i) - t_s(i-1) \tag{2.35}$$

One pitch cycle can be represented by applying a Hamming window, w, of length two pitch periods, to the signal. The term "overlap and add" comes from the use of overlapping windowed segments.

$$e_i(n) = w_i(n)e(n) \tag{2.36}$$

If we replace the source timing instances, t_s, with the ones from the target, t_t, we get the desired pitch contour.

$$\tilde{e}(n) = \sum_{i=-\infty}^{\infty} e_i(n - t_t(i)) \tag{2.37}$$

The same procedure can be used on unvoiced speech as well, if the epoch distance is set uniformly and smaller than 10 ms [23].

2.6 Robustness

To increase the performance of the transformation certain pre-processing techniques can be applied to emphasise signal characteristics important for the transformation.

2.6.1 Endpoint Detection

Excluding non-informative information, e.g. silence in the beginning and end of a sentence, could enhance the transformation because it excludes unwanted training data. If the signal-to-noise ratio is high it is a trivial task. However, if there is some background noise, it is not that easy to detect the boundaries of the speech utterance. The basic approach is to derive the energy of the signal and choose a boundary for how much energy there is in the utterance and clip the signal where the energy crosses this boundary. But if there is a peak in the noise level when the speaker is silent, the peak could be recognised as the start of a sentence.

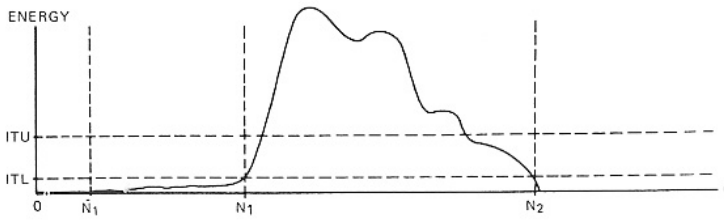


Figure 2.6: Endpoint detection with utterance boundary ITU and silence boundary ITL [4].

A more robust method of endpoint detection, proposed by [4], is to locate a point where there a high probability for an utterance, and backtrace from this point to the silence boundary, illustrated in Figure 2.6. A supplementary technique for avoiding peak-noise error is the use of short-time energy. Instead of calculating the energy in each sample separately, we can introduce a memory component to smooth out the energy contour.

$$E(n) = \sum_{j=0}^{n} \alpha^j x^2(n - j) \tag{2.38}$$

where $\alpha < 1$ decides how much information from the previous samples to include in the current energy value. Equation (2.38) can be simplified to

$$\begin{aligned} E(n+1) =& \sum_{j=0}^{n+1} \alpha^j x^2(n+1-j) \\ =& x^2(n+1) + \sum_{j=1}^{n+1} \alpha^j x^2(n+1-j) \\ =& x^2(n+1) + \sum_{k=0}^{n} \alpha^{k+1} x^2(n-k) \\ =& x^2(n+1) + \alpha E(n), \quad E(0) = x^2(0) \end{aligned} \tag{2.39}$$

By estimating the energy of the signal by (2.39), the endpoints can be detected by choosing a utterance boundary, of e.g. $ITU = 0.02$, and a lower silence boundary, e.g. $ITL = 0.002$. The algorithm will seek for the first sample which exceeds ITU and backtrace from this sample to the first sample with lower energy than ITL.

2.6.2 Pre-emphasis

Some implementations of frequency transformation lack a good transformation of high frequencies [24]. A temporary increase of the energy in high frequencies during the transformation operations could cope with this problem. This can be achieved by passing the speech signals through a pre-emphasis filter in the pre-processing and do the inverse operation before synthesis. The pre-emphasis filter is a simple one state IIR filter

$$H(z) = \frac{1}{1 - \alpha z^{-1}} \tag{2.40}$$

where $0 < \alpha < 1$. The frequency response with $\alpha = 0.97$ is depicted in Figure 2.7.

According to [24], pre-emphasis yields a significant improvement for sampling frequencies greater than 16 kHz for LSF feature vectors. Furthermore [24] claims that pre-emphasis relaxes the time-aligned criteria for codebook matching with the transformation input vector. However, whether it provides an increase in the overall quality of the frequency transformation with cepstral coefficients, is not thoroughly tested.

2.7 Objective Result Metrics

The best quality metric of a voice transformation is the subjective impression from a listening test. However, it is dependent of the listeners and it could be very time consuming. Objective metrics could therefore be useful in the development phase. They are often easy to measure and give a consistent result if used properly.

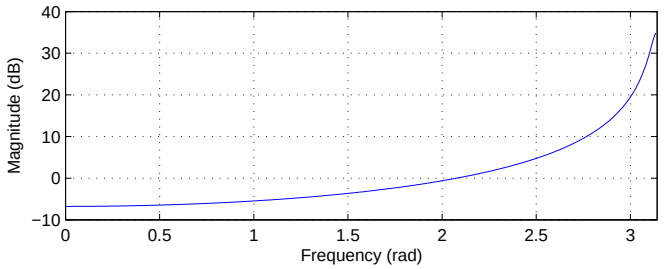


Figure 2.7: Frequency response of a pre-emphasis filter with feedback coefficient $\alpha = 0.97$.

2.7.1 Frequency Metrics

Itakura distance (2.17) is a distance metric to measure the difference between LP vectors. Cepstral distance is another metric useful in measuring the quality of the frequency transformation from the cepstral domain.

Definition 2.3. Let $\tilde{\mathbf{c}}$ be a vector of transformed cepstral coefficients and $\mathbf{c}^t$ the vector of target coefficients. The cepstral distance between the two vectors is

$$\epsilon(\tilde{\mathbf{c}}, \mathbf{c}^t) = \sum_{i=1}^{n-1} (c_i^t - \tilde{c}_i)^2 \tag{2.41}$$

where the energy coefficient c_0 is not considered.

The Normalised Cepstral Distortion (NCD) of the transformed vector with source vector $\mathbf{c}^s$, is

$$\epsilon(\tilde{\mathbf{c}}, \mathbf{c}^t, \mathbf{c}^s) = \frac{\sum_{i=1}^{n-1} (c_i^t - \tilde{c}_i)^2}{\sum_{j=1}^{n-1} (c_j^t - c_j^s)^2} \tag{2.42}$$

Which measures the relative improvement of the transformation [25].

While NCD measures the improvement of the transform relative to untransformed source vectors in the cepstral domain, the L_2 metric operates in the frequency domain and measures the deviation of the transformed spectrum to the target spectrum.

Definition 2.4. Given the power spectrum $S(\omega)$ of a signal

$$S(\omega) = \frac{\sigma^2}{|1 - \sum_i \alpha_i e^{-j\omega_i}|^2} \tag{2.43}$$

let $\mathbf{a}$ and $\mathbf{b}$ be target and the transformed signals, respectively. The log RMS distortion is

$$d_{rms}(\mathbf{a}, \mathbf{b}) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left| \ln \frac{S_a(\omega)}{S_b(\omega)} \right|^2 d\omega \tag{2.44}$$

the L_2 metric is defined as [26]

$$L_2(\mathbf{a}, \mathbf{b}) = 4.343\sqrt{d_{rms}(\mathbf{a}, \mathbf{b})} \quad [dB] \tag{2.45}$$

The L_2 metric is a true metric in the sense that is satisfy the triangle inequality in addition to being symmetric and positive definite [19].

Remark 2.2. The L_2 metric is the root mean square (RMS) log spectral measure in decibel. The factor 4.343 comes from the conversion from natural logarithm to decibel

$$10\log(x) = 10\frac{\ln(x)}{\ln(10)} = 4.343\ln(x) \tag{2.46}$$

The L_2 metric normalised with the source spectrum is

$$\begin{aligned} Nd_{rms}(\mathbf{a}, \mathbf{b}, \mathbf{c}) =& \frac{\int |\ln S_a(\omega) - \ln S_b(\omega)|^2 d\omega}{\int |\ln S_a(\omega) - \ln S_c(\omega)|^2 d\omega} \\ NL_2(\mathbf{a}, \mathbf{b}, \mathbf{c}) =& 4.343\sqrt{Nd_{rms}(\mathbf{a}, \mathbf{b}, \mathbf{c})} \quad [dB] \end{aligned} \tag{2.47}$$

2.7.2 Pitch Metrics

The quality of a pitch transformation can be measured by controlling the fundamental frequency f_0 of each pitch period. The fundamental frequency is the frequency of pitch marks pm, i.e.

$$f_0(i) = \frac{1}{pm(i) - pm(i-1)} \tag{2.48}$$

For a sentence of several pitch periods the average error in fundamental frequency is

$$\mu_\epsilon = \frac{1}{N}\sum_{i=0}^{N-1} f_0^t(i) - \tilde{f}_0(i) \tag{2.49}$$

where f_0^t is the target fundamental frequency and $\tilde{f}_0$ is the transformed fundamental frequency.

A metric for relative improvement from the source pitch f_0^s used in this paper is the Normalised Pitch Distortion (NPD) [25]

$$NPD = \sqrt{\frac{\sum_{i=0}^{N-1}\left(f_0^t(i) - \tilde{f}_0(i)\right)^2}{\sum_{i=0}^{N-1}\left(f_0^t(i) - f_0^s(i)\right)^2}} \tag{2.50}$$

CHAPTER 3

IMPLEMENTATION

This chapter covers how the theory in Chapter 2 can be implemented and the different versions of the system that were tested. The important steps in the process are *a*) the pre-processing with endpoint-detection and unvoiced detection, *b*) the pitch synchronous LPC analysis and the dynamic time warping for training data alignment, *c*) the frequency conversion in the cepstrum domain to utilise diagonal covariance properties, *d*) the pitch transform via f_0 and *e*) PSOLA and pitch synchronous synthesis filtering.

3.1 Analysis

The recorded speech samples used in the implementation included parts of silence especially in the beginning and end of the recordings. Each recording consisted of one sentence, which means there were negligible parts of silence during the utterances. The endpoints of each sentence were detected by the algorithm described in Section 2.6.1.

The stripped recordings with a sampling frequency of 8k kHz were used in a pitch-synchronous linear prediction analysis of order 10, i.e. the signal was segmented by two-pitch period length Hamming windows, centered around a pitch pulse. According to [27], there is a negligible difference between the autocorrelation and the covariance method for natural speech with window length of 2 pitch periods. The autocorrelation method was implemented, as described in Section 2.1.1. The pitch markings and the fundamental frequency f_0, as well as detection of unvoiced frames, were read from a database of earlier studies at NTNU [28], which used the RAPT algorithm outlined in Section 2.1.4.

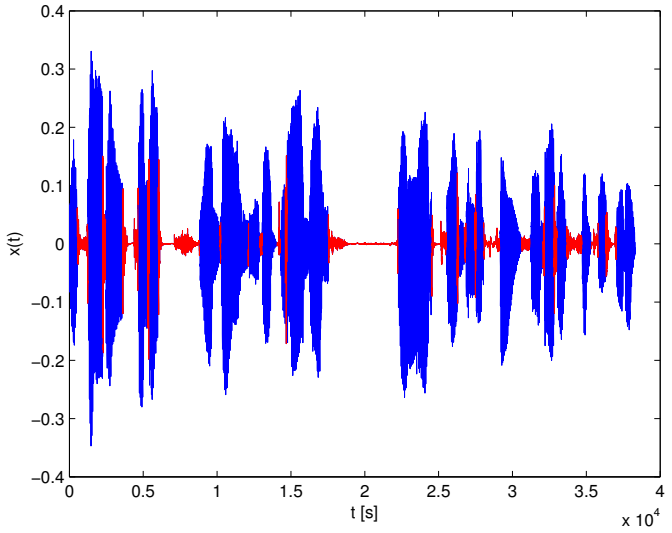


Figure 3.1: Voiced/unvoiced detection. Red colour indicates segments of a speech signal detected as unvoiced.

3.2 Dynamic Time Warping

The LPC frames marked as voiced were used in the dynamic time warping algorithm, Section 2.3, to find the optimal mapping between the source and target LP frames. The local constraints for the dynamic time warping were optimised for the Itakura distance (2.17), which resulted in the weighting factors $\alpha = 1$, $\beta = 1$ and $\gamma = 1$.

The global constraints were chosen to prevent stripping of the source LPC vectors and to allow open ends, of size 20 frames, for the target vectors as shown in Figure 3.2. Without open ends, the first and last source frame will always be mapped to the first and last, respectively, target frame. Moreover, the global constraints decreased the possible paths by an outer border to relax the computational complexity and constrain a maximum modification of the target signal. The greyscale depicts the cumulated Itakura distance where a light shade illustrates a close match. The red line in Figure 3.2 depicts the resulting shortest path in the distance matrix. If the red path has a horizontal line along its path, i.e. one source vector is mapped to more than one target vector; only one of the target vectors was used in the mapping. By this configuration the source vectors are unaltered and the target vectors are duplicated or discarded to best match the source vectors.

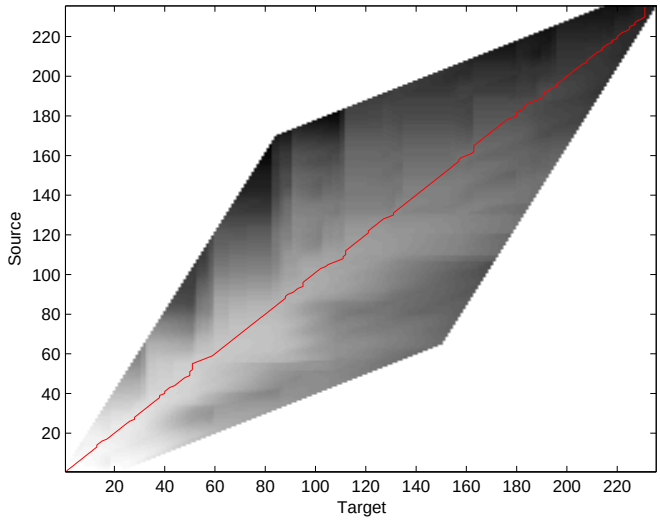


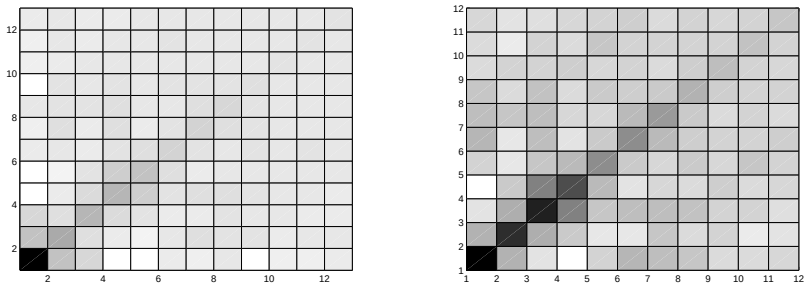
Figure 3.2: Time alignment by dynamic time warping and the Itakura distance. The grey-scale indicates the cumulative Itakura distance and the red line denotes the resulting shortest path.

3.3 Filter Transformation

By transforming the filter we alter the magnitude spectrum of the frequency response of the vocal tract. The filter representation was cepstral coefficients, which have the property of low intra-frame correlation. This was exploited by approximating the covariance matrices $\mathbf{\Sigma}^{xx}$ and $\mathbf{\Sigma}^{yx}$ to diagonal matrices. The Gaussian mixture model was trained with cepstrum vectors of order 13 from the source speaker, $\mathbf{X}_{cc}$. 128 mixtures were used in the GMM initialised by vector quantization classification computed by the k-means algorithm [17]. The covariance matrices had a lower bound of 10^{-5} to ensure convergence. With the GMM fitted to the source speaker and a set of time aligned source and target cepstral vectors, the cross-covariance matrices $\mathbf{\Sigma}^{yx}$ and the mean value vectors $\boldsymbol{\mu}^{y}$ were estimated as described in Section 2.4.1.

The GMM fitted to the source, and the estimated $\mathbf{\Sigma}^{yx}$ and $\boldsymbol{\mu}^{y}$, were used as input to the transformation function as well as a set of cepstrum vectors from the source speaker, not included in the training set. The converted $\tilde{\mathbf{Y}}_{cc}$ vectors were checked for stability in the reflection coefficients domain. To get reflection coefficients from cepstrum by the recursion (2.11), the cepstrum vectors were first converted to 13 order LP by (2.10). The coefficients that were not stable were mirrored to be inside the unit circle. Mirroring means reducing the amplitude, in a polar coordinate sense, to be inside the unit circle without altering the angular frequency [11].

The LP order in the analysis was less than the order of CC in the transformation. Since the transformation is a probabilistic modification of coefficients, it is not very likely that the normal recursion for CC to LP conversion (2.10) would yield 0 for the excessive coefficients. Instead cepstral coefficients were converted to power spectrum to get the LP coefficients via autocorrelation, Figure 2.2.



(a) Full covariance matrix

(b) Covariance matrix without c_0

Figure 3.3: The impact on the covariance matrix by excluding the cepstral energy coefficient c_0. A dark square indicates a high correlation.

Four different configurations were tested. The straightforward configuration with all cepstrum coefficients and no special pre-processing was tested and used as a baseline. Pre-emphasis, Section 2.6.2, was implemented with a filter coefficient $\alpha = 0.97$ and applied on the speech signals in the very first phase of the process and the inverse operation was applied after the synthesis. According to [24], pre-emphasis could enhance transformation of high frequencies. The third setup excluded the first cepstrum coefficient c_0, which is related to frame energy, from the transformation as suggested by [1] and the fourth setup was a combination of the latter two. By excluding the energy coefficient c_0, the approximation of diagonal covariance matrices is less crude, as depicted in Figure 3.3. A dark square indicates a high correlation.

3.4 Pitch Transformation

The easiest way of transforming the pitch is to scale the fundamental frequency such that the mean value match the mean of the target f_0, but the short-time error, or standard deviation of the error, could be huge. The goal of the GMM transformation is to do better than that.

Given a GMM fitted to a vector $\mathbf{z}$, the transformation function has the property of finding remaining elements of a $\mathbf{z}$ vector given a new incomplete vector. Four such $\mathbf{z}$ vectors were tested to find the corresponding f_0 of the target speaker to any given frame, namely $\mathbf{z} = [\mathbf{y}_{cc}, f_0^y]$, $\mathbf{z} = [\mathbf{x}_{cc}, f_0^y]$, $\mathbf{z} = [\mathbf{x}_{cc}, \mathbf{y}_{cc}, f_0^y]$ and $\mathbf{z} = [f_0^x, f_0^y]$, where x represents the source speaker and y the target speaker.

The pitch was transformed with the transformation function (2.25) with a 64 mixture, full-matrix GMM, fitted to a joint vector of target fundamental frequency f_0 and some corresponding feature vector available as input in the transformation. Only voiced frames were used in the training and transformation, where the first cepstral coefficient $\mathbf{c}_0$, which is related to the frame energy, was excluded.

For the training vector $\mathbf{z} = [\mathbf{y}_{cc}, f_0^y]$ suggested by [29], the fundamental frequency has to be scaled. The dynamic range of the fundamental frequency is in the range $[50, 500]$ Hz. It is advantageous to scale the f_0 parameters to values in the neighbourhood of the cepstrum coefficients since they are used in a joint vector. This can be achieved by a log-normalisation [29].

$$f_{log} = \log\left(f_0/\bar{f}_0\right) \tag{3.1}$$

where $\bar{f}_0$ is the average f_0 of all voiced frames.

For the training vector $\mathbf{z} = [\mathbf{y}_{cc}, f_0^y]$, the input to the transformation function were the transformed spectral parameters $\tilde{\mathbf{y}}_{cc}$ and the output were the log-normalised f_0.

$$\begin{aligned}\tilde{f}_0 =& \mathcal{F}(\tilde{\mathbf{y}}_{cc}) \\ =& \sum_{i=1}^{m} P(C_i|\tilde{\mathbf{y}}_{cc}) \left[\boldsymbol{\mu}_i^{f_0} + \boldsymbol{\Sigma}_i^{f_0 y} \left(\boldsymbol{\Sigma}_i^{yy}\right)^{-1} \left(\tilde{\mathbf{y}}_{cc} - \boldsymbol{\mu}_i^{y}\right)\right]\end{aligned} \tag{3.2}$$

The final f_0 value was derived from the de-normalisation of the transformation output. The same procedure was done for all versions of the $\mathbf{z}$ vector.

The unvoiced parts of the signal were not transformed. However, they were assigned a uniform f_0 value, equal to the average target f_0, used in PSOLA and pitch synchronous synthesis.

3.4.1 Smoothing techniques

To prevent an unnatural fluctuation in the f_0 contour, several smoothing techniques were tested.

A good and simple smoothing technique is the moving average filter which simply computes the current value as a average over the last M samples [11].

$$\bar{f}_0(n) = \frac{1}{M} \sum_{k=0}^{M-1} f_0(n-k) \tag{3.3}$$

which can be implemented as a recursion

$$\begin{aligned}\bar{f}_0(n) =& \frac{1}{M} \sum_{k=0}^{M-1} f_0(n-k-1) + \frac{1}{M}\left[f_0(n) - f_0(n-M)\right] \\ =& \bar{f}_0(n-1) + \frac{1}{M}\left[f_0(n) - f_0(n-M)\right]\end{aligned} \tag{3.4}$$

The moving average smoothing is in fact a lowpass FIR filter as depicted in Figure 3.4. The moving average filter is optimal in removing random noise, while

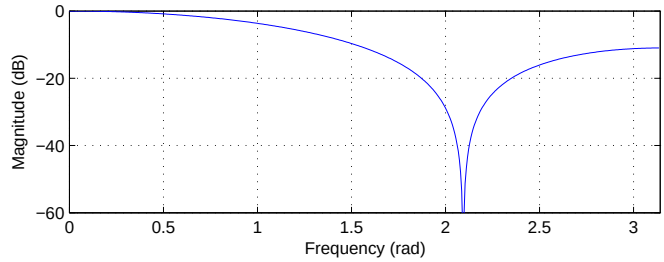


Figure 3.4: Frequency response of a moving average filter with $M = 3$.

retaining a sharp and steep response. It is not, however, a very good lowpass filter due to its slow roll-off and bad stopband attenuation [11]. The impact of

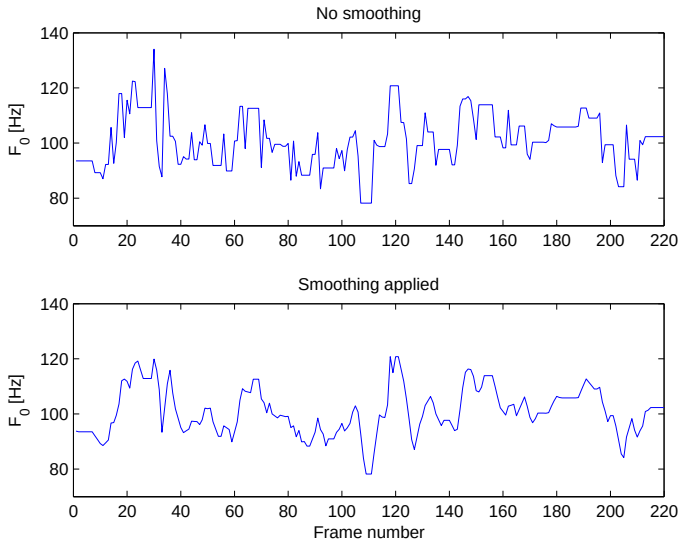


Figure 3.5: A typical effect of smoothing the f_0 contour.

f_0-smoothing is depicted in Figure 3.5.

An alternative to the moving average smoothing is to average over several transformation of the same f_0, henceforth referred to as *modified moving average*. If the GMM is trained with a joint vector of the previous and next f_0 value in addition to the current f_0,

$$\mathbf{z}_t = [\mathbf{y}_{cc,t}, f_{0,t}(n-1), f_{0,t}(n), f_{0,t}(n+1)] \tag{3.5}$$

The output of the transformation would yield a vector of three f_0 values. Three consecutive transformation would include the same f_0 value and were used to com-

pute an average transformation

$$\bar{f}_{0,t} = \frac{1}{3}\left(f_{0,t-1}(n+1) + f_{0,t}(n) + f_{0,t+1}(n-1)\right) \tag{3.6}$$

A third smoothing technique was tested as well. By including three consecutive f_0 values in the transformation, the gradient in the f_0 contour is available

$$\Delta f_{0,t} = \frac{1}{2}\left(f_{0,t}(n+1) - f_{0,t}(n-1)\right) \tag{3.7}$$

The gradient was utilised to control the change of consecutive transformations by the fact that the current transformation should be close to

$$f_{0,t} = f_{0,t-1} + \Delta f_{0,t-1} \tag{3.8}$$

The Δ values could therefore be used to compute limits of the transformation to prevent over-fluctuating contour, henceforth called *delta-limit smoothing*. The limits were set to be

$$\begin{aligned} B_U =& f_{0,t-1} + 2\Delta \\ B_L =& f_{0,t-1} - \frac{\Delta}{2} \end{aligned} \tag{3.9}$$

Remark 3.1. The lower limit is stricter than the upper to allow more deviation in the "correct" direction.

The GMM model used in the transformation treats the input vectors as independent of each other. A model that utilises temporal dependency information might work better with the modified moving average smoothing and the delta-limit smoothing. All three smoothing techniques, and the straightforward f_0 transformation, were tested. There was no transformation of unvoiced parts of the excitation signal.

3.5 Synthesis of Transformed Feature Parameters

The LP synthesis was done pitch-synchronously the same way as the analysis, only with new pitch labels and new LP coefficients for voiced frames. The transformed fundamental frequency $\tilde{f}_0$ was used to find the new pitch markings. Each frame of LP coefficient was mapped to its corresponding $f_0(n)$, hence the corresponding pitch label can then be found by the cumulative sum of $1/\tilde{f}_0(n)$, and multiplying by the sampling frequency to get the label in samples.

$$\begin{aligned} \tilde{p}(n) =& F_s \sum_{i=1}^{n} \frac{1}{\tilde{f}_0(i)} \\ =& \tilde{p}(n-1) + \frac{F_s}{\tilde{f}_0(n)} \end{aligned} \tag{3.10}$$

The new pitch labels were used in the PSOLA routine, Section 2.5.1, to alter the pitch of the utterances to be transformed. Pitch cycles were not discarded or duplicated which means that time-scale modifications were not implemented. This is mainly because such mapping information is not available with the presented implementation of DTW. The pitch modifications were implemented on the excitation signal from the inverse LP filtering to create a new excitation with a transformed pitch. The transformed excitation was then used as input to the LP synthesis filter with the transformed LP coefficients.

$$\tilde{x}(n) = \frac{1}{1 - \sum_{k=1}^{p-1} \tilde{\alpha}_k \tilde{e}(n-k)} \tag{3.11}$$

The last memory states from the synthesis of the current frame were used as initial states in the next frame to get a smooth frame concatenation.

CHAPTER 4

RESULTS

4.1 Fundamental Frequency Transformation

The smoothing techniques for the frequency transform were tested in a controlled environment with the target cepstrum as input, not included in the training set however. The transformation used a 64 mixture GMM with full covariance matrices and was trained with 100 sentences where the unvoiced frames were discarded, yielding a total of 33 000 training vectors. The results from the different smoothing techniques are presented in Table 4.1. The source f_0^s used in the NPD metric is scaled to have the same mean value as the target, values greater than 1 means that the source pitch is closer to the target pitch than the transformed pitch. The standard deviation of the error by using a scaled source pitch is 23.38 Hz.

Table 4.1: f_0 transformation with target cepstrum vectors as input for 10 sentences.

Method	μ_{error}	σ_{error}	*NPD*
No smoothing	-0.75 Hz	20.31 Hz	0.8686
Moving average	-0.77 Hz	19.71 Hz	0.8429
Modified m-avg	-1.58 Hz	19.19 Hz	0.8209
Delta limit	-1.71 Hz	19.93 Hz	0.8524

The results are comparable to the findings of T. En-Najjary *et al.* [29], and indicate that there is a good correlation between cepstral coefficients and fundamental frequency. Given that the target CCs are available, the fundamental frequency transformation works better than simple scaling of the source pitch. "Improved moving average" has the smallest standard deviation, but the mean error is not as good as the "moving average" or the "no smoothing" setups. Since the mean

error could be set to zero by a scaling, the standard deviation and NPD are the most important metrics. The mean f_0 varies a lot from one sentence to another in the original speech signal. So even if a scaling would yield zero overall mean error, it will in most cases be large errors on a sentence-by-sentence basis. Another drawback by this transformation is that the transformed pitch is more monotone than the original speech. The standard deviation in the $f_0(n)$ sequence of 10 sentences of the transformed f_0 is approximately 13 Hz while in the original speech it is about 25 Hz.

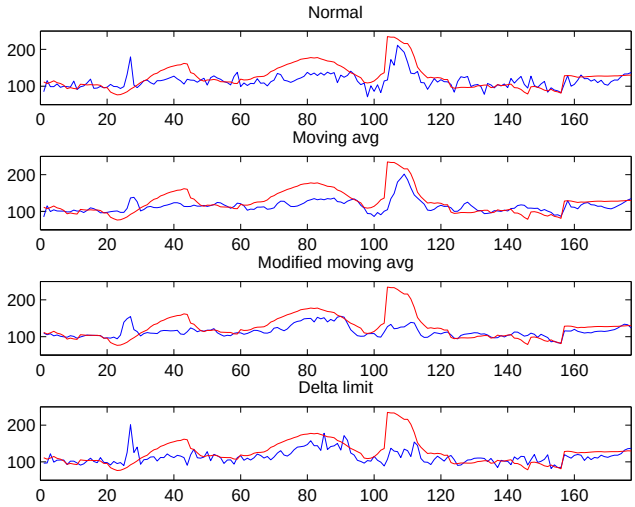


Figure 4.1: Typical contour of f_0 transformation with different smoothing techniques and target Y_{cc} as input. Target f_0 contour is plotted in red.

Figure 4.1 depicts the transformed $f_0(n)$ for a test sentence. The transform without smoothing does not appear to be too bad from Table 4.1, but Figure 4.1 reveals a fluctuating f_0, which is not very natural. The smoothing techniques produce a f_0 contour that is similar to the correct contour, and in this particular case the moving average with $M = 3$ seems to be the best choice.

In a real scenario the available input to the f_0 transform mentioned above, is the transformed $\tilde{\mathbf{Y}}_{cc}$ vectors. This introduces another source of error and the transformation precision is degraded as shown in Table 4.2. Again, the modified moving average had the lowest standard deviation. The table shows an average result from a transformation of 10 test sentences where all the different setups had a NPD higher than 1, hence, they performed worse than a scaling of the source pitch.

The modified moving average is a moving average filter over three independent transformation of one $f_0(t)$. But is not certain that it is transformations of the

Table 4.2: f_0 transformation error, with transformed cepstral vector as input, for 10 test sentences.

Method	μ_e	σ_e	*NPD*
No smoothing	-0.05 Hz	26.29 Hz	1.1246
Moving average	-0.08 Hz	24.57 Hz	1.0511
Modified m-avg	1.18 Hz	23.59 Hz	1.0092
Delta limit	1.25 Hz	24.56 Hz	1.0505

same $f_0(t)$, since the value from the previous transformation $f_{0,t-1(n+1)}$ is the f_0 that should be the next after the current $f_{0,t}(n)$. When the transformation input is independent transformed cepstral vectors, it is not given that consecutive input vectors would appear as consecutive in a real speech segment of the target speaker, hence the three f_0 values in the average operation might correspond to quite different pitch periods. Even though modified moving average was the best smoothing technique in the presented test, a model that utilises temporal information might perform better than GMM for pitch transformation and modified moving average smoothing.

Since there is a correlation between the source cepstrum and the target cepstrum as well as the target cepstrum and the target fundamental frequency, a transformation of target f_0 with source cepstrum as input is a worthy alternative. A comparison of transformation with source cepstrum $\mathbf{x}_{cc}$ and transformed cepstrum $\tilde{\mathbf{y}}_{cc}$ as input is shown in Table 4.3

Table 4.3: Fundamental frequency transform with source cepstrum as input.

Method	μ_e	σ_e	*NPD*
Transformed cepstrum	-0.05 Hz	26.29 Hz	1.1246
Source cepstrum	-2.93 Hz	22.63 Hz	0.9678

To relax the computational complexity, the fundamental frequency can be transformed together with the spectrum as a joint vector $\mathbf{z} = [\mathbf{x}_{cc}, \mathbf{y}_{cc}, f_0]$ instead of two separate transformations. Table 4.4 shows a comparison of the standalone transformation with transformed cepstrum vectors as input and the joint transformation. Table 4.4 shows that the joint transformation decreases the standard deviation and the NPD, and surprisingly, performs better than the transformation with source cepstrum as input, Table 4.3.

The fourth transformation that was tested was the source f_0 to target f_0 transformation with training vector $\mathbf{z} = [f_0^s, f_0^t]$. Such a transformation is independent of the frequency spectrum and does utilise the correlation [25] depended upon. As shown in Table 4.5, the mean error is very large in this transformation, but the standard deviation was low and with a scaling, it had the best NPD score and

Table 4.4: Joint fundamental frequency and spectrum transformation

Method	μ_e	σ_e	*NPD*
Standalone	-0.05 Hz	26.29 Hz	1.1246
Joint transform	-3.43 Hz	22.26 Hz	0.9522

hence, it was the best transformation method of all the tested configurations.

Table 4.5: Fundamental frequency transform with source f_0 as input

Method	μ_e	σ_e	*NPD*
Standalone	-0.05 Hz	26.29 Hz	1.1246
f_0 input	39.03 Hz	20.61 Hz	0.8815

Although the transformations presented yield a perfect average fundamental frequency by scaling, the prosody in a transformed sentence is not necessarily correct. It could, however, be improved by certain ad-hoc adjustments. The transformation in question is clearly not context dependent. If the context was known the f_0 contour could be scaled, with e.g. a parabola for a typical statement, to enhance the performance. Without context information this is impossible since the prosody in a statement is clearly different than in a question.

As mentioned above, the mean error could be dealt with by a scaling, which means that the transformation with source f_0 as input with modified moving average smoothing, yields the best transformation with the GMM and test sentences in question.

4.2 Frequency Spectrum Transformation

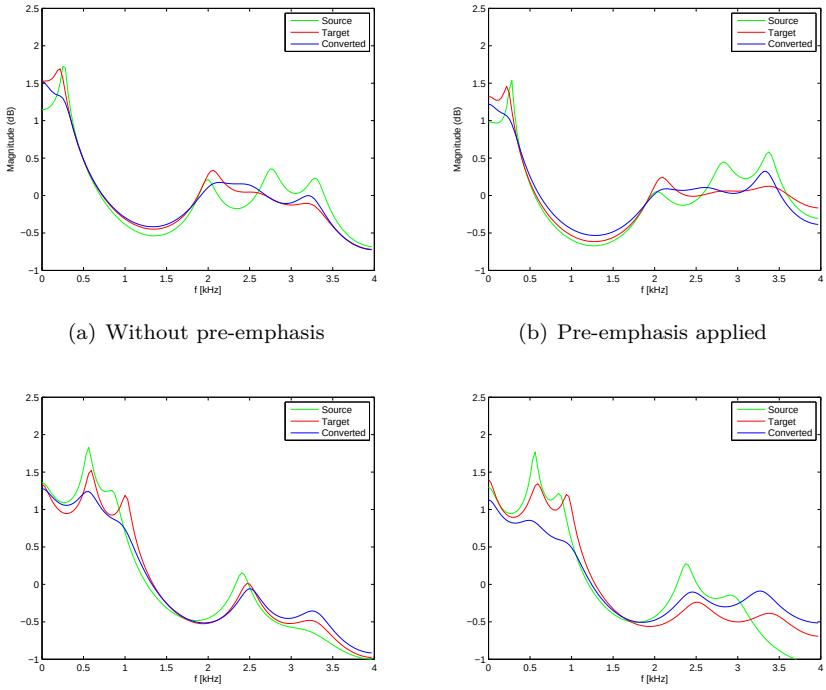
The frequency transformation was done in the cepstrum domain. A 128 mixture GMM with diagonal covariance matrices was trained with the same training sentences as the f_0 transformation, with both voiced and unvoiced segments. Four different setups were tested and the average results of the Itakura distance (2.17), the L_2 metric (2.45) and the cepstral distance (2.41) of 10 sentences are shown in Table 4.6. The distances of the source and target feature vectors before transformation are: Itakura = 0.6053, $L_2 = 4.4985$ and CD = 0.4483. An ideal transformation would yield 0 for all metrics.

The different techniques do not appear to have a positive effect on the performance compared to the straightforward transformation. By excluding the energy coefficient c_0, some of the prosody is taken out of the question. While this might be a good choice for a transformation meant for dubbing or other applications where the source speaker tried to impostor a target speaker, it is of no help in a general transformation.

Table 4.6: Absolute error in frequency transformation.

Setup	*Itakura*	L_2	*CD*
Normal	0.4280	3.4959	0.2475
Pre-emphasis	0.5139	3.8256	0.2862
c_0 excluded	0.4559	3.6762	0.2491
Pre-emph and c_0 excluded	0.5731	4.1233	0.2872

The motivation for applying pre-emphasis is to enhance the transformation quality in high frequencies. As depicted in Figure 4.2, pre-emphasis does not



(a) Without pre-emphasis

(b) Pre-emphasis applied

(c) Without pre-emphasis

(d) Pre-emphasis applied

Figure 4.2: The effect of pre-emphasis on magnitude spectrum.

appear to be necessary. Transformed spectrum does not have a higher error in the high frequencies than in the low frequencies, Figure 4.2(a) and 4.2(c). With pre-emphasis applied, the overall error is only increased, Figure 4.2(b) and 4.2(d), which supports the decrease in the objective results in Table 4.6.

A different point of view is the relative improvement to the starting point. The

normalised results are shown in Table 4.7 for the same 10 sentences. By implementing pre-emphasis the source and target spectra are also affected, which do not justify the absolute results of the pre-emphasis transform. The relative distance measures are therefore a better metric in this regard. The greatest relative improvement is achieved with pre-emphasis applied. However, the pre-emphasis made the source and target spectrum differentiate more, resulting in a larger absolute error than without pre-emphasis as shown in Table 4.6.

Table 4.7: Normalised error in frequency transformation.

Setup	*N-Itakura*	*N-L_2*	*NCD*
Normal	0.7058	0.2260	0.6059
Pre-emphasis	0.6795	0.2185	0.5954
c_0 excluded	0.7517	0.2342	0.6098
Pre-emph and c_0 excluded	0.7577	0.2312	0.5974

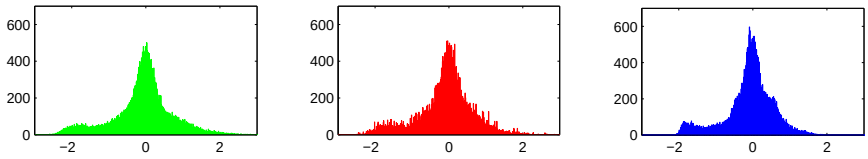
The results of the joint transformation with the fundamental frequency is compared to the standalone transformation in Table 4.8. The joint transform did not have a dramatic effect on the f_0 transform, in fact it had a lower standard deviation than the standalone transform. But the joint transformation of the spectrum is even worse than no transformation at all. The joint transformation could be used for the fundamental frequency, but the cepstrum part should be discarded and transformed separately.

Table 4.8: Comparison of standalone and joint spectrum transformation

Method	*Itakura*	*L_2*	*CD*
Standalone	0.4565	3.6852	0.2939
Joint transform	3.2840	10.157	3.0131

The absolute error is the most important metric; hence the straightforward approach was the best alternative. The average density of the LPC coefficients from 10 sentences, excluding the first which was 1 for all vectors, yielding a total of 28 000 parameters, are depicted in Figure 4.3. The source, Figure 4.3(a), and target LP coefficients, Figure 4.3(b), are almost evenly distributed, while the distribution of the converted coefficients Figure 4.3(c) differ from the target distribution. The target distribution is narrower than the source distribution, and the transformed distribution reflects these differences but a bit too much. Even though they have the same mean value, the converted parameters have a more narrow distribution and are dense in different regions than the target distribution.

Table 4.6 and Figure 4.3 show that the transformation is not perfect. It was off course not predicted to be either, but it could be better by using more mixtures in the GMM. Using more mixtures in the GMM would also require more training



(a) Source LP (b) Target LP (c) Transformed LP

Figure 4.3: Comparison of LP parameter distributions.

vectors for the fitting algorithm (EM) to converge. This is not always practically possible, but it would increase the performance as shown in the research of Y. Stylianou *et al.* [1]. 100 training sentences, as used in this implementation, is already a lot to ask of a "source speaker" in many cases. E. Helander *et al.* [30] proposed a different approach with a very small training set. By using line spectral frequencies (LSF) and exploiting their intra-frame correlations, a separate GMM can be created for each source speaker LSF element and the target LSF elements that best correlate with the current source LSF element. Yielding one full covariance GMM for each LSF element with vectors of only a few elements and only a few mixtures required for a good transformation result.

4.3 Listening Test

10 sentences with random choice of source and target speaker were transformed and presented to a test group of 12 students. The test persons were asked which speaker they thought the transformed speech belonged to. The test persons were also given the choice of no distinctive recognition. The results are presented in Table 4.9. The two test speakers had a highly distinguishable voice and there was

Table 4.9: Subjective listening results. A successful transformation was recognised as the "target".

Source speaker	*Recognised as:* *Target*	*Source*	*No decision*
A	77 %	15 %	8 %
B	67 %	28 %	5 %

no misconception in which untransformed sentence belonged to which speaker. A more detailed test score table could be found in Appendix A.

The small differences in the objective results of the frequency transformation were reflected as even smaller differences in the synthesised voice. A poor frequency transform was tested, i.e. few training vectors and small number of GMM mixtures, and yielded noise artefacts in the synthesised signal. However, the differences in

the final setups had negligible differences in the synthesised signal. Nevertheless, there are no significant disadvantages in choosing the best setup.

The pitch transform benefitted from the smoothing techniques, but it were small differences between the different smoothing techniques. Synthesised voice without smoothing applied was less natural than with smoothing. Even though the average error in the transformed f_0 was zero, the standard deviation was almost 22 Hz which resulted in a mean error in one of the test sentences was as high as 35 Hz which where definitively noticeable.

The results in Table 4.9 are promising, but the test persons were only given two choices really. If there were presented with several possible target voices, the results would probably be a lot worse. The major shortcomings of this transformation implementation are the lack of time-scale modifications. There are some time-scale modifications in the PSOLA with the transformed pitch, but pitch cycles are not discarded or duplicated. Meaning that the number of pitch periods are the same for the source speaker and the transformed voice. The test voices in the subjective test differed in a lot of aspects. The mean fundamental frequency differed by almost 30 Hz and the length in seconds of the same sentence differed on average by 20 %. It seemed to be easy to detect the correct voice if only concentrating on the pitch; even the pronunciation of words seemed recognisable. But the fact that the length of a sentence was not modified enough was the huge source of recognition error. This was especially noticeable in transformation from the slow low-pitched voice "B" to the faster speaker with a higher pitch "A". It might be possible to implement duplications and discarding of frames in the dynamic time warping. The statistical data for which frames to discard or duplication might be collected from the DTW when one feature frame is mapped to more than one frame. The number of source frames is not modified in the current implementation.

CHAPTER 5

CONCLUSION

In this thesis, a probabilistic model for transforming a voice to sound like another specific voice has been tested. The model is fully automatic and only requires some 100 training sentences from both speakers with the same acoustic content. The classical source-filter decomposition allows prosodic and spectral transformation to be performed independently. The transformations are based on a Gaussian mixture model and a transformation function suggested by Y. Stylianou [1]. Feature vectors of the same content from the source and target speaker, aligned in time by dynamic time warping, were fitted to the GMM. The short time spectra, represented as cepstral coefficients and derived from LPC [2], and the pitch periods, represented as fundamental frequency estimated from the RAPT algorithm [3], were transformed with the same probabilistic transformation function.

An important part of the training procedure is the time alignment by dynamic time warping (DTW). The DTW was configured to not alter the source feature vectors, to get a discrete mapping of every source vector to a target vector. The global constraints were set to guarantee a maximum modification of the target vectors and the local constraints were optimised for the Itakura distance [18].

Time aligned feature vectors were fitted to a 128 mixture GMM with diagonal covariance matrices, used in the transformation function. The cross-correlation matrix $\mathbf{\Sigma}^{yx}$, which is not available from the diagonal GMM, was estimated from training data. When both input and output of the transformation are known, the only unknown parameter is the cross-correlation matrix, which could be least-square optimised by the normal equations [22]. Pre-emphasis [24] and exclusion of the cepstral energy coefficient c_0 [1] were tested in the filter transform, but did not enhance the transformation quality. The transform was tested on 10 sentences and revealed an over-smoothed spectrum but with small mean deviation from the target spectrum. The fundamental frequency was transformed by a 64-mixture full-matrix GMM with different configurations of the training vector, and tested on the same 10 sentences. Objective results reviled a smaller standard deviation

than a simple scaling of the source pitch.

It is not easy to compare the quality of the transformations to other works because the different articles tend to use different metrics and even different implementations of the same metrics [1, 25, 31, 32]. However, the frequency transform is pretty straightforward, so any deviation from other's result would mainly be due to different training and test sets, or implementation errors. The main novel approach in this thesis is the use of the transformation function in the fundamental frequency transform.

The transformation function has the property of finding remaining elements of a vector given a new incomplete vector. Several such vector combinations were tested in the pitch transform. Given that the cepstrum transform was perfect, the best pitch (fundamental frequency) transformation was by using the transformed cepstrum as input. As this was not the case, the best transformation of the fundamental frequency was by mapping the source fundamental frequency to the target. The fundamental frequency contour was smoothed by a novel "modified moving average" algorithm. In spite of the fact that the modified moving average algorithm uses temporal information from consecutive transformation, which the GMM does not utilise, it performed better than a normal moving average filter.

A listening test was performed with the best setup from objective tests and the results indicate that it is possible to recognise the transformed voice as the target speaker with a 72 % probability. However, it was a pretty basic test where the identification options were limited to only the source and target speaker. The synthesised voice was affected by a muffling effect due to incorrect frequency transformation and the prosody sounded somewhat robotic. There is clearly still some work to be done before the voice transformation could be used in commercial product.

5.1 Future Work

The major shortcoming of the presented implementation is the lack of time-scale modifications. Although there is some scaling in the PSOLA synthesis where the pitch period is modified, the pitch cycles are not discarded or duplicated. If the source and target speaker have a significant different speaking pace, the lack of time-scale modification yields a problem in the speaker identification, verified in the listening test. A mapping rule for discarding or duplicating pitch cycles could be found in the DTW mapping where one vector is mapped to several vectors. However, if the goal is to impersonate a target speaker this would not be a problem. Energy transformation to modify the perceived loudness of in speech would also make the transformation more convincing. While this somewhat implemented by transforming the c_0 coefficient with questionable success, it could be implemented as part of the excitation modification as well.

The increase to correlation of consecutive transformed fundamental frequencies, the GMM could be trained by a set of f_0 parameters corresponding to a set of cepstral vectors, $\mathbf{Z} = \left[\mathbf{y}_{cc}^1, \mathbf{y}_{cc}^2, \ldots, \mathbf{y}_{cc}^n, f_0^1, f_0^2, \ldots, f_0^n\right]$ yielding a many-to-many mapping instead of one-to-one, as suggested by [25]. The complexity of the pitch

transform would increase by many orders of magnitudes, but the transformed f_0 contour might be more natural. One could also use a model that utilises temporal dependency information [32], which the GMM does not.

The excitation from the inverse LP filtering is far from white noise. By listening to the excitation signal the content of the original speech signal could be recognised. Instead of using the excitation from the source speaker in the synthesis, a small database of excitation signals from the target speaker could be used with the same transformation function to map the excitation signal to one from the target speaker. The excitation signal must represented in regular time domain which yields vectors of approximately 80 sample for a 10 ms frame and a sample frequency of 8 kHz. By choosing a small GMM, with e.g. 16 mixtures, to computational time would be acceptable.

The frequency transform would benefit from more mixtures in the GMM, but again, that would require more training data, which might be a problem. E. Helander [30] suggested a method based on LSF feature vectors instead of cepstral coefficients which might cope with the training data issue.

APPENDIX A

INDIVIDUAL LISTENING TEST SCORE

Table A.1 shows the answers of test person T to each test sentence S.

Table A.1: Listening test score

	S1	*S2*	*S3*	*S4*	*S5*	*S6*	*S7*	*S8*	*S9*	*S10*
Correct	**B**	**A**	**B**	**B**	**A**	**A**	**A**	**B**	**A**	**B**
T1	B	A	B	B	A	A	A	B	A	B
T2	A	A	B	B	A	A	A	B	A	B
T3	B	A	B	B	A	B	A	X	A	X
T4	A	A	B	B	B	A	A	A	A	B
T5	B	B	A	B	A	X	A	B	X	B
T6	B	A	B	B	B	A	X	B	A	A
T7	B	A	B	B	B	A	B	B	A	X
T8	B	A	B	B	B	B	B	B	A	B
T9	A	A	B	B	B	B	A	A	A	X
T10	B	A	B	A	B	B	A	X	B	B
T11	A	A	B	B	A	A	B	B	B	B
T12	B	A	B	B	B	A	A	B	A	B
Correct (%)	67	92	92	92	42	58	67	67	75	75

BIBLIOGRAPHY

[1] Y. Stylianou, O. Cappe, and E. Moulines, "Continuous probabilistic transform for voice conversion," *IEEE Transactions on Speech and Audio Processing*, vol. 6, pp. 131–142, March 1998.

[2] B. Atal and M. Schroeder, "Predictive coding of speech signals," *Report of the 6th. Int. Congres on Acoustics*, Tokyo, Japan 1968.

[3] D. Talkin, "A robust algorithm for pitch tracking (rapt)," in *Speech Coding and Synthesis* (W. B. Kleijn and K. K. Paliwal, eds.), ch. 14, Elsevier Science B. V., 1995.

[4] L. R. Rabiner and M. R. Sambur, "An algorithm for determining the endpoints of isolated utterances," *Bell Syst. Tech. J*, vol. 54, pp. 297–315, February 1975.

[5] Y. Stylianou, *Springer Handbook of Speech Processing*, ch. 24, pp. 489–503. Springer, October 2008.

[6] X. Huang, A. Acero, and H.-W. Hon, *Spoken Language Processing.* New Jersey: Prentice Hall PTR, 2001.

[7] M. Abe, S. Nakamura, K. Shikano, and H. Kuwabara, "Voice conversion through vector quantization," *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, vol. 1, pp. 655–658, April 1988.

[8] A. Kain and Y. Stylianou, "Stochastic modeling of spectral adjustment for high quality pitchmodification," *Proceedings of IEEE ICASSP*, vol. 2, pp. 949–952, 2000.

[9] Y. Stylianou, "Voice transformation: A survey," in *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, pp. 3585–3588, April 2009.

[10] B. P. Nguyen, *Studies on Spectral Modification in Voice Transformation.* PhD thesis, Japan Advanced Insitute of Science and Technology, March 2009.

[11] J. G. Proakis and D. G. Manolakis, *Digital Signal Processing.* Pearson Prentice Hall, 2007.

[12] G. Cybenko, "The numerical stability of the levinson-durbin algorithm for toeplitz systems of equations," *SIAM Journal on Scientific and Statistical Computing*, vol. 1, September 1980.

[13] B. P. Bogert, M. J. R. Healy, and J. W. Tukey, "The quefrency alanysis of time series for echoes," *Proceedings of the Symposium on Time Series Analysis*, pp. 209–243, 1963.

[14] W. B. Kleijn and K. K. Paliwal, *Speech Coding and Synthesis*. Elsevier Science B. V., 1995.

[15] R. M. Gray, "Vector quantization," *IEEE ASSP Magazine*, pp. 4–29, April 1984.

[16] R. E. Walpole, R. H. Myers, S. L. Myers, and K. Ye, *Probability & Statistics for Engineers & Scientists*. Pearson Prentice Hall, 8th ed., 2007.

[17] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. 28, no. 1, pp. 84–95, 1980.

[18] F. Itakura, "Minimum prediction residual principle applied to speech recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-23, pp. 67–72, February 1975.

[19] E. Kreyszig, *Introductory functional analysis with applications*. New York: John Wiley & Sons, 1989.

[20] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. New Jersey: Prentice Hall International, Inc., 1993.

[21] Y. Stylianou, O. Cappe, and E. Moulines, "Statistical methods for voice quality transformation," *Fourth European Conference on Speech Communication and Technology*, vol. 95, pp. 447–450, 1995.

[22] G. Strang, *Linear Algebra and its Applications*. Thomson Brooks/Cole, 4th ed., 2006.

[23] E. Moulines and W. Verhelst, "Prosodic modifications of speech," in *Speech Coding and Synthesis* (W. B. Kleijn and K. K. Paliwal, eds.), pp. 519–555, Elsevier, 1995.

[24] O. Turk and L. M. Arslan, "Robust processing techniques for voice conversion," *Computer Speech & Language*, vol. 20, no. 4, pp. 441–467, 2006.

[25] T. En-Najjary, O. Rosec, and T. Chonavel, "A voice conversion method based on joint pitch and spectral envelope transformation," *Eighth International Conference on Spoken Language Processing*, 2004.

[26] J. Gray, Augustine H. and J. D. Markel, "Distance measures for speech processing," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 24, no. 5, pp. 380–391, 1976.

[27] S. Chandra and W. C. Lin, "Experimental comparison between stationary and nonstationary formulations of linear prediction applied to voiced speech analysis," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 22, pp. 403–415, December 1974.

[28] NTNU and Telenor, "Fonema project." `http://www.iet.ntnu.no/projects/FONEMA/`.

[29] T. En-Najjary, O. Rosec, and T. Chonavel, "A new method for pitch prediction from spectral envelope and its application in voice conversion," *Eighth European Conference on Speech Communication and Technology*, 2003.

[30] E. Helander, J. Nurminen, and M. Gabbouj, "Lsf mapping for voice conversion with very small training sets," *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, vol. 4672, 2008.

[31] H. Ye and S. Young, "Quality-enhanced voice morphing using maximum likelihood transformations," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, pp. 1301–1312, July 2006.

[32] E.-K. Kim, S. Lee, and Y.-H. Oh, "Hidden markov model based voice conversion using dynamic characteristics of speaker," in *Fifth European Conference on Speech Communication and Technology*, 1997.