Methods for Voice Conversion

By:

Gina Upperman
Matthew Hutchinson
Brian Van Osdol
Justin Chen

Methods for Voice Conversion

By:

Gina Upperman
Matthew Hutchinson
Brian Van Osdol
Justin Chen

Online:
< http://cnx.org/content/col10252/1.2/ >

C O N N E X I O N S

Rice University, Houston, Texas

Table of Contents

Chapter 1

Introduction to Methods for Voice Conversion[1]

Speech processing is currently a key focus for many researchers in the area of DSP. In this project, we focus on the topic of **voice conversion**, which involves producing the words from one person (the "source speaker") in the voice of another person (the "target speaker").

We can do this using DSP because every person's distinct vocal qualities are essentially caused by their vocal tract, which forms a transfer function between the input excitation and the output signal that we hear. We can isolate this transfer function through methods such as **cepstral analysis** and **linear prediction coding**, which we describe in detail. The second major identifier between different speakers is the pitch range of their words. We can change the pitch through methods such as the **PSOLA**, which we also describe.

The vocal tract transfer function and pitch range are different for different sounds. Thus, in synthesizing a phrase, we must first break the signal into smaller segments and analyze each individually. Our windowing algorithm divides the signal based on breaks between different syllables and words. We then use functions from the Praat program developed by Paul Boersma and David Weenink of the University of Amsterdam (www.praat.org[2]) to perform the analysis and synthesis.

Voice conversion has numerous applications, such as the areas of foreign language training and movie dubbing. It is closely related to the process of **speech synthesis**, which usually refers to converting text into spoken language, and has many applications, especially relating to assistance for the blind and deaf. Other areas in speech processing, such as **speaker verification**, have applications in security. All of these different types of speech signal processing involve related methods that we investigated through this project, especially cepstral analysis, linear prediction coding, and the PSOLA method.

Project Contents

- Introduction
- The Source Filter Model of Speech (Chapter 2)
- Deconvolution Basics (Chapter 3)
- The Cepstrum (Chapter 4)
- Linear Predictive Coding (Chapter 5)
- Changing Pitch with PSOLA (Chapter 6)
- Signal Windowing (Chapter 7)
- Using Praat (Chapter 8)
- MATALB and Praat Code (Chapter 9)
- Conclusion (Chapter 10)

[1]This content is available online at <http://cnx.org/content/m12479/1.4/>.
[2]http://cnx.org/content/m12479/latest/www.praat.org

Interactive Demonstration

This is an unsupported media type. To view, please see
http://cnx.org/content/m12479/latest/project_demo.swf

Figure 1.1: This comic is compliments of Brian VanOsdol. Go here for more BoyDog Comics.[3]

[3]http://www.owlnet.rice.edu/~vanosdol/gallery/boy_dog_03.html

Chapter 2

The Source Filter Model of Speech[1]

2.1 The Make-Up of Speech

The components of speech are the **words** and the **voice**.

Every phrase is a union of these two components - they are the foundations of the spoken language. One or the other does not mean much without its conterpart. Words without voice lack intonation, so they have no meaning. Voice without words is devoid of structure and cannot possibly transfer information. Only the fusion of the two can claim to be such a thing as speech.

In biology, the components of speech are produced in different organs. To speak, air is first released over the vocal cords, which expand and contract to give the air column structure. This is the biological concept of words. The words are then passed through the vocal tract where they are shaped, giving them intonation. This shaping of the words is the biological concept of voice. Such a biological process can be easily modeled.

So far, we have determined that speech is a collection of words shaped by voice. Here, we present a model of this. In this model, the words are called the **source**. Since the words are modified by voice, we say the source passes through a **filter**. This brings us to the **source filter model** of speech.

Definition 2.1: Source Filter Model
The **source filter model** is a model of speech where the spoken word is comprised of a source component originating from the vocal cords which is then shaped by a filter immitating the effect of the vocal tract.

Example

The Source Filter Model

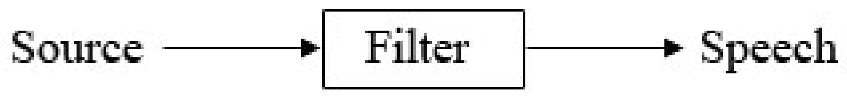

Figure 2.1: A simple model of speech.

[1]This content is available online at <http://cnx.org/content/m12470/1.5/>.

This model has possibility for application in many different fields. We will focus on the topic of signal processing here.

2.2 Signal Processing Considerations

The source filter model can easily be extended to signal processing. The source is simply a signal $x(t)$. This signal is the input to the filter and is called the **excitation signal** since it excites the vocal tract. The vocal tract is a filter similar to all filters we have studied so far: it is a linear time-invariant system with impulse response $h(t)$. This is sometimes called the **transfer function** of speech since it is what transfers the excitation signal to speech - it adds voice to words. Speech is the output $y(t)$ of the source signal $x(t)$ passed through the filter with impulse response $h(t)$. Thus, the output is given by $y(t) = x(t) * h(t)$. This is depicted below:

Signal Processing Representation of the Source Filter Model

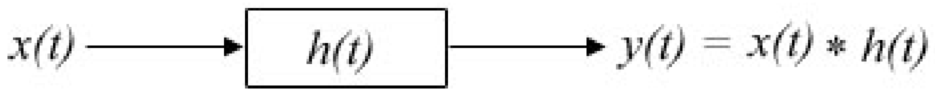

Figure 2.2: An input x(t) to a filter with impulse response h(t) yields the convolution of the two.

Since speech is simply a convolution of a source signal $x(t)$ with a filter's input response $h(t)$, we can analyze these signals to determine the characeristics of a speech signal $y(t)$. However, we must first deconvolve these signals so that they can be processed individually. This topic is explored in the next section covering deconvolution (Chapter 3).

2.3 References

Huckvale, Mark. "Lecture 8: Source-Filter Model of Speech Production." B214: Phonetic Science: Acoustics of Speech and Hearing. University College London. http://www.phon.ucl.ac.uk/courses/spsci/b214/lect1-8.pdf[2] .

Johnson, Don. Connexions module m0049: Modeling the Speech Signal[3].

[2]http://www.phon.ucl.ac.uk/courses/spsci/b214/lect1-8.pdf
[3]"Modeling the Speech Signal" <http://cnx.org/content/m0049/latest/>

Chapter 3

Deconvolution Basics[1]

3.1 Definition

Deconvolution is exactly what it sounds like: the undoing of convolution. This means that instead of mixing two signals like in convolution, we are isolating them. This is useful for analyzing the characteristics of the input signal and the impulse response when only given the output of the system. For example, when given a convolved signal $y(t) = x(t) * h(t)$, the system should isolate the components $x(t)$ and $h(t)$ so that we may study each individually. An ideal deconvolution system is shown below:

Ideal Deconvolution System

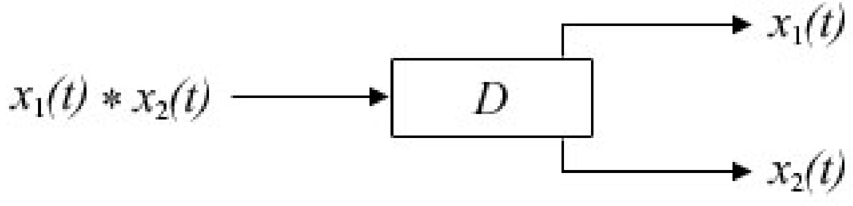

Figure 3.1: A system that performs deconvolution separates two convolved signals.

3.2 Approach

Instead of producing one system that outputs both the convolved signals, it will be much easier for our purposes to consider separate systems that output one of the signals at a time. Thus, we desire the following systems:

[1]This content is available online at <http://cnx.org/content/m12472/1.5/>.

Separate Systems

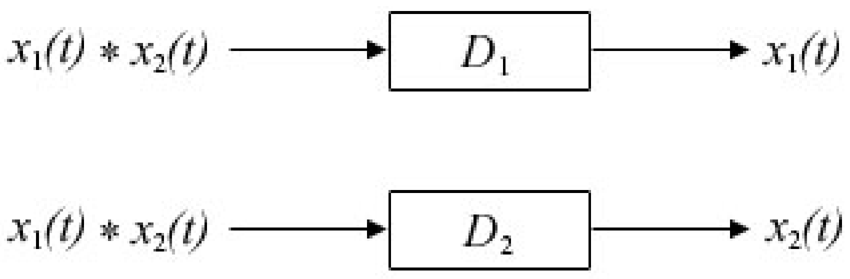

Figure 3.2: The process of deconvolution is facilitated by splitting the process into separate systems.

What it looks like each of these systems is doing is annihilating the undesired signal. This is, in fact, exactly correct. This system is a **homomorphic filter**.

Definition 3.1: Homomorphic Filter
A **homomorphic filter** is a system which accepts a signal composed of two components and returns the signal with one of the components removed.

A frequently applied method is to convert the convolution of two signals into a sum, and then implement a homomorphic filter to remove one of the signal components. This is the basis for cepstral analysis, so we will cover this later. A diagram of this method follows:

A Possible Deconvolution Method

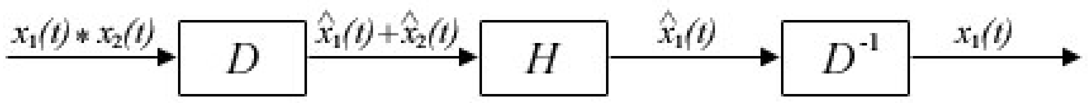

Figure 3.3: This is the basic deconvolution method implemented in cepstral analysis.

The isolation of two convolved signals depends greatly on the characteristics of both signals. Thus, a wide variety of deconvolution methods exist. Since this is a study on speech analysis, we will cover only the deconvolution methods which focus the signals of the source filter model: the excitation signal and the impulse response of the vocal tract filter.

3.3 Deconvolution Methods for Speech Analysis

A few deconvolution methods that we will use in speech analysis are:

- Cepstral Analysis (Chapter 4)
- Linear Predictive Coding (Chapter 5)

We study the first of these in the next area covering the cepstrum (Chapter 4).

3.4 References

Rabiner, Lawrence R, and Schafer, Ronald W. Digital Processing of Speech Signals. Bell Laboratories, 1978.

Chapter 4

Cepstrum[1]

4.1 The Cepstrum Domain

The **cepstrum** is a common transform used to gain information from a person's speech signal. It can be used to separate the **excitation** signal (which contains the words and the pitch) and the transfer function (which contains the voice quality). It is similar to a channel vocoder or LPC (Chapter 5) in its applications, but using the cepstrum as a spectral analyzer is a completely different process. (It is also worth pointing out that cepstrum is "spectrum" with the first syllable flipped... we will encounter several words with this naming convention.) Before describing the details of the cepstrum, a little background in speech models is needed.

4.2 Background Information

Within human speech, there are two methods employed to form our words. These sounds are categorized into the **voiced** and **unvoiced**. For the voiced part, our throat acts like a transfer function. The vowel sounds are included in this category. The unvoiced part describes the "noisy" sounds of speech. These are the sounds made with our mouth and tongue (as opposed to our throat), such as the "f" sound, the "s" sound, and the "th" sound. This way of looking at speech as two seperable parts is known as the Source Filter Model of Speech (Chapter 2) or the Linear Separable Equivalent Circuit Model. Mathematically, they are described in the time domain as:

$$x(t) = \int_0^t g(\tau)\, h(t-\tau)\, d\tau$$

Since convolution in the time domain is multiplication in the frequency domain, this become:

$$X(\omega) = G(\omega)\, H(\omega)$$

There is a mathematical process with which we are familiar that can separate two multiplied variables. If we take the log of the magnitude of both sides of this transform, we reach:

$$\log|X(\omega)| = \log|G(\omega)| + \log|H(\omega)|$$

Computing the inverse Fourier Transform of this equation brings us into the realm of "quefrency."

$$F^{-1}\log|X(\omega)| = F^{-1}\log|G(\omega)| + F^{-1}\log|H(\omega)|$$

[1] This content is available online at <http://cnx.org/content/m12469/1.4/>.

Quefrency is the x-axis of the cepstrum. Its units are in time. Typically the areas of intest are from 0ms to around 10ms. See figure 1 below for the full process.

Cepstrum Block Diagram

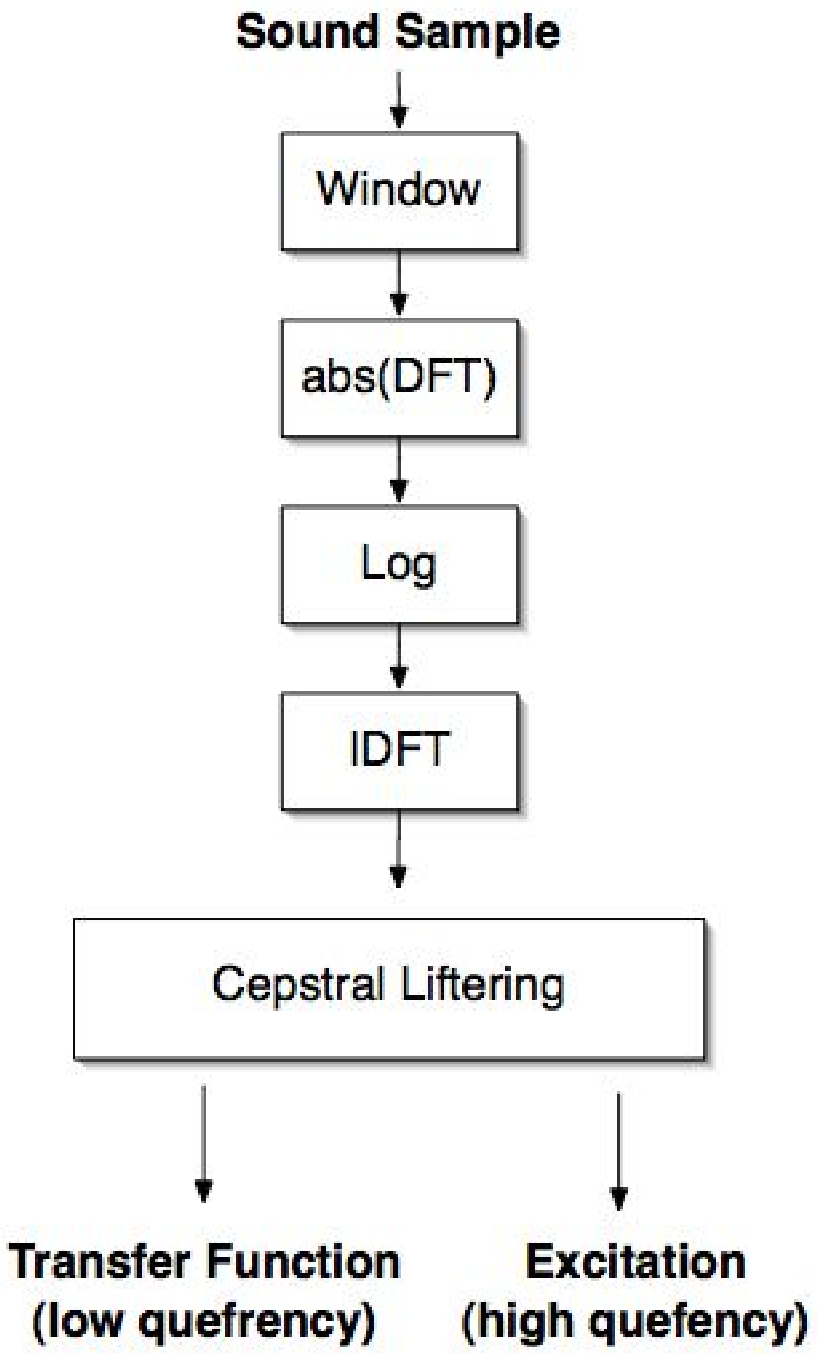

Figure 4.1: This is the process used to compute the cepstrum

4.3 The Use of the Cepstrum

We have now seen the process by which we calculate the cepstrum of a signal. It is now time to dicuss some the uses of the cepstrum. Often after having calculated the cepstrum, we will want to "lifter" the signal. (Once again the naming scheme has been used. This time on the word filter) When we lifter, we are seperating the **transfer function** (the spectral envelope) and the **excitation signal**. The transfer function usually appears as a steep slant at the beginning of the plot. The excitation appears as **periodic peaks** occurring after around 5ms. Below we can see examples of several cepstrum plots. Note how the female voice has peaks occurring more often then in the male's cepstrum. This is due to the higher pitch of a female voice.

Cepstrum Samples

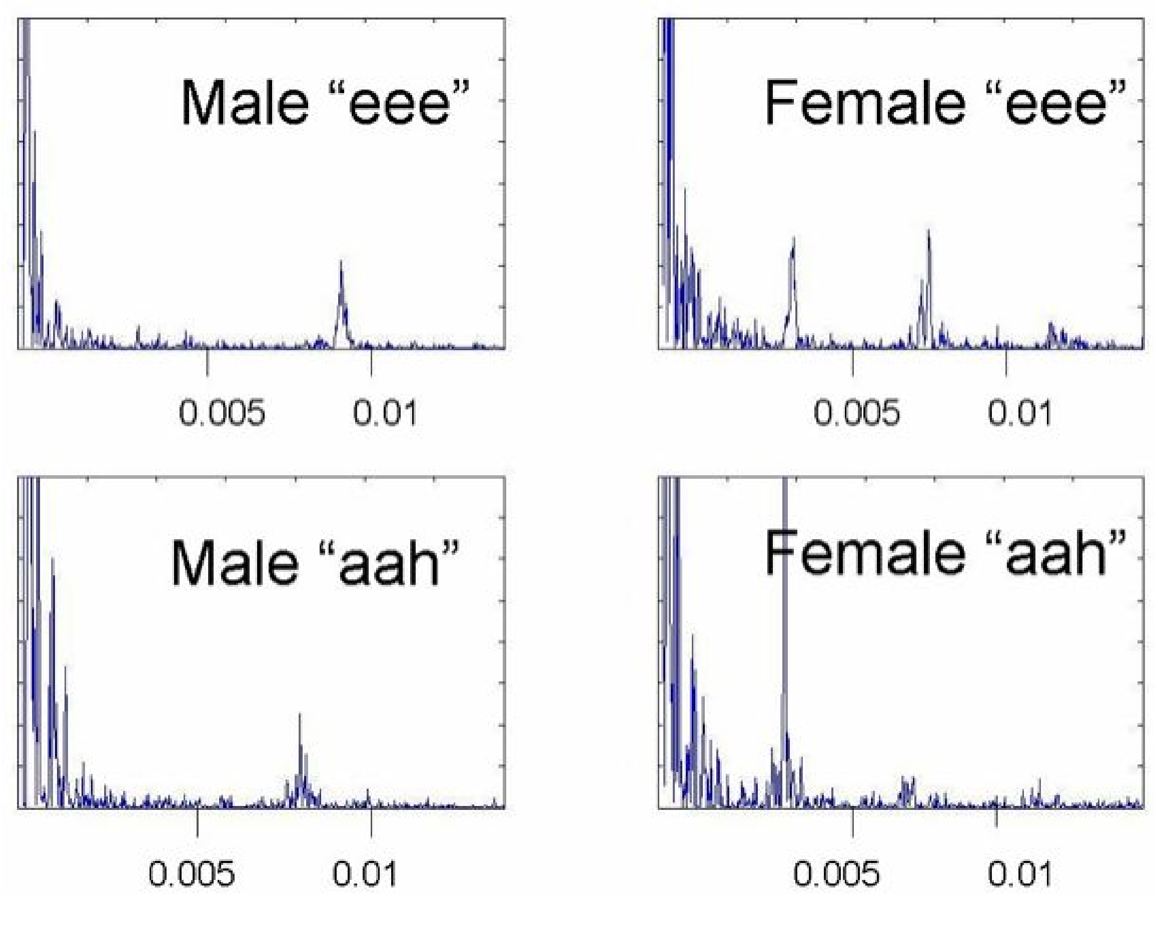

Figure 4.2: These are vowel samples of a male and female. The x-axis is in milliseconds and the y-axis is the magnitude. The sounds "aah" and "eee" for both speakers are shown.

4.4 Reference

Furui, Sadaoki. Digital Speech Processing, Synthesis, and Recognition. Marcel Deccer, Inc.: New York.

Gold, Ben and Nelson Morgan. Speech and Audio Signal Processing: Processing and Perception of Speech and Music. John Wiley and Sons, Inc: New York. 2000.

Chapter 5

Linear Predictive Coding in Voice Conversion[1]

5.1 Background on Linear Predictive Coding

Linear Predictive Coding (or "LPC") is a method of predicting a sample of a speech signal based on several previous samples. Similar to the method employed by the cepstrum (Chapter 4), we can use the LPC coefficients to separate a speech signal into two parts: the transfer function (which contains the vocal quality) and the **excitation** (which contains the pitch and the sound). The method of looking at speech as two parts which can be separated is known as the Source Filter Model of Speech (Chapter 2).

We can predict that the nth sample in a sequence of speech samples is represented by the weighted sum of the p previous samples:

$$\hat{s} = \sum_{k=1}^{p} a_k s\left[n-k\right]$$

The number of samples (p) is referred to as the "order" of the LPC. As p approaches infinity, we should be able to predict the nth sample exactly. However, p is usually on the order of ten to twenty, where it can provide an accurate enough representation with a limited cost of computation. The weights on the previous samples (ak) are chosen in order to minimize the squared error between the real sample and its predicted value. Thus, we want the error signal e(n), which is sometimes referred to as the LPC residual, to be as small as possible:

$$e\left[n\right] = s\left[n\right] - \hat{s}\left[n\right] = s\left[n\right] - \sum_{k=1}^{p} a_k s\left[n-k\right]$$

We can take the z-transform of the above equation:

$$E\left(z\right) = S\left(z\right) - \sum_{k=1}^{p} a_k S\left(z\right) z^{-k} = S\left(z\right)\left[1 - \sum_{k=1}^{p} a_k z^{-k}\right] = S\left(z\right) A\left(z\right)$$

Thus, we can represent the error signal E(z) as the product of our original speech signal S(z) and the transfer function A(z). A(z) represents an all-zero digital filter, where the ak coefficients correspond to the zeros in the filter's z-plane. Similarly, we can represent our original speech signal S(z) as the product of the error signal E(z) and the transfer function 1 / A(z):

[1]This content is available online at <http://cnx.org/content/m12473/1.4/>.

$$S(z) = \frac{E(z)}{A(z)}$$

The transfer function 1/A(z) represents an all-pole digital filter, where the ak coefficients correspond to the poles in the filter's z-plane. Note that the roots of the A(z) polynomial must all lie within the unit circle to ensure stability of this filter.

The spectrum of the error signal E(z) will have a different structure depending on whether the sound it comes from is voiced or unvoiced. Voiced sounds are produced by vibrations of the vocal cords. Their spectrum is periodic with some fundamental frequency (which corresponds to the pitch). Examples of voiced sounds include all of the vowels. Unvoiced signals, however, do not have a fundamental frequency or a harmonic structure. Instead, they are just white noise.

5.2 LPC in Voice Conversion

In speech processing, computing the LPC coefficients of a signal gives us its ak values. From here, we can get the filter A(z) as described above. A(z) is the transfer function between the original signal s[n] and the excitation component e[n]. The transfer function of a speech signal is the part dealing with the voice quality: what distinguishes one person's voice from another. The excitation component of a speech signal is the part dealing with the particular sounds and words that are produced. In the time domain, the excitation and transfer function are convolved to create the output voice signal. As shown in the figure below, we can put the original signal through the filter to get the excitation component. Putting the excitation component through the inverse filter (1 / A(z)) gives us the original signal back.

A Voice Conversion Algorithm

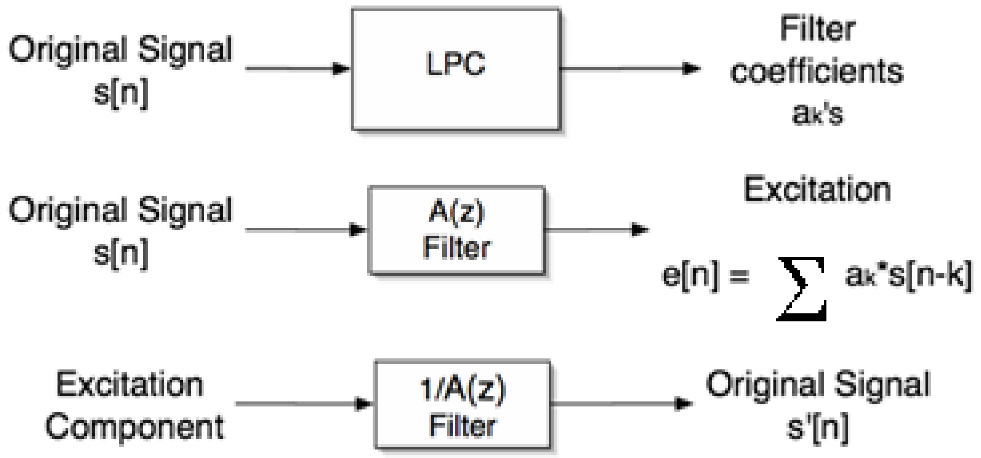

Figure 5.1: Using Linear Predictive Coding to separate the two parts of a speech signal: transfer function and excitation.

We can perform voice conversion by replacing the excitation component from the given speaker with a new one. Since we are still using the same transfer function A(z), the resulting speech sample will have the same voice quality as the original. However, since we are using a different excitation component, the resulting speech sample will have the same sounds as the new speaker.

5.3 Pre-Emphasis

In speech processing, a process called **pre-emphasis** is applied to the input signal before the LPC analysis. During the reconstruction following the LPC analysis, a de-emphasis process is applied to the signal to reverse the effects of pre-emphasis.

Pre- and de- emphasis are necessary because, in the spectrum of a human speech signal, the energy in the signal decreases as the frequency increases. Pre-emphasis increases the energy in parts of the signal by an amount inversely proportional to its frequency. Thus, as the frequency increases, pre-emphasis raises the energy of the speech signal by an increasing amount. This process therefore serves to flatten the signal so that the resulting spectrum consists of formants of similar heights. (Formants are the highly visible resonances or peaks in the spectrum of the speech signal, where most of the energy is concentrated.) The flatter spectrum allows the LPC analysis to more accurately model the speech segment. Without pre-emphasis, the linear prediction would incorrectly focus on the lower-frequency components of speech, losing important information about certain sounds.

5.4 References

Deng, Li and Douglas O"Shaughnessy. Speech Processing: A Dynamic and Optimization-Oriented Approach. Marcel Dekker, Inc: New York. 2003.

Gold, Ben and Nelson Morgan. Speech and Audio Signal Processing: Processing and Perception of Speech and Music. John Wiley and Sons, Inc: New York. 2000.

Lemmetty, Sami. Review of Speech Synthesis Technology. (Master's Thesis: Helsinki University of Technology) March 1999. http://www.acoustics.hut.fi/~slemmett/dippa/thesis.pdf[2] .

Markel, J.D. and A.H. Gray, Jr. Linear Predition of Speech. Springer-Verlag: Berlin. 1976.

[2] http://www.acoustics.hut.fi/~slemmett/dippa/thesis.pdf

Chapter 6

Changing Pitch with PSOLA for Voice Conversion[1]

PSOLA ("Pitch-Synchronous Overlap and Add") is a method used to manipulate the pitch of a speech signal to match it to that of the target speaker.

PSOLA deals with **diphones**, which are the units of speech that extend from the middle of one region of steady-state sound to the middle of the next; thus, they represent transitions between speech sounds. Some researchers have suggested that this classification of transitions between sounds is the key element in humans understanding and recognizing segments of speech.

The basic algorithm for the PSOLA technique consists of three steps. First, the speech signal is divided into separate but overlapping smaller signals. This is accomplished by windowing segments around each "pitch mark" or peak amplitude in the original signal. The windowed segments usually contain two to four pitch periods. Secondly, the smaller signals are modified by either repeating or leaving out speech segments, depending on whether the pitch of the target speaker is higher or lower than the pitch of the source speaker. This modifies the duration of the signal, therefore changing the fundamental frequency. Lastly, the remaining smaller segments are recombined through overlapping and adding. The result is a signal with the same spectrum as the original but with a different fundamental frequency. Thus, the pitch changes, but the other vocal qualities remain the same.

The original PSOLA is now often referred to as the **TD-PSOLA**, for "time domain." An alternative is referred to as the **LP-PSOLA**. Rather than storing the diphone waveforms, the LP-PSOLA stores LP ("linear predictor") coefficients in order to represent the segment of a signal.

6.1 References

Gold, Ben and Nelson Morgan. Speech and Audio Signal Processing: Processing and Perception of Speech and Music. John Wiley and Sons, Inc: New York. 2000.

Lemmetty, Sami. Review of Speech Synthesis Technology. (Master's Thesis: Helsinki University of Technology) March 1999. http://www.acoustics.hut.fi/~slemmett/dippa/thesis.pdf[2] .

[1]This content is available online at <http://cnx.org/content/m12474/1.3/>.
[2]http://www.acoustics.hut.fi/~slemmett/dippa/thesis.pdf

Chapter 7

Windowing in Voice Conversion[1]

The characteristics of a speech signal vary with time. For this reason, it is difficult to process an entire phrase as tone, pitch, and other characteristics may have a very large range over the whole of the signal. Thus, it is necessary to split the phrase into many parts, or **window** the speech signal.

There are many different ways to window a signal. The first, and most common, method is to split the signal into equal parts. While this may prove useful for many applications, it is not the best technique for speech processing.

Speech is composed of many strung together segments known as syllables. Since each syllable contains unique characteristics, we find it very useful to split phrases into syllables before processing the signal. Syllable extraction amounts to looking for the breaks in the speech signal. This can be accomplished by performing **envelope extraction** on the absolute value of the speech signal and comparing to a threshold.

Definition 7.1: Envelope Extraction

Envelope Extraction is the process of obtaining the **evelope**, or general shape of, a signal.

In speech processing, the envelope can be extracted by taking the absolute value of the speech signal and subjecting it to an averager. An averager simply convolves the signal with a boxcar. The signal envelope is then compared to a threshold.

This is the process we employed in our windowing algorithm. A diagram is shown below:

[1]This content is available online at <http://cnx.org/content/m12476/1.4/>.

Speech Signal Envelope

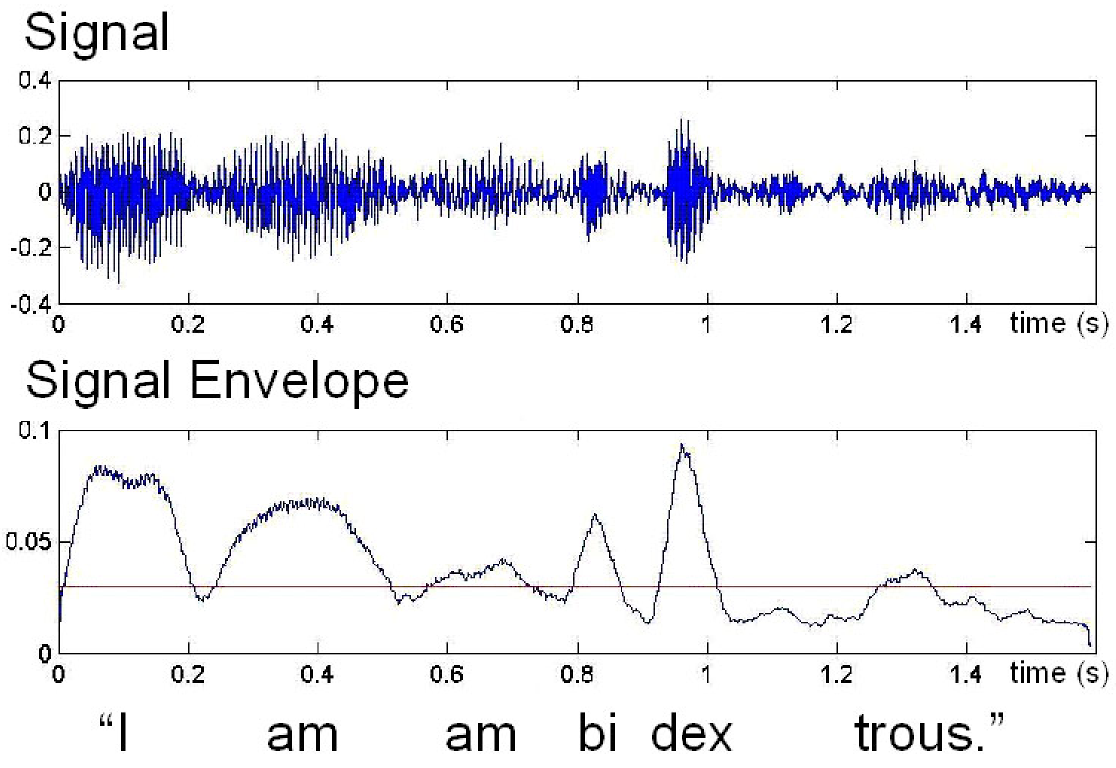

Figure 7.1: A speech signal and the envelope of its magnitude compared to a threshold.

When the signal envelope falls below the predetermined threshold, the speech signal is assumed to have a "break" in it - the end of one syllable and the beginning of another. Thus, the signal is split and each syllable is sent to the system for processing. After processing, the windows are reassembled to obtain the results for the entire phrase.

Chapter 8

Voice Conversion in Praat[1]

The Praat Program, developed by Paul Boersma and David Weenink of the University of Amsterdam, provides several modules. The ones we were interested in were namely the LPC and pitch conversion modules. These modules can be combined to convert from a source speaker to a target.

8.1 Opening a Sound File in Praat

Praat is a program that offers several different ways to process and manipulate sound files. When a sound file is opened in Praat, it is automatically converted into an "object," which is the only type of data that the program can work with. To open a sound file, select "Read from file..." from the "Read" menu.

8.2 LPC Filters

Using Praat, it is possible to calculate the LPC filter coefficients of a sound object. To do this, select the sound object and use the function "To LPC (burg)" under "Formants and LPC." This function requires the following arguments:

Prediction order - The number of linear prediction coefficients.

Analysis window duration - The duration of each analysis frame, in seconds.

Time step - The time step between two consecutive analysis frames, in seconds.

Pre-emphasis frequency - A +6dB / octave filtering will be applied above this frequency (Hz). If you do not want pre-emphasis, choose a frequency greater than the Nyquist frequency.

This function will return an LPC object. To filter or inversely filter a sound object with an LPC object, simply select both of them simultaneously and choose the appropriate option. Inversely filtering a sound object with its associated LPC object will yield the excitation (or source) part of the sound. This excitation can be filtered with a different LPC object than the one it was created with to obtain the characteristics of another sound.

8.3 Changing Pitch

Praat can also be used to extract pitch information from a sound. To do this, a sound object must first be converted to a manipulation object through the "To Manipulation..." function. When a sound object is converted to a manipulation object, Praat automatically calculates the sound's pitch information using the PSOLA method. After selecting a manipulation object, the "Extract pitch tier" function can be used to obtain this pitch information.

[1]This content is available online at <http://cnx.org/content/m12475/1.4/>.

A manipulation object's pitch tier can be replaced with a separate pitch tier object. Selecting both objects and using the "Replace pitch tier" function will accomplish this. In order to get a sound object from a manipulation object, select "Resynthesize (LPC)."

8.4 Voice Conversion Algorithm

The preceding processes can be automated through a Praat script. The following block diagram illustrates one method of performing voice conversion in Praat:

A Voice Conversion Algorithm

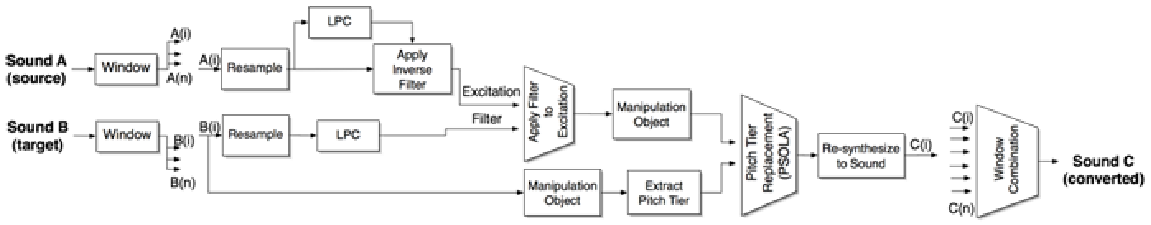

Figure 8.1: The windowing for this process is done in Matlab. The rest of the system can be coded entirely in Praat.

8.5 References

Praat: Doing Phonetics by Computer. Paul Boersma and David Weenink of the University of Amsterdam. www.praat.org[2] .

[2]http://cnx.org/content/m12475/latest/www.praat.org

Chapter 9

Code for Voice Conversion[1]

Praat and MATLAB Code Files for Voice Converstion

- ursula.praat[2] - The Praat code to convert windowed source files into the corresponding windowed filter files. (Named after the antagonist of "The Little Mermaid", who imitates voices.)
- add_wav.m[3] - Adds the sound data to the wav array.
- convert_voice.m[4] - Coverts the voice of the source to that of the filter.
- example_plot.m[5] - Plots example cepstrums.
- get_mins.m[6] - Obtains a rough list of minimums in a signal.
- get_word_breaks.m[7] - Gets the position of the spaces in the speech signal.
- load_wav.m[8] - Loads the specified wav file.
- load_wavs.m[9] - Loads all wavs in the current directory.
- plot_cepstrum.m[10] - Plots the cepstrum of the signal.
- plot_contour.m[11] - Plots the signal, its absolute value, and its contour.
- save_chunks.m[12] - Saves segments of the source and filter files.
- save_words.m[13] - Saves the words of the source and filter files.
- sig_chunks.m[14] - Breaks a signal into chunks.
- split_sig.m[15] - Splits the signal in half.

For more details, consult the comments in each code file.

[1]This content is available online at <http://cnx.org/content/m12477/1.6/>.
[2]http://cnx.org/content/m12477/latest/ursula.pratt
[3]http://cnx.org/content/m12477/latest/add_wav.m
[4]http://cnx.org/content/m12477/latest/convert_voice.m
[5]http://cnx.org/content/m12477/latest/example_plot.m
[6]http://cnx.org/content/m12477/latest/get_mins.m
[7]http://cnx.org/content/m12477/latest/get_word_breaks.m
[8]http://cnx.org/content/m12477/latest/load_wav.m
[9]http://cnx.org/content/m12477/latest/load_wavs.m
[10]http://cnx.org/content/m12477/latest/plot_cepstrum.m
[11]http://cnx.org/content/m12477/latest/plot_contour.m
[12]http://cnx.org/content/m12477/latest/save_chunks.m
[13]http://cnx.org/content/m12477/latest/save_words.m
[14]http://cnx.org/content/m12477/latest/sig_chunks.m
[15]http://cnx.org/content/m12477/latest/split_sig.m

Chapter 10

Voice Conversion Experiment and Conclusion[1]

10.1 Description of Experiment

To test our voice conversion algorithm, we administered a speaker identification test to twelve randomly selected people. Prior to the experiment, we recorded speech samples from four different speakers (two male and two female) and used our algorithm to convert between various combinations of their voices. For example, we took the sound of speaker #1 (the "source speaker") saying a certain phrase and converted it to the voice of speaker #2 (the "target speaker"). The participants listened to a series of these synthesized sounds, and we asked them to identify the speaker (the target) as well as the speaker's gender.

10.2 Results of Experiment

The target speaker was correctly identified **74%** of the time.

The target speaker's gender was correctly identified **93%** of the time.

[1]This content is available online at <http://cnx.org/content/m12478/1.4/>.

Graph of Gender-specific Conversion Accuracy

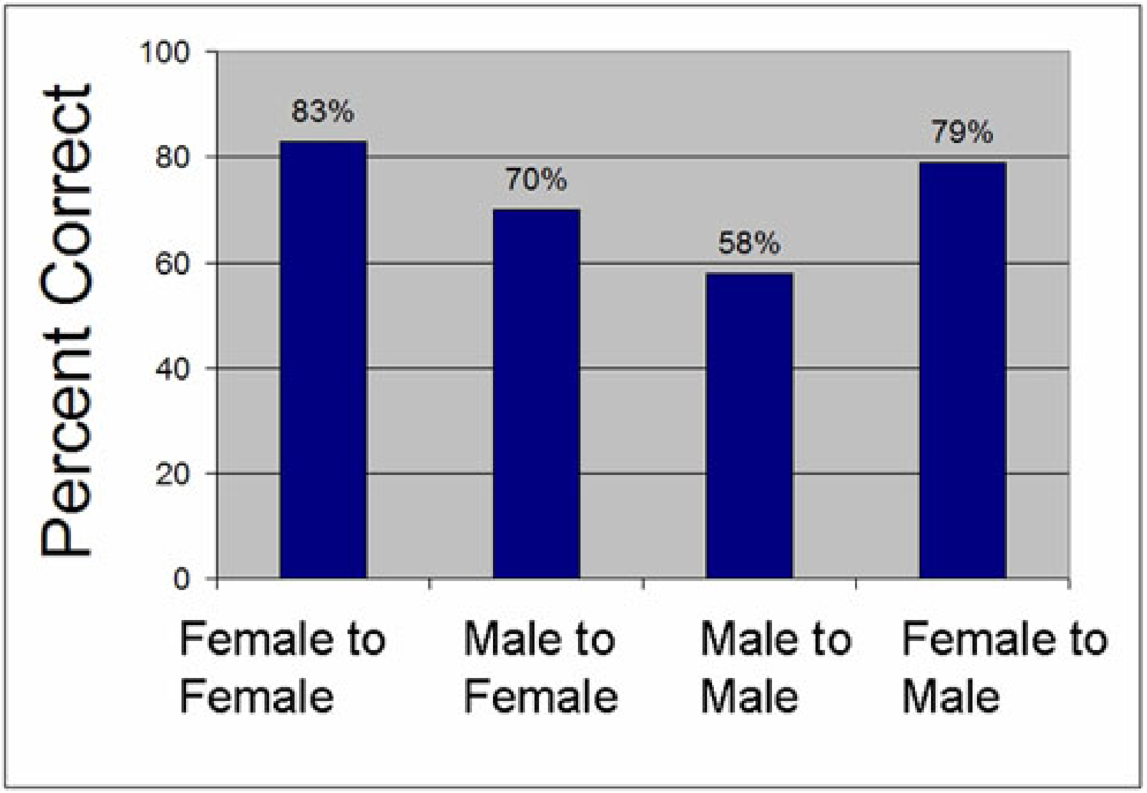

Figure 10.1: The first bar, "Female to Female," indicates a conversion from a female source speaker to a female target speaker was correctly identified 83% of the time.

10.3 Conclusions

Our voice conversion system was fairly effective at imitating a certain target speaker. From the "Gender-Specific Conversion Accuracy" graph, it can be implied that our system was better at converting female source speakers than male source speakers. One reason for this may be that the voices of the two male speakers used in the experiment had only a minor difference in pitch. The female speakers' voices, however, had a more noticeable difference.

10.4 Possible Improvements

At its current state, our system can only convert between two voices when it has samples of the speakers saying the same word or phrase. In order to make our system text-independent, we would need to implement **neural mapping**. This could be accomplished by using the **cepstrum** to identify certain characteristic sounds (such as vowel sounds) in the target speaker's speech sample and mapping their filters to the corresponding characteristic sounds in the source speaker's sample. In addition to adding text-independence to our system, we could add a **band-pass filter** at the end of our system to help eradicate speech artifacts in our synthesized sounds. The filter would block out frequencies that are not in the range of human speech.

Glossary

E **Envelope Extraction**

Envelope Extraction is the process of obtaining the **evelope**, or general shape of, a signal.

H **Homomorphic Filter**

A **homomorphic filter** is a system which accepts a signal composed of two components and returns the signal with one of the components removed.

S **Source Filter Model**

The **source filter model** is a model of speech where the spoken word is comprised of a source component originating from the vocal cords which is then shaped by a filter immitating the effect of the vocal tract.

Example:

The Source Filter Model

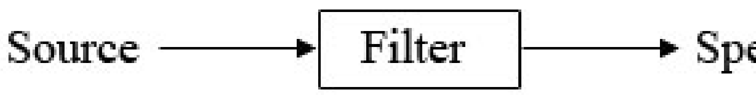

Figure 2.1: A simple model of speech.

Index of Keywords and Terms

Keywords are listed by the section with that keyword (page numbers are in parentheses). Keywords do not necessarily appear in the text of the page. They are merely associated with that section. *Ex.* apples, § 1.1 (1) **Terms** are referenced by the page they appear on. *Ex.* apples, 1

Attributions

Methods for Voice Conversion
This course explores methods in signal processing to perform voice conversion: producing the words from one speaker in the voice of another. This is the Elec 301 project of Justin Chen, Matthew Hutchinson, Gina Upperman, and Brian VanOsdol.