Introduction to Computer Music

Week 9 The Human Voice

Version 1, 15 Oct 2018

Roger B. Dannenberg

Topics Discussed: **Source Filter Models, Linear Prediction Coding (LPC) Vocoder, VOSIM, FOF, Phase Vocoder, MQ, SMS**

1 Introduction: Source Filter Models

This chapter is about creating sounds of the human voice. You have undoubtedly heard speech synthesizers, sometimes called text-to-speech systems. Musicians have adopted speech synthesis algorithms to music and created some new algorithms used exclusively for music.

Why speech and singing? Vocal sounds are very important in daily life, and we are very sensitive to the sound of the human voice. Singing synthesis and vocal sounds are often used in computer music because that they grab our attention. The tension formed when a sound is recognizably both human and not human at the same time makes for interesting listening. In this lecture, we are going to learn about how voice sounds are made, what are their characteristics and also a number of different algorithms for achieving those kinds of results.

We will begin this story of human voice and singing synthesis with the introduction to *source-filter models*.

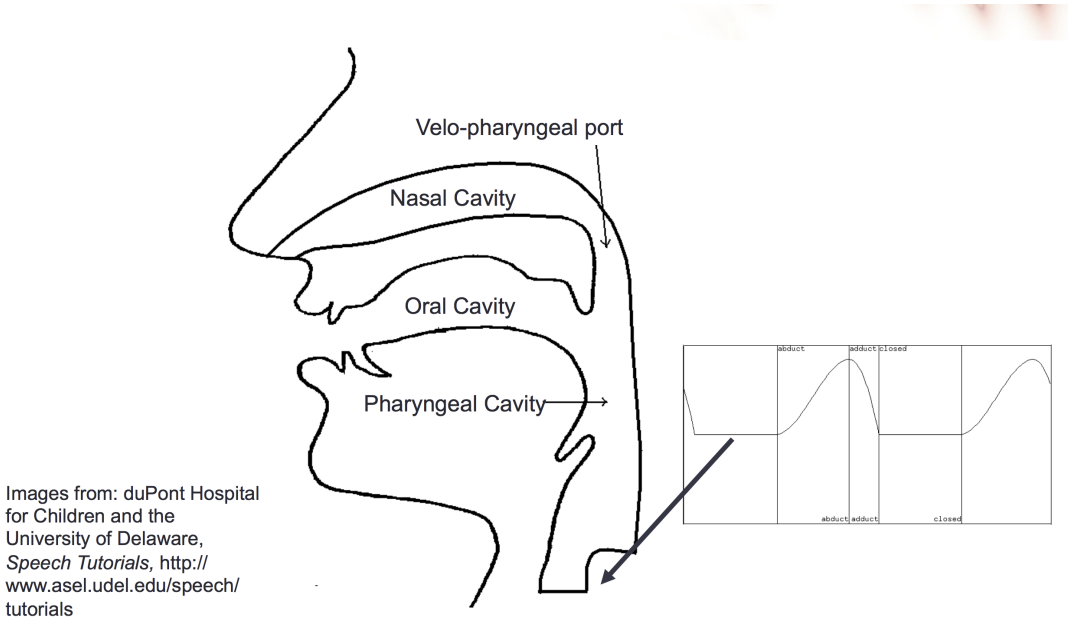
1.1 How The Voice Works

Figure 1 is a diagram of a human head, taken at a cross-section so that we can see the air passageways and how the voice works. At the bottom of the diagram are the vocal folds (commonly called *vocal cords*), which is the structure that vibrates when you push air through the larynx. The waveform at the vocal fold is a series of pulses formed as the air passage opens and closes. The right side of Figure 1 shows roughly what the waveform looks like. Pulses from the vocal folds are rounded, and we have some control over both the shape and frequency of the pulses by using our muscles.

These pulses go into a resonant chamber that is formed by the oral cavity (the mouth) and the nasal cavity. The shape of the oral cavity can be manipulated extensively by the tongue and lips. Although you are usually unaware of your tongue and lips while speaking, you can easily attend to them as you produce different vocal sounds to get some idea of their function in speech production and singing.

One of the most fundamental concepts of the speech production is the vowel. Vowels are sounds we can sustain, and include "a, e, i, o, u". When speaking these vowels with same pitch and volume, what changes is the spectrum. In fact, the main characteristic of the spectral changes between different vowels are changes in *resonances*, which are narrow ranges of frequencies that are boosted. The main characteristic of a resonance is the center frequency, and the resonances with the two lowest frequencies are the most important ones in determining the vowel we hear. These two resonances are also called *formants*.

If we plot the first formant frequency verses the second formant frequency for different vowel sounds (see Figure 2, we can get the picture shown in Figure 2. For example, if the first formant is a little below 500 Hz, and the second formant is around 1500 Hz, we will get vowel sound `"ir"` as in `"sir"`. On the other hand, if we increase those frequencies to around 800 Hz for the first formant and close to 2000 Hz for the second one, we will get an “a” as in “sat,” which is perceived quite differently. After plotting all of the vowels, we see that vowels form regions in the space of of formant frequencies. We control those frequencies by changing the shape of our vocal tract to make vowel sounds.



Images from: duPont Hospital for Children and the University of Delaware, *Speech Tutorials,* http://www.asel.udel.edu/speech/tutorials

Figure 1: Diagram of a human head to illustrate how the voice works.

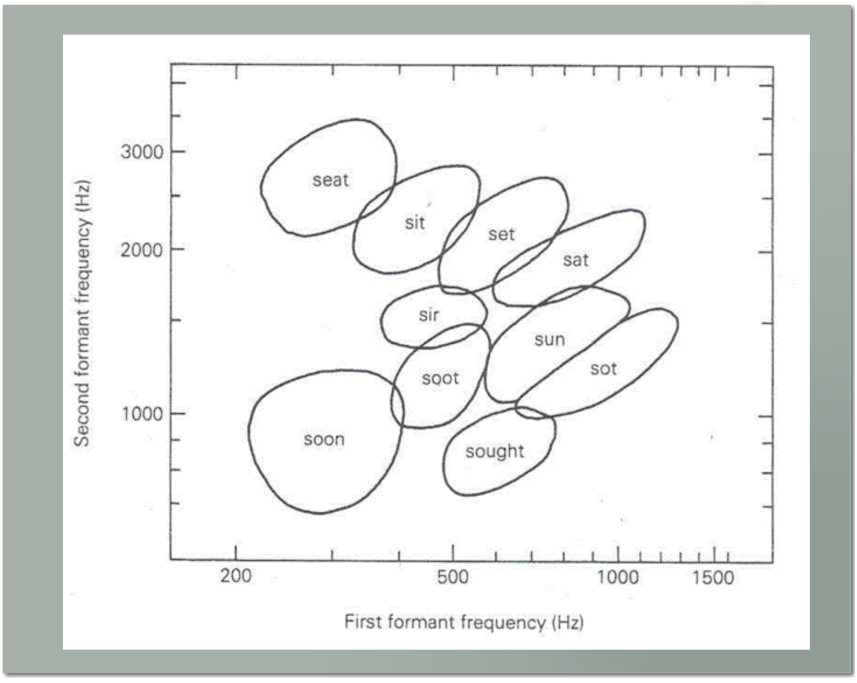


Figure 2: Frequency plots for vowels. (from http://www.phy.davidson.edu/fachome/dmb/digitalspeech/formantmap.gif)

1.2 Source-Filter Models

The above mechanism leads to a very common and simple model which we can build for voice synthesis. It is called the *source-filter model*, which describes sound production as a *source* (e.g. the vocal folds and their production of pulses, as shown in Figure 1) and a *filter*, which is mainly the effect of resonances (formants) created by the vocal tract (see Figure 3).

The *source* in most voice synthesis systems is actually a choice of either a noise signal (whispered / unvoiced sound, where the vocal folds are not vibrating, used for sounds like "sss"), or a pulse train of periodic impulses creating "voiced" speech as is normal for vowels. The selected source input goes into vocal tract filter, which is characterized by the resonances or formants that we saw in the previous paragraphs.

Notice that the frequency of the source (i.e. the rate of pulses) determines the pitch of the voiced sound, and this is independent of the resonant frequencies (formant frequencies). The source spectrum for voiced sounds is a harmonic series with many strong harmonics arising from the periodic pulses. The effect of filtering is to multiply the spectrum by the filter response, which boosts harmonics at or near the formant frequencies.

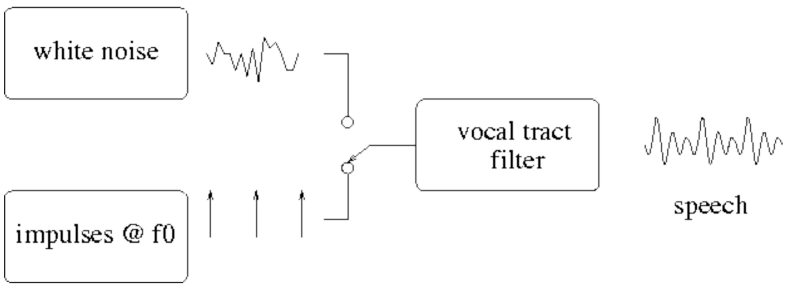


Figure 3: Framework of source filter models. (From Tony Robinson, *Speech Analysis*, http://svr-www.eng.cam.ac.uk/~ajr/SA95/SpeechAnalysis.html)

2 Linear Prediction Coding (LPC)

In the previous section, we have learned how the source-filter model is used to model voice and produce voice sounds. Now we are going to talk about a specific speech analysis/synthesis method that can be used to construct the filter part of the source-filter model. This method is called *Linear Prediction Coding* (LPC).

2.1 LPC Basics

The basic model of LPC is shown as the simple equation below.

$$S_n = \sum_{i=1}^{p} a_i S_{n-i}.$$

It means predicting next sample S_n as a weighted sum of past samples, where the weights are arbitrary a_i, previous samples are S_{n-i}, and p is the number of previous samples that we look at. This formulation gives rise to an *allpole* filter, in other words, the weighted sum of the samples is the filter, and the response of this filter consists of resonant peaks. We can analyze real signals to estimate the a_i; LPC analysis finds the filter that best approximates the signal spectrum. We typically choose p to be a somewhat small value, normally less than 20, so that the filter response can at best be modeled as a smooth function with a few peaks (resonances). If instead we use hundreds of coefficients, the analysis might treat each harmonic as a very sharp resonance, but we prefer not to allow this since in the source-filter model, the harmonics are produced by the source. Thus, we try to restrict the detail in the filter approximation

so that it captures the fairly broad and smooth resonant formants, but not the exact wave shape or other details that come from the spectrum of the source.

2.2 LPC Analysis

In LPC analysis, we divide an input signal into short segments, and we estimate the a_i coefficients that minimize the overall error in predicting the S_n samples of the signal. We will skip the details of this optimization problem.

As shown in Figure 4, the physical analogy of LPC is a tube with varying cross-section. Think about the vocal tract: the vocal folds inject pulses into a tube, which is the throat, mouth and all the way to the mouth opening. It is not a simple tube and there is also the nasal cavity that makes a difference in the sound of your voice. (If you hold your nose and talk, you can hear the importance of the nasal cavity!) However, here we only consider a simple model: putting the sound through the tube and varying the cross-section of the tube in the same way that your tongue and mouth are varying the cross-section of your vocal tract.

Figure 4: The physical analogy of LPC: a tube with varying cross-section.

Figure 5 shows another example where the vocal tract is modeled as a tube. We start with the real vocal tract colored red. Then, we construct a tube (on the right) by cutting layers of plastic such that the cross-section of the tube matches the cross-section of the human vocal tract. If we put a source (such as a duck call) up to this tube and listen to what comes out, the sound will be the "AH" vowel, matching the human's sound.

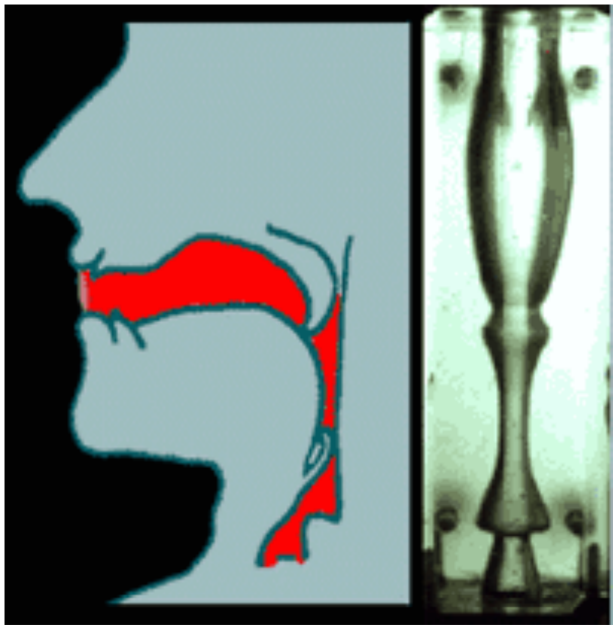


Figure 5: Acoustic Tube Producing "AH". [1]

Another important point here is that we are not talking about a fixed shape but a shape that is rapidly changing, which makes speech possible. We are making sequences of vowel sounds and consonants by rapidly moving the tongue and mouth as well as changing the source sound. To apply LPC to speech, we analyze speech sounds in

short segments (and as usual, we call short analysis segments *frames*). This is analogous to short-time windows in the SFFT. At each LPC analysis frame, we re-estimate the geometry of the tube, which means we re-estimate the coefficient values that determine the filter. Frames give rise to changing coefficients, which model changes in tube geometry (or vocal tract shape).

LPC actually creates an inverse filter. Applying the inverse filter to a vocal sound yields a *residual* that sounds like the source signal. The residual may either be an estimate of glottal pulses, making the residual useful for estimating the pitch of the source, or noise-like.

Taking all this into account, an overall LPC analysis system is shown in Figure 6. We have an input signal and do formant analysis, which is the estimation of coefficients that can reconstruct the filter part of the source filter model. After applying the result filter of the formant analysis to the input signal, we can get the residual and then do further analysis to detect the pitch. We can also do analysis to see if the input is periodic or not so that it gives us a decision of whether the input is voiced or unvoiced. Finally, we can compute the RMS amplitude of the signal. So, for each frame of LPS analysis, we get not only the filter part of the source filter model, but also estimated pitch, the amplitude, and whether the source is voiced or unvoiced.

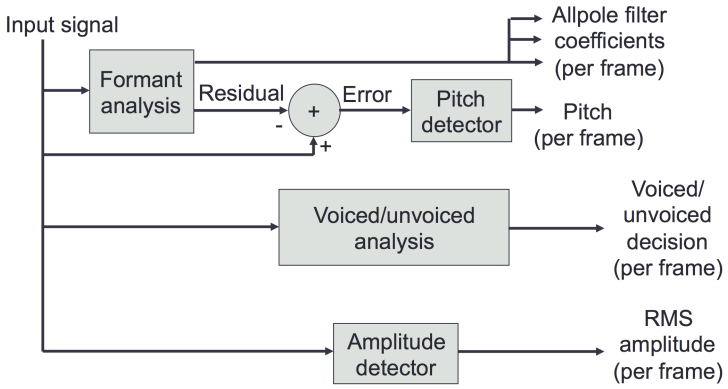


Figure 6: Diagram of LPC.

2.3 Musical Applications

There are many musical applications of voice sounds and source-filter models. One common application pursued by many composers is to replace the source with some other sound. For example, rather than having a pitched source or noise source based on the human voice, you can apply human sounding formants to a recording of an orchestra to make it sound like speech. This idea of taking elements from two different sounds and putting them together is sometimes called "cross-synthesis." You can also "warp" the filter frequencies to get human-like but not really human filters. Also, you can modify the source and LPC coefficients (glottal pulses or noise) to perform time stretching, slowing speech down or speeding up.

3 Vocoder

The Vocoder is related to LPC in that it makes a low-dimensional (i.e. smooth, very approximate) estimation of the spectral shape. Thus, formants can be captured and modeled. The main idea of the vocoder is that the input signal is divided into frequency "channels" and bandpass filters are used to isolate each channel and measure the amplitude in that channel. An FFT can be used since it acts as a bank of bandpass filters, but vocoders are also implemented in analog hardware with electronic filters, or you could use a bank of digital bandpass filters instead of the FFT.

Vocoders are often used for cross-synthesis where the formants are estimated using one sound source (the "modulator") and then applied to a second sound source (the "carrier"). For example, one could analyze a voice

to extract formants and apply them to chords played on a keyboard. Figure 7 illustrates a vocoder in this cross-synthesis application. The "ENV Follow" box takes a filtered signal and extracts an amplitude "envelope" that is used to modulate the amplitude of the carrier using a "VCA"-voltage-controlled amplifier. Only two frequency bands are shown, but a typical vocoder (even in hardware) would have 8 or more channels.

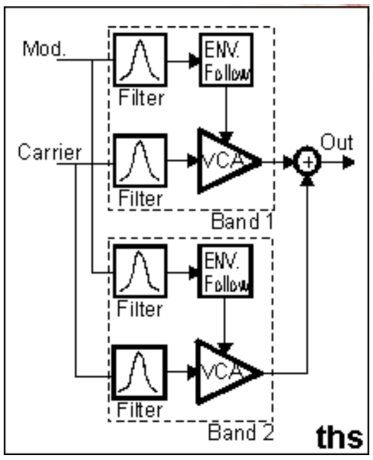


Figure 7: Vocoder

4 VOSIM

VOSIM is a simple and fun synthesis method inspired by the voice. VOSIM is also fast and efficient. It was developed in the 70s by Kaegi and Tempelaars. The basic idea of VOSIM is to think about what happens in the *time domain* when a glottal pulse (pulse coming through the vocal folds) hits a resonance (a formant resonance in the vocal tract). The answer is that you get an exponentially damped sinusoid. In speech production, we have multiple formants. When a pulse enters a complex filter with multiple resonances, we get the superposition of many damped sinusiods, one for each resonance. The exact details are influenced by the shape of the pulse. VOSIM is the first of several time-domain models for speech synthesis that we will consider.

4.1 VOSIM Parameters

VOSIM uses a pulse train of sin^2 pulses (that is not a footnote, it means we *square* the sinusoid. Doing so doubles the pulse frequency and makes the signal non-negative.) The pulses diminish in amplitude. Figure 8 shows one period of VOSIM. The parameters of VOSIM are: an initial amplitude A, a period T, a decay factor b (not a continuous decay but each pulse is the previous one multiplied by a factor b), the number of pulses N, and a duration of M silence that follows the pulses. As you can see in Figure 8, the result is at least *somewhat* like a decaying sinusoid, and this gives an approximation of a pulse filtered by a single resonance. The fundamental frequency (establishing the pitch) is determined by the entire period, which is $NT + M$ and the resonant frequency is $1/T$, so we expect the harmonics near $1/T$ to be stronger.

4.2 VOSIM Application

In the original VOSIM paper, Tempelaars used various "delta" or "increment" parameters so that the continuous parameters T, M, and b could change over time. This was probably just a simplification over using more general envelopes to control change over time. The `vosim` extension in Nyquist (open the Extension Manager in the NyquistIDE) contains some code and examples.

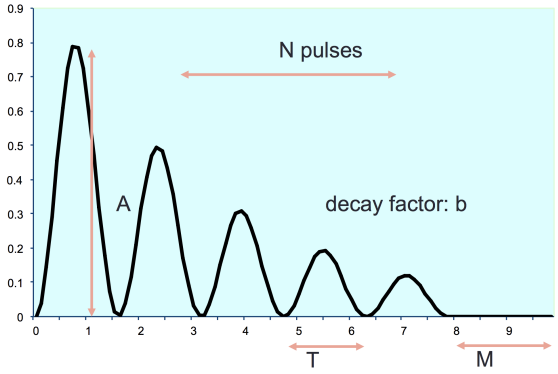


Figure 8: Diagram of VOSIM.

5 FOF Synthesis

FOF is a French acronym for "Formant WaveFunction" (Synthesis). Like VOSIM, FOF is a time-domain approach to creating formants and speech-like sounds. Compared to VOSIM, FOF can be seen as a more sophisticated and more accurate model of a pulse train filtered by a resonant filter. Instead of the sine-squared pulses of VOSIM, FOF uses a sinusoid with an exponential decay (see Figure 9), and instead of just N pulses of the decaying resonant frequency, FOF allows the sinusoid to decay to a very small amplitude. This might result in overlapping decaying sinusoids (recall that in VOSIM, N decaying pulses are followed by M seconds of silence before the next pulse, so pulses do not overlap.) This makes the implementation of FOF more difficult, and we will describe the implemenation below.

One final detail about FOF is that the decaying sinusoid has an initial smooth rise time. The rise time gives some additional control over the resulting spectrum, as shown in Figure 10.

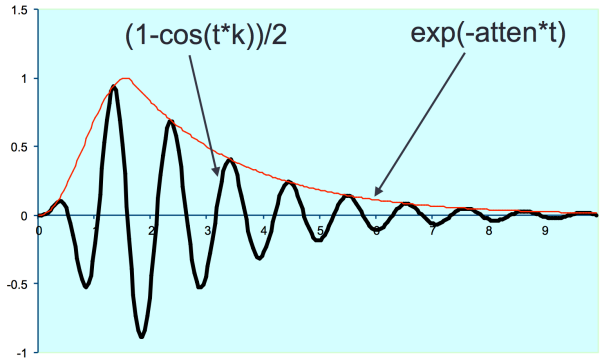


Figure 9: FOF

5.1 FOF Analysis

One of the advantages of FOF synthesis is that FOF parameters can be obtained automatically. The basic approach is to analyze the spectrum of a sound with an FFT. Then, analyse the shape of the FFT to detect formants. Use a FOF generator for each formant. If the spectrum indicates many harmonics, the first harmonic or the spacing between harmonics can be used as the fundamental period.

Since analysis can be performed using any spectrum, FOF is not limited to voice synthesis, and FOF has been used for a wide range of sounds.

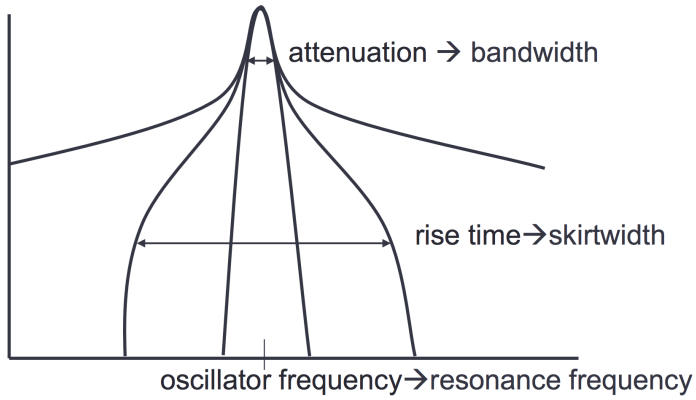


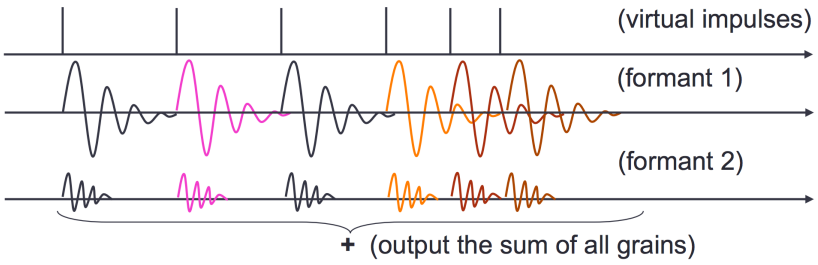
Figure 10: FOF in the frequency domain.

5.2 FOF Synthesis

Consider the FOF signal shown in Figure 9. This represents the response of a single formant to a single impulse – essentially a grain. Let's assume we have a *FOF generator* that can create this grain, probably using an envelope generator for the onset, a sine oscillator for the main signal, an exponential decay generator, and multipliers to combine them.

To model a periodic pulse, we need a pool of these FOF generators. We allocate one for each formant at every pulse period and release the generators back to the pool when their output decays to a negligible value. The size of the pool depends on the amount of overlap (longer decays and higher fundamental frequencies mean more overlap), so the implementation may need to make the pool size dynamically expandable or simply allocate every FOF generator from the general memory heap.

Figure 11 illustrates a collection of FOF "grains:" at each pulse period (the fundamental frequency), there is one grain for each formant. The overlapping grains from FOF generators are summed to form a single output signal.



Example: Jean-Baptiste Barriere, *Chreode I,*
`https://www.youtube.com/watch?v=5AEFhybYrPg#t=128`

Figure 11: FOF synthesis.

6 Phase Vocoder

The Phase Vocoder is not really a speech synthesis method but it is often used to manipulate speech signals. The main assumption of the Phase Vocoder is that input sounds consist of fairly steady sinusoidal partials, there are relatively few partials and they are well-separated in frequency. If that is the case, then we can use the FFT to

separate the partials into different bins (different coefficients) of the discrete spectrum. Each frequency bin is assigned an amplitude and phase. A signal is reconstructed using the inverse FFT (IFFT).

The main attraction of the phase vocoder is that by changing the spacing of analysis frames, the reconstructed signal can be made to play faster or slower, but without pitch changes because the partial frequencies stay the same. Time stretching without pitch change also permits pitch change without time stretching! This may seem contradictory, but the technique is simple: play the sound faster or slower (by resampling to a new sample rate) to make the pitch go up or down. This changes the speed as well, so use a phase vocoder to change the altered speed back to the original speed. Now you have pitch change without speed change, and of course you can have any combination or even time-variable pitch and speed.

When successive FFT frames are used to reconstruct signals, there is the potential that partials in one frame will be out of phase with partials in the next frame and some cancellation will occur where the frames overlap. This should not be a problem if the signal is simply analyzed and reconstructed with no manipulation, but if we want to achieve a speed-up or slow-down of the reconstructed signal, then the partials can become out of phase.

In the Phase Vocoder, we use phase measurements to shift the phase of partials in each frame to match the phase in the previous frame so that cancellation does not occur. This tends to be more pleasing because it avoids the somewhat buzzy sound of partials being amplitude-modulated at the FFT frame rate. In return for eliminating the frame-rate discontinuities, the Phase Vocoder often smears out transients (by shifting phases more-or-less randomly), resulting in a sort of fuzzy or smeared quality, especially if frames are long. As we often see (and hear), there is a time-frequency uncertainty principle at play: shorter frames give better transient (time) behavior, and longer frames give better reproduction of sustained timbres due to better frequency resolution.

7 MacAulay-Quatieri (MQ) Synthesis

MacAulay-Quatieri developed a speech compression technique where the signal is decomposed into *time-varying* as well as amplitude-varying sinusoids. This differs from the phase vocoder mainly in that with the phase vocoder, we assume all sinusoids are fixed in frequency and match the FFT bin frequencies.[2] In contrast, MQ analysis tracks peaks from analysis frame to analysis frame and creates a description in which sinusoids can change frequency dramatically. The analysis for MQ is quite different from phase vocoder analysis. Reconstruction is also different and relies upon a bank of sinusoid oscillators with frequency and amplitude controls. This is just additive synthesis using sinusoids.

With MQ's additive-synthesis representation, time stretching and frequency shifting are easily accomplished just by scaling the amplitude and frequency control functions. The reconstruction does not involve the IFFT or overlapping windows, so there are no problems with frames or phase discontinuities, i.e. no "buzz" at the frame rate.

Some limitations of MQ are that it takes large numbers of sinusoids to model noise sounds. Also, resonances are not modeled (similar to phase vocoder), so one cannot easily treat resonance details as control parameters. Because it uses additive synthesis, MQ is not particularly efficient of flexible. With source-filter models, we saw how one could replace the source to create a "cross-synthesis" model, but there is no obvious way to achieve cross-synthesis with MQ.

8 Spectral Modeling Synthesis (SMS)

Spectral Modeling Synthesis (SMS) was developed by Xavier Serra in his Ph.D. thesis work and continued in the Music Technology Group, Audiovisual Institute, Pompeu Fabra University, Barcelona. SMS extends MQ analysis/synthesis with an explicit model for noise. The model is shown in Figure 12.

You can see the box labeled "additive synthesis" and this essentially outputs a resynthesis of the input signal using MQ analysis/synthesis. Rather than stopping there, SMS subtracts this resynthesis from the input signal to obtain a *residual* signal. Assuming that most of the spectral peaks representing sinusoidal partials have been

[2]This is a bit of over-simplification. In fact, the phase change from frame to frame in the phase vocoder indicates small deviations from the center frequency of each bin, so the phase vocoder, when properly implemented, is modeling frequencies accurately, but the model is still one where partial frequencies are modeled as fixed frequencies.

modeled, the subtraction removes them and the residual should contain mostly noise. The "spectral fitting" stage constructs a low-dimensional (i.e. few coefficients) description of the noise spectrum. SMS considers the sinusoidal tracks to be the "deterministic" part of the signal, and the residual noise spectrum to be the "stocastic" part of the signal.

Resynthesis is accomplished by additive synthesis of the sinusoidal tracks (as in MQ synthesis) *and* adding filtered noise, using the "stocastic" coefficients to control the noise filter. As with many synthesis techniques, one can time stretch and manipulate all the control parameters. The noise modeling in SMS allows for more accurate reconstruction as well as control over how much of the noise in the signal should be retained (or for that matter emphasized) in synthesis.

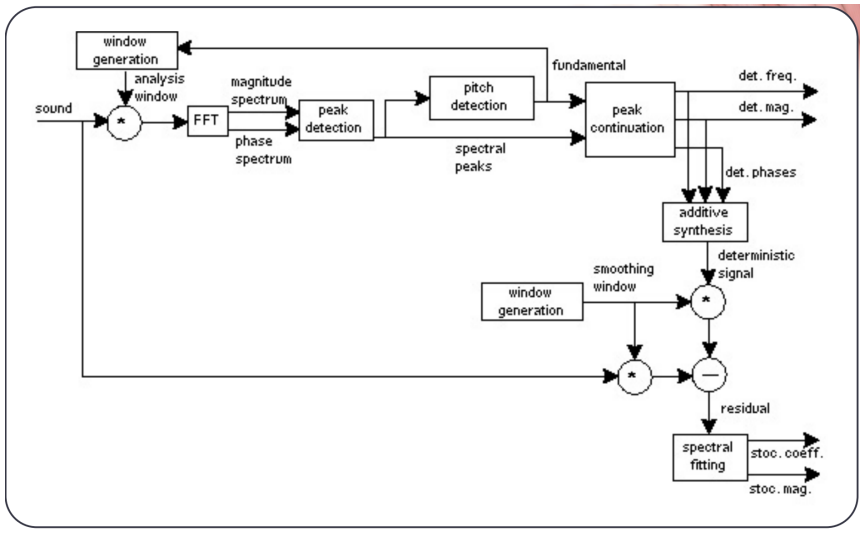


Figure 12: SMS

9 Summary

We have considered many approaches to voice modeling and synthesis. The number of methods is some indication of the broad attraction of the human voice in computer music, or perhaps in all music, considering the prevalence of vocals in popular music. The source-filter model is a good way to think about sound production in the voice: a source signal establishes the pitch, amplitude and harmonicity or noisiness of the voice, and more-or-less independently, formant filters modify the source spectrum to impose resonances that we can perceive as vowel sounds.

LPC and the Vocoder are examples of models that actually use resonant filters to modify a source sound. These methods are open to interesting cross-synthesis applications where the source sound of some original vocal sound is replaced by another, perhaps non-human, sound.

Resonances can also be modeled in the time domain as decaying sinusoids. VOSIM and FOF synthesis are the main examples. Both can be seen as related to granular synthesis, where each source pulse drives a resonance to produce a new grain in the form of a decaying sinusoid.

In the spectral domain, the Phase Vocoder is a powerful method that is especially useful for high-quality time-stretching without pitch change. Time-stretching also allows us to "undo" time stretch due to resampling, so we can use the Phase Vocoder to achieve pitch shifting without stretching. MQ analysis and synthesis models sounds as sinusoidal tracks that can vary in amplitude and frequency, and it uses addition of sinusoidal partials for synthesis. Spectral Modeling Synthesis (SMS) extends MQ analysis/synthesis by modeling noise separately from partials, leading to a more compact representation and offering more possibilities for sound manipulation.

Although we have focused on basic sound production, the voice is complex, and singing involves vibrato, phrasing, actual speech production beyond vowel sounds, and many other details. Undoubtedly, there is room for

more research and for applying recent developments in speech modeling to musical applications. At the same time, one might hope that the speech community could become aware of sophisticated control and modeling techniques as well as the need for very high quality signals arising in the computer music community.

10 Acknowledgments

Thanks to Shuqi Dai and Sai Samarth for editing assistance.

Portions of this work are taken almost verbatim from *Music and Computers, A Theoretical and Historical Approach* (http://sites.music.columbia.edu/cmc/MusicAndComputers/) by Phil Burk, Larry Polansky, Douglas Repetto, Mary Robert, and Dan Rockmore. Other portions are taken almost verbatim from *Introduction to Computer Music: Volume One* (http://www.indiana.edu/~emusic/etext/toc.shtml) by Jeffrey Hass. I would like to thank these authors for generously sharing their work and knowledge.