UNIVERSITY OF CALIFORNIA
Santa Barbara

Toward Timbral Synthesis: a new method for synthesizing sound based on timbre description schemes

A Thesis submitted in partial satisfaction
of the requirements for the degree of

Master of Science

in

Media Arts and Technology

by

Daniel Mintz

Committee in Charge:

Professor Curtis Roads, Chair

Xavier Amatrian, Ph. D.

Professor JoAnn Kuchera-Morin

Professor John Hajda

June 2007

The Thesis of
Daniel Mintz is approved:

Xavier Amatrian, Ph. D.

Professor JoAnn Kuchera-Morin

Professor John Hajda

Professor Curtis Roads, Committee Chairperson

June 2007

Toward Timbral Synthesis: a new method for synthesizing sound based on timbre description schemes

by

Daniel Mintz

Acknowledgements

First and foremost, I must express my deep gratitude to Xavier Amatriain. His guidance, support, knowledge, and mentoring were invaluable through every step of this process. This thesis simply would not have been possible without his assistance.

I would also like to thank:

—the other members of my thesis committee, John Hajda, JoAnn Kuchera-Morin, and Curtis Roads, for their input, their support, their critical feedback, and their patience.

—Alex Norman for his numerous insights and for always being there.

—Phil Coakley, Randall Farmer, Angus Forbes, and Will Wolcott for their crucial help working out various equations, programming problems, and other obstacles at every stage of my work.

—the staff of MoveOn, for graciously giving me the time away I needed to finish my thesis.

—Eileen Koven, for being my partner in navigating the sometimes labyrinthine thesis process.

—Professor Subhash Suri, for an instructive meeting on linear programming software.

—my friends, Charlie, Eric, Jorge, Haru, Laura, Sarah, Melissa, Basak, Melissa, Bob, Wes, Bo, Erin, August, FSI, and Antonio, for getting me through good times, bad times, and all the times in between.

—finally, my family—Mom, Dad, and Jenny—for their love, encouragement, and proofreading.

Abstract

Toward Timbral Synthesis: a new method for synthesizing sound based on timbre description schemes

Daniel Mintz

This thesis proposes a novel approach to sound synthesis—timbral synthesis—that allows users to precisely and predictably generate sounds having the timbres they desire. By relying on timbral descriptors derived from extensive psychoacoustic research and incorporated into the MPEG-7 timbre description scheme, timbral synthesis focuses on those aspects of sound that are most salient in our perception of timbre. Users specify the timbre they want using standardized descriptors that affect spectral shape, amplitude envelope, and spectro-temporal variation, and the timbral synthesis engine converts these descriptor values into control envelopes by constructing and solving linearly-constrained optimization problems. The paper consists of a full exposition of the theory behind timbral synthesis and a description of a first implementation.

Professor Curtis Roads
Thesis Committee Chair

Contents

Chapter 1

Introduction

The production of musical sound involves control over five largely orthogonal dimensions: pitch, loudness, duration, spatialization, and timbre. And while musicians and composers have become quite skilled at manipulating the first four parameters, broadly categorizing and controlling the fifth, timbre, remains difficult. This paper proposes a method for sound synthesis that gives the user direct control over this crucial dimension. Most other synthesis methods are defined in terms of their signal inputs (e.g. additive synthesis, frequency modulation, granular synthesis, physical modeling), addressing timbre primarily as the result of these inputs. The approach proposed here *begins* with the specification of the desired timbre and then determines the inputs on the basis of the requested timbre. Specifically, it lets the user precisely define how the result should sound according to seven standardized descriptors, all of which have been experimentally shown to influence how timbre is perceived.

Timbre, defined as “the characteristic quality of a sound, independent of pitch and loudness, from which its source or manner of production can be inferred” [8], is, in essence, everything about a sound that doesn’t fit well into the other four dimensions. As is clear from the imprecision of this definition, timbre and our perception of timbre are less well understood than the other dimensions. Because timbre is inextricably tied to the production mechanism in acoustic instruments, timbral control is limited primarily by the physicality of the instrument. In other words, a violin must sound like a violin, and not a piano, because it is handheld and has four strings controlled by pressure on a fingerboard rather than freestanding with more than 200 strings actuated by hammers.

Beyond the physical constraints on an acoustic instrument’s available timbres, the complexity of timbre and its high dimensionality as compared to pitch, loudness, duration, and spatialization place higher demands on the musician for control. Thus, although the advent of synthetic sound production offered the possibility of an infinitely reconfigurable instrument that could fulfill any timbral demand, the problem of control only intensified. Max Mathews, an early innovator in the computer generation of sound, wrote, “There are no theoretical limitations to the performance of the computer as a source of musical sounds, in contrast to the performance of ordinary instruments” [24, pg. 553].

Yet, in the same article, Mathews recognizes that "the range of computer music is limited principally by cost and by our knowledge of psychoacoustics"[24, pg. 553]. Several years later, Mathews went further, identifying "the two fundamental problems in sound synthesis" as "(1) the vast amount of data needed to specify a pressure function—hence the necessity of a very fast program—and (2) the need for a simple, powerful language in which to describe a complex sequence of sounds"[25, pg. 34]. In the years since Mathews wrote those words, researchers have made great strides in addressing the first issue he raised, but still struggle with the second.

Thus, the problem that remains unsolved is how to offer computer musicians a synthesis tool that (1) can create a wide variety of sounds, (2) is controlled by a manageable interface, and (3) offers perceptually meaningful control parameters. Such a tool would allow composers to quickly and reliably translate sounds they imagine into sounds they can work with—manipulating timbre as easily as they do pitch, loudness, duration, and, to some degree, spatialization. Because of the significant advantages in terms of timbral variety that sound synthesis offers over acoustic instruments, solving this problem would represent a major advance in how composers could use timbre in their works.

The many synthesis tools currently available excel differently in terms of the three criteria sought, but none is ideal in all three areas. The implementation of timbral synthesis proposed in this paper is a novel approach that shows promise on all three of the above tests. Minimal further work developing application-specific implementations could yield powerful tools for musicians and researchers alike. And because our understanding of the psychoacoustics of timbre is continually growing, the fact that timbral synthesis exploits that understanding positions it to improve and grow along with the research that underlies it.

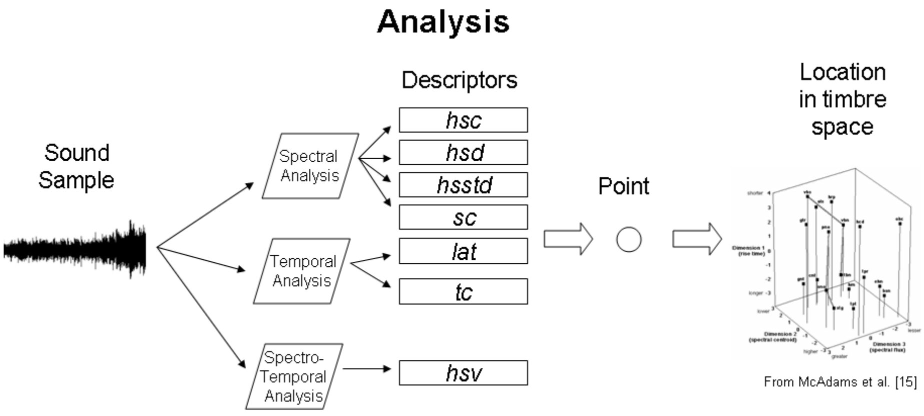
1.1 Overview

Today, as Mathews predicted, our understanding of timbre, while still incomplete, has grown enormously, and our computers have grown exponentially more powerful. Taking advantage of this fuller understanding of timbre and the increasing ability of computers to complete complex calculations in real or near real-time, this paper proposes a method for timbral synthesis based directly on the current understanding of how we hear sound. By allowing the user direct control over those physical properties of sound that psychoacoustic studies have shown to be most perceptually salient, this method of timbral synthesis gets at the heart of the problem of how to control timbre.

Rather than treating timbre as a byproduct of the synthesis process, timbral synthesis, as the name implies, elevates timbral control to the position of primary importance. Since increased timbral range is the major advance that synthetic sounds offer over natural sounds, treating timbral control as the point, rather than as a side effect, seems worthwhile. In order to do this, timbral synthesis unravels the multidimensional construct of timbre and gives users direct control over those dimensions that shape what we hear.

In short, timbral synthesis lets users precisely define the timbre (as well as the pitch, duration, loudness and, conceivably, spatialization) of the tone they want. Timbre is specified according to its salient attributes as determined by advanced psychoacoustic research. Once set in motion, timbral synthesis quickly identifies, generates, and outputs a tone that matches the one requested by the user exactly. In terms of the criteria posited above, by using timbre spaces that theoretically encompass all harmonic and percussive tones, timbral synthesis is capable of producing a wide variety of timbres; by limiting the number of control parameters (because research has shown that there are only a handful of perceptually meaningful physical properties of sound that affect timbre), timbral synthesis presents a highly manageable interface to the user; and by basing the controls on models developed from perceptual experiments, timbral synthesis ensures that the controls will map well to the results the user hears.

In its current form, timbral synthesis reverse engineers the analysis tools developed for the MPEG-7 timbre description scheme and repurposes them as synthesis



From McAdams et al. [15]

Figure 1.1: A graphical overview of the timbral analysis process

tools. Timbral analysis takes pre-existing sounds and breaks down their spectral, temporal and spectro-temporal envelopes into a series of descriptor values; timbral synthesis begins with those descriptor values and generates envelopes for a sound that matches. In terms of timbre spaces, timbral analysis decomposes a sound into discrete dimensions to locate it in a multidimensional space (Figure 1.1); timbral synthesis locates a point in the timbre space and composes a sound located at that point (Figure 1.2).

Past implementations of timbre-based synthesis have generally taken one of two directions. Some, like Wessel [47] and Hourdin et al. [16], have worked with a reduced set of descriptors, precisely defining the sound by mapping each descriptor to a single physical parameter of the synthesized tone. This approach is limited by the low dimensionality of its representation of timbre, oversimplifying what we

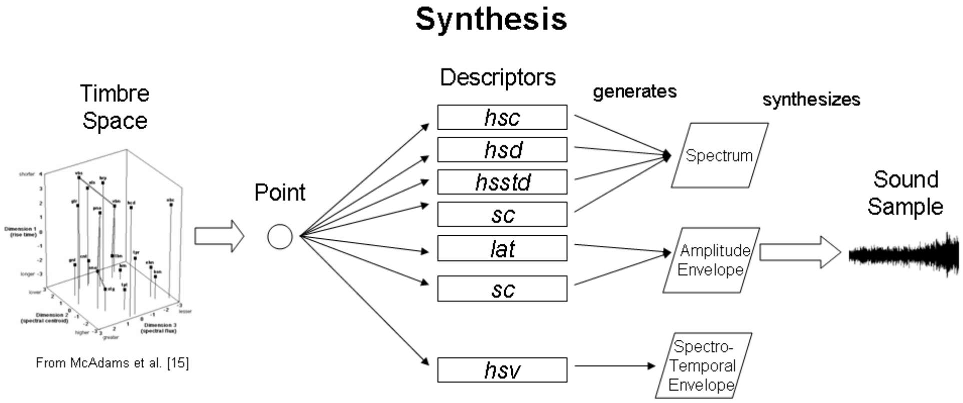


Figure 1.2: A graphical overview of the timbral synthesis process

know to be a highly complex phenomenon. Others, including Jehan [18] and Le Groux [22], have used machine learning methods to "teach" the synthesis engine what different timbres sound like. From this teaching set, these synthesizers can then intuit what attributes new sounds would have in relation to the learned sounds. By avoiding the complexity that comes with trying to identify particular dimensions of timbre, machine learning implementations let the computer figure out the mappings in the background, but consequently lacked the precision of the former method.

The type of timbral synthesis defined in this paper uses a new method, relying on powerful linear programming solvers to grapple with precise timbral definitions in highly-dimensional timbre spaces. By formulating the synthesis parameters as a linearly constrained maximization problem, timbral synthesis can produce tones that correspond exactly to the descriptor values specified by the user. This method offers a possible way forward for timbral synthesis that is precise, efficient, and in line with our increasingly complex and nuanced understanding of timbre.

By allowing the user to directly specify timbre, this implementation of timbral synthesis divorces the other dimensions of a sound (pitch, loudness, duration, and spatialization) from the synthesis process. This means that timbre can theoretically be manipulated orthogonally from these other dimensions in a way that is difficult to achieve using other synthesis techniques.[1] Thus, because loudness can be specified independently from other timbral synthesis parameters, one can vary the brightness of the tone without having to worry that the addition of higher-order harmonics will increase the overall loudness, as one would using additive synthesis, for example.

One final benefit that timbral synthesis offers in comparison to other synthesis methods is that, in addition to control parameters being perceptually meaningful, as determined by psychoacoustic experiments, changes in these parameters can be effected on a subjective scale that coincides with our perception. In synthesis using frequency modulation, for example, one often makes a small change to a control parameter and gets a perceptually small change in the resulting sound, then makes some other equally small change to another control parameter, only

[1] The qualification “theoretically” is necessary because numerous interrelations between the different dimensions belie their orthogonality. Even so, treating these dimensions as largely distinct from each other is useful in thinking about how we can control sound production mechanisms.

to get a drastic shift in timbre. Because numerous experiments have shown not only which physical properties of sound are salient to timbre, but also the relative salience of different properties, users can expect a more predictable response when they “twiddle the knob” of their timbral synthesizer than they can with other synthesizers.

1.2 Motivations

The problem of defining and controlling timbre remains an important one. Players of acoustic instruments have long understood how to get the sound they want out of their instrument, but the range of timbres that most of these instruments can produce is extremely limited. Electroacoustic instruments would seem to provide a solution to this problem, but decades of experience have shown us that inherent in this “solution” are many problems. In many cases, the current crop of sound synthesis methods leaves users casting about for interesting timbres while the task of turning a specific, imagined sound into a realized, controllable, synthesized sound often remains arduous.

Thus, a desirable instrument would afford its user perceptually meaningful control over a limited number of dimensions but would still be able to produce a wide variety of sounds. Ideally, one could think up any sound, set a few control parameters, and obtain that sound. One major obstacle to creating this ideal instrument lies in our still incomplete understanding of timbre. Yet, as we have begun to better understand the relationship between the physical characteristics of sound waves and the perceived timbre of that sound, new, more direct controls of timbre have become possible.

The timbre description scheme in MPEG-7 represents an important point of consensus about how we hear and interpret timbre. As such, putting its timbral descriptors to work for sound synthesis presents an enticing opportunity. Timbral synthesis, as proposed in this paper, is not the final solution to the problem of timbral control. But hopefully, it will prove to be an important addition to the methods for timbral control already available to researchers and musicians.

Chapter 2

Related Work

Before describing the specific theory of timbral synthesis that forms the heart of this paper, it is useful to first review both the existing literature in the field and the tools used in the project. This chapter focuses on describing several important sound synthesis methods in terms of their capacity for timbral control, reviewing our current understanding of timbre, and explaining the basics of the MPEG-7 timbre description scheme.

2.1 Synthesis Methods

Different synthesis methods address the problem of timbre control differently, but in general, the tradeoff that defines all solutions to this problem is one of control versus manageability. Synthesis methods that offer highly refined timbre control usually either force users to restrict the range of timbres that can be produced or grapple with unmanageably large demands for control data. Synthesis methods that offer manageable timbre control over a broad range of timbres, on the other hand, offer only very coarse or unpredictable control over the timbres produced.

2.1.1 Additive Synthesis

The most basic form of additive synthesis, in which individual sine wave oscillators are combined to produce sound, provides the highest level of control of any synthesis method. As Curtis Roads points out in *The Computer Music Tutorial*, at the most fundamental level, a composer can control the individual frequency and amplitude envelopes for each oscillator for each event. This control provides a level of precision and predictability that is unmatched by any other synthesis method. For any instant in the piece, the composer can determine the exact spectrum being produced by simply finding the frequency value of each oscillator, multiplying that output by each oscillator's amplitude envelope, and summing the resulting values. This can be represented mathematically by the following equation:

$$x(t) = \sum_{k=1}^{k_{max}} a_k(t) cos\left(\omega_k t + \phi_k\right)$$

where t is time, k is the number of the partial, a_k is the amplitude of the partial, ω_k is the radial velocity (frequency) of the partial, and ϕ_k is the phase offset of the partial.

However, this extreme level of control means that the number of control values the composer must specify quickly become unmanageable. For example, “if a piece contains 10,000 events (such as a typical orchestral score), each with up to 24 partials, one needs to have 240,000 frequency envelopes and 240,000 amplitude envelopes on hand” [38, pg. 143]. Producing sounds with any timbral complexity requires an enormous amount of work. And while there are certainly specific timbres that can best be achieved using pure additive synthesis, the unmanageable quantity of control values that a work of any length requires means that few composers use additive synthesis without a means for improving manageability.

Other than pure additive synthesis methods, there are also hybrid methods that combine additive synthesis with other types of synthesis or that offer higher-level controls to make the synthesis process more manageable. Xavier Serra proposes a method that does both—combining additive synthesis with subtractive synthesis and using FFT analysis to control many of the parameters so that the user doesn’t have to. The method described by Serra uses additive synthesis to model the deterministic parts of a sound (i.e. the stable harmonic components of the spectrum being resynthesized). But the residual parts of the sound (those that remain after the deterministic parts have been removed) are modeled using broadband noise shaped by a filter. This model requires far less user input than a pure additive synthesis instrument, but is largely restricted to resynthesizing or transforming existing sounds. That said, Serra also envisions using this “sinusoids plus noise” method to model whole families of timbres, like the full range of an instrument, by analyzing a selected sample of sounds produced by that instrument. Even in this application though, the advantage of better manageability is only conferred when the user remains within the bounds of pre-existing, analyzed sounds [39].

Serra’s synthesis method, dividing the harmonic and inharmonic spectrum up and synthesizing them separately, using different synthesis techniques, is based on work he did with Julius O. Smith. In Smith and Serra’s original work, only the most prominent peaks from an FFT analysis are resynthesized [42]. With the addition of broadband, filtered noise, Serra’s technique has further applications, including the synthesis portion of timbral synthesis as described in this paper. By synthesizing the harmonic and inharmonic spectra separately, more nuanced control schemes that are sensitive to the particular demands of the different spectra can be developed. As we will see, it is in this vein that timbral synthesis addresses the synthesis of harmonic and inharmonic spectra separately so that the user can exert meaningful control on each.

2.1.2 Frequency Modulation

Another common synthesis technique is frequency modulation (FM). By modulating the frequency of one oscillator with the output of another, composers can create highly complex spectra and thus, highly complex timbres. FM synthesis falls on the opposite side of the control versus manageability spectrum. In the simplest FM system, one need only specify an amplitude and frequency envelope for the modulating oscillator and an amplitude envelope for the carrier oscillator. With just these three control envelopes, FM synthesis can generate a complex spectrum—a spectrum that would require many times more control envelopes using additive synthesis. Furthermore, more complicated implementations of FM, involving multiple carriers, multiple modulators, feedback loops and the summation of the outputs of several FM systems, can produce immensely complicated spectra with hundreds or thousands of partials.

However, with this increase in manageability comes an accompanying decrease in control. Producing a desired spectrum often requires either extensive calculations or extensive experimentation, or both. Using Bessel functions, one can determine exactly what spectrum will result from an FM system for a given set of control values, but as you move toward more complex FM systems, the calculations required become correspondingly more complicated [38, pp. 231 - 242]. In fact, when calculating the spectrum of an FM signal, it becomes clear that, when the modulation index (the ratio of the modulator amplitude to the modulator frequency) is not zero, there are an infinite number of sidebands (although clearly only some fall within the audible frequency and amplitude range). This calculation, as explained by Julius O. Smith [41], is

$$x(t) = \sum_{k=-\infty}^{\infty} J_k(\beta) cos[(\omega_c + k\omega_m)t]$$

where t is time, J_k is the Bessel function of order k, β is the modulation index, ω_c is the frequency of the carrier oscillator, and ω_m is the frequency of the modulator oscillator.

A closely related problem with FM synthesis is that control parameters do not necessarily map well to perceptual results. Although control values like harmonicity (the ratio of the carrier frequency to the modulator frequency) and modulation index are more perceptually meaningful than directly controlling frequency and amplitude envelopes in FM synthesis, knowing what sound will result from a FM instrument is not always easy. Whereas in additive synthesis, the composer adds each partial at a specified frequency and a specified amplitude, a composer tweaking her FM synthesis instrument may see changes in amplitude over many frequencies simultaneously. As one would expect, having a comparatively limited number of controls to worry about gives the composer a much more manageable instrument, but far less control over the sounds that instrument produces.

2.1.3 Physical Modeling

Another prominent synthesis method developed in the last forty years is one based on physical modeling of sound-producing objects. By modeling the physical properties of the materials that make up an instrument, along with the couplings between those materials and the method used to excite them, we can produce relatively simple, but realistic computational synthesis models. Besides striving to precisely reproduce the sound of existing instruments, physical modeling can also be employed to create impossible instruments, including hybrids like bowed wind instruments or physically impossible ones like a piano with 100 foot long strings [40].

In terms of the control versus manageability tradeoff, physical modeling synthesis does well with both requirements. Controls for physical models are relatively precise and intuitive in their relationship to the sound produced. This is because these control parameters map directly to aspects of the virtual instrument being created, whether those be the tautness of a string, the shape of a bell, or the way the instrument is excited. The number of controls available to the user, while not tiny, is certainly more manageable than the multitude present in additive synthesis. How to exercise nuanced control of physical models is still an open research area, but basic control structures that perform reasonably well are already available.

The caveat with physical modeling is that, like the real instruments the physical models are derived from, the range of sounds that can be modeled is limited. Precisely because virtual physical models retain ties to their physical ancestors, the limitations that keep a violin sounding like a violin similarly constrain a physical model of a violin. Hybridization and the absence of real physical constraints allow physical modeling synthesis models to expand the range of sounds that can be created. But physical modeling's fundamental reliance on pre-existing instruments inevitably limits the timbres it can produce.

2.1.4 Other Implementations of Feature-based Synthesis

The goal of a feature-based synthesis method is not new. In 1979, David Wessel proposed a timbral synthesizer using just two timbral descriptors: spectral energy distribution and attack rate. This rudimentary timbral synthesizer controlled a bank of oscillators. Moving the control handles along the two axes of a graphical terminal changed the shape of the line segment envelopes that shaped the spectrum and amplitude envelope [47].

More recently, Hourdin, Charbonneau, and Moussa proposed a more nuanced feature-based synthesizer. They rely on an analysis of timbral similarity among synthesized tones and use the timbre space generated by a multidimensional scaling analysis to drive their system. They represent note events in this three-dimensional space by drawing a curve moving through the space and concentrate much of their paper on the specific task of resynthesizing existing tones with a limited amount of information. Although they address the possibility of creating original sounds, it is only in the context of creating hybrid tones between two existing known timbre curves. Thus, they theorize, from a known *forte* note and a known *piano* note, one could interpolate the timbre of a *mezzo forte* note. Because their focus is primarily on reducing the dimensionality of the timbre space to facilitate resynthesis with reduced data, they do not spend time exploring the possibility of feature-based synthesis for the creation of wholly new sounds [16].

Another recent way of manipulating timbre is proposed by James Beauchamp. Using robust phase vocoder or frequency tracking analysis to decompose existing sounds' spectra, Beauchamp writes of a series of tools that can be used to modify important physical parameters of the sounds before they are resynthesized. These physical parameters are generally ones that have been shown to affect our perception of timbre, including spectral centroid, spectral irregularity, and spectro-temporal incoherence, and by altering these during resynthesis, Beauchamp's theory allows for the modification of the timbre of existing sounds. This focus on timbre is notable, but like Hourdin et al., Beauchamp is concerned primarily with analysis and resynthesis of existing sounds, not the creation of new ones [1].

One other recent proposal for timbral manipulation that maps timbre as an explorable dimension comes from Haken, Fitz, and Christensen. They focus primarily on timbral morphing—moving smoothly between the timbres of two or more different sources. By formulating what they term a "timbre control space cube" and using eight source sounds, each corresponding to a corner of the cube, they map movement along the cube's three dimensions as moving from loud to soft, low pitch to high pitch, and cello to trombone timbre. Again, this idea allows direct control over timbre, but in its current form is limited to pre-existing sounds. Furthermore, although what the user controls as she moves along the timbral dimension of the cube is undoubtedly timbre, there is no attempt to deconstruct timbre into more precise, quantifiable, physical characteristics of the sounds [15].

Other recent implementations of feature-based synthesis take a new direction in approaching the problem of timbre's multi-dimensionality. The methods proposed by Jehan, by Le Groux, and by Johnson and Gounaropoulos all use machine learning to tackle the problem of timbre. Jehan's project focuses on transforming parameters of live sounds and outputting those sounds, with new timbres, in real time. His process begins by creating timbre models that span the range of timbres produced by a single instrument or instrument family. After a thorough, non-real-time analysis of this timbre set, Jehan's program can analyze the source sounds as they are being produced and map them to different parts of the pre-analyzed timbre space.

The sonic parameters that Jehan chooses to analyze from the input sound are pitch, loudness, and brightness (which he identifies as the percept associated with spectral centroid). From a given sound's analysis on these three dimensions, the machine learning algorithms that drive the synthesis process can choose a new timbre and output a corresponding note without affecting pitch, loudness and, if so desired, brightness. Jehan's project was intended primarily for use with "hyperinstruments" as a way to extend their timbral range, as well as for timbral morphing made possible when one controller instrument's output is mapped to the timbral map created from another instrument. Another possible application Jehan identifies is compression and resynthesis. This is because, after the analysis of an instrument's timbral range, these timbres can be represented by a reduced data set that, when driven by the same controller, can reproduce the full timbral range based only on control values. Nonetheless, by focusing on the particular demands of live performance, Jehan's timbral synthesizer sacrifices the user's ability to precisely define the timbre sought [18].

Sylvain Le Groux also uses machine learning to deal with timbre's multidimensionality, but unlike Jehan, Le Groux uses Principal Component Analysis to reduce the dimensionality of the problem space. Even so, he finds that to produce a sound, he still must specify roughly 80 coefficients. By providing a training set of sounds of varying pitches, loudnesses, and timbres, he found that he could replicate other sounds within the bounds of his training set. Le Groux's goals are more in line with those of this paper than Jehan's were. Particularly, Le Groux worked to take pre-existing sounds and vary their pitch and loudness while holding their timbre relatively constant.

This approach is certainly interesting and Le Groux's use of Principal Component Analysis to reduce the dimensionality of his problem could be applied even if using other, non-machine learning approaches to solve the problem. Yet this method, like the others mentioned above that attempt to synthesize sound based on timbre, does not take full advantage of our greater understanding of the physical correlates of the psychoacoustic phenomenon of timbre. Rather than allowing the user direct control over physical parameters, LeGroux and others substitute descriptive adjectives like "brightness" and roughly approximate these descriptors with physical properties [22].

Johnson and Gounaropoulos are particularly interested in the relationship between the adjectives that musicians use to describe a sound and the timbral characteristics these adjectives define. They rely on machine learning methods to analyze sounds in the training set and correlate the adjectival timbre descriptions provided by listeners with particular characteristics of the sounds. To link the results of this analysis with a synthesis method, they acknowledge that there are two general approaches to the problem: an analytic approach and a machine learning approach. They choose the latter, using a neural network on a reduced number of components extracted from sounds in the training set, and shaping the values of those components according to the user's adjectival timbre description [11] [19]. This paper explores the other option—the analytic approach. Using this approach, timbral synthesis aims to give users direct, unmediated control over specific aspects of timbre and thus, to allow users to be as precise in specifying the timbre they want as our current understanding of timbre allows.

2.2 Timbre

Understanding what timbre is and how we perceive it is a significant challenge and one that still has not been fully resolved. However, our understanding of timbre and our ability to quantify it have advanced greatly in the last few decades. Since timbral synthesis depends on this ability to quantify the underlying physical properties of sound from which we derive timbral recognition, understanding the current state of research on timbre is essential to understanding this implementation of timbral synthesis, its advantages, and its weaknesses.

2.2.1 Complexity of Timbre

Compared to pitch, loudness and duration, timbre is a relatively complex attribute of sound. In part, this complexity probably derives from the nebulous way in which timbre is defined: everything about a sound that is not pitch, loudness, duration, or spatialization. When investigations of the psychoacoustic phenomenon of timbre began in earnest, more than fifty years ago, there was already a recognition that the distribution of energy in the spectrum had a profound effect on how that spectrum sounded. However, there was also recognition that spectral envelope alone was not determinative of timbre, and that spectral envelope also affected pitch and loudness. Thus, in 1951, J.C.R. Licklider wrote, "the timbre of a complex sound has usually been defined as the subjective quality that depends upon the complexity or overtone structure of the physical sound. We have seen, however, that both the loudness and the pitch of a complex tone are influenced to some extent by its overtone structure" [23, pg. 1019].

This early recognition that timbre is a complex percept is crystallized in Licklider's conclusion, "it can hardly be possible to say more about timbre than that it is a 'multidimensional' dimension" [23, pg. 1019]. Yet for composers, the possibility of manipulating existing timbres by physically modifying existing instruments or creating entirely novel timbres working in the electronic domain is one of the richest areas for exploration. Music technologists working on sound synthesis and psychoacousticians studying timbre are also deeply invested in finding better ways to quantify timbre. Yet the problem of defining timbre remains unsolved.

In her 1989 presentation entitled "Why is Musical Timbre so hard to understand?," Carol Krumhansl identifies the numerous holes that remain in our knowledge about timbre. She suggests that a universal, fully inclusive definition of timbre may be impossible and that different groups, given their differing needs, will adopt different working definitions of timbre that suit their purposes. She lays out one possible hierarchy of timbral analysis, pointing to expressive variations of timbre that players of acoustic instruments can effect as the micro level, broad similarities between different tones produced by the same instrument as an intermediate level, and classes of sounds (e.g. percussive sounds, bowed sounds, etc.) as a macro level. Krumhansl's delineation of the difficulties involved in defining timbre is worth keeping in mind [20].

This paper, and the timbral synthesis method it proposes, focuses on one particular way of defining timbre using the physical properties of sounds and their relative perceptual salience in determining timbral similarity.[1] Given the different ways that one could approach the problem of defining timbre (and synthesizing sound based on that definition), there are certainly other potential forms of timbral synthesis worth exploring that would vary markedly from this one.

[1] For a full discussion of the different methods for experimenting with timbre and the advantages and drawbacks of each, see Hajda et al. [14]

2.2.2 Timbre Experiments

Before discussing the specific physical properties that factor into the proposed method of timbral synthesis, it is helpful to undertake a brief review of the history of the experiments that led to our current understanding of timbre. Early experiments on timbre, like Kenneth Berger's in 1963, began to expand the understanding of timbre, particularly in recognizing the temporal attributes of a sound that contribute to timbre. In Berger's experiment, ten tones were recorded from ten different wind instruments, all of them at the same pitch, roughly the same loudness (as monitored by the players themselves using a C-weighted sound-level meter), and all of the same duration. These tones were played back for a group of musically-trained students, who were asked to identify which tone belonged to which instrument. In addition to the unaltered tones, Berger also created experimental tones in which the attack and release portion of the notes were removed, in which the tones were played backwards (reversing the shape of the amplitude envelope), and in which all harmonics above the fundamental were filtered out [2].

Each of these modifications resulted in less accurate identification of the tones, giving indications of the influence on timbre of the attack and release portions of the amplitude envelope, the general shape of the amplitude envelope, and the upper harmonics of a note, respectively. A decade later, more nuanced studies of timbre were going on, many of them using multidimensional scaling to show the relative similarities of different tones and to identify which physical characteristics of the sound were most salient to timbral recognition. The reasons for designing experiments around dissimilarity ratings and multidimensional scaling are compelling. As McAdams et al. put it, the reasons include "(1) the judgments [of dissimilarity] are relatively easy to make for subjects; (2) the technique makes no a priori assumptions about the nature of the dimensions that underly [sic] the perceptual representation used by subjects to compare the timbres of two sound events; (3) the resulting geometric representation of the data can be readily visualized in a spatial model; and (4) the spatial model has been found to have predictive power" [27, pg. 178].

In 1976, John Grey designed an experiment using computer-synthesized tones resynthesized from recordings of 16 orchestral instruments. Grey emphasized the importance of using computer-synthesized tones because it allowed him to definitively equalize the test tones in the dimensions orthogonal to timbre. His confirmation that the tones were precisely equivalent in pitch, loudness and duration was certainly an advance over earlier studies, though the potentially confounding effects of the analysis and resynthesis process may not have been given ample weight. The other major advance evident in Grey's analysis is the introduction of multi-dimensional scaling. Whereas earlier studies like Berger's relied on the researcher's initial hypotheses to drive the analysis of the results, Grey and others used multidimensional scaling, with the advantage that they did not need to predict which physical parameters of sound would be salient in their subjects' timbre dissimilarity ratings. Grey also made use of a more complete set of analysis tools to dissect the physical properties of the different sounds. With these tools, he was able to create timbre spaces and to begin to assign relative salience to different properties of sound [12].

Grey's study, like many more recent psychoacoustic studies of timbral perception, asked subjects to give pairs of test tones a graded similarity rating, rather than just try to identify them. Thus, rather than just presenting confusion matrices in his results as in earlier studies, Grey used multidimensional scaling to create two- and three-dimensional timbre spaces. From these timbre spaces, Grey hypothesizes that the first axis corresponds to "spectral energy distribution." According to his definition, he is including both harmonic spectral centroid and harmonic spectral standard deviation (or harmonic spectral spread). He hypothesizes that the second axis corresponds to the "synchronicity" of the upper harmonics during the attack and decay portions of the tone, as well as the overall "spectral fluctuation" during the steady state portion of the tone (harmonic spectral variation). He hypothesizes that the third axis corresponds to the noisiness of spectrum during the attack portion of the tone (high-frequency, low-amplitude, inharmonic energy) [12, pp. 1273-1274].

Though Grey's conclusions have been significantly refined in the time since his study was completed, his use of multidimensional scaling and electronically manipulated test tones foreshadowed the direction that timbral research would take. More recently, researchers have focused on an increasing number of physical properties of sound to determine which contribute most to our perception of timbre. Particularly, they have explored ways to reduce the complexity of the data needed to resynthesize sounds in a perceptually lossless way. In 1998, McAdams, Beauchamp and Meneguzzi examined different simplifications of spectro-temporal parameters to try to identify which simplifications were most audible to listeners comparing the resynthesized sounds to the unaltered originals. Their conclusions point to "spectral-envelope shape (jagged vs. smooth)"[2] and "spectral flux"[3] as having the highest salience. But they also point out that the relative salience of different simplifications varied widely depending on which instrument they were resynthesizing. They explain this variance by noting that the absolute salience of a given parameter is critical, but that some sounds exhibit more or less activity in regards to different parameters, and this relative activity may also play a role. In other words, although spectral flux may be a highly salient parameter, it may contribute more to timbral recognition in tones with high spectral flux, while contributing more modestly in tones with little spectral flux [26, pg. 894].

[2] This corresponds roughly to harmonic spectral deviation as defined in the MPEG-7 standard.

[3] This corresponds to harmonic spectral variation as defined in the MPEG-7 standard.

This finding is less relevant for "pure synthesis" applications where the goal is not the reproduction of an existing sound. But it is an important consideration for anyone using timbral synthesis (or other synthesis techniques, for that matter) to recreate an existing sound. This is particularly true given McAdams et al.'s finding that when several parameters vary simultaneously, subjects concentrate on the most salient of those parameters, to the point that they may largely disregard other parameters. The authors hypothesize that the ability of subjects to discriminate between sounds that vary in several dimensions at once may depend solely on the most salient dimension and that predictions about the subjects' performance may safely be made based only on that parameter. Thus, lack of attention to a particularly salient factor during resynthesis could greatly hamper the creation of an accurate reproduction.

Some other factors to consider when examining these timbre studies are raised in fuller reviews by Hajda et al. [14] and Hajda [13]. The shortcomings identified in those reviews limit the applicability of these timbre studies' results and point toward areas needing further research. One of these areas, for example, is how timbre is perceived in real-world musical contexts. Specifically, does timbre function differently in a multi-note phrase than in single notes (a distinction that few timbre studies have made). Another area is whether defining timbral descriptors in different ways affects their relative salience. For example, as Hajda points out, there are different ways of defining attack time that, because of important changes in spectral stability during the attack, could have a profound effect on how listeners perceive the attack and what role it plays in their perception of timbre [13, pp. 253–255]. These weaknesses are important to keep in mind, since they remind us that our current understanding of timbre, while far more complete than it was in the past, remains incomplete.

2.2.3 Current Understanding of Timbre

In the last decade and a half, several researchers have focused their efforts on identifying exactly which physical attributes of sound matter most in our perception of timbre. One of the studies that still carries great weight today is McAdams et al.'s study from 1995. Specifically, because this study was used as the basis for part of the MPEG-7 timbral descriptors (described in more detail below), it is of great importance in the current context. The study included resynthesized versions of traditional instruments, as well as hybrid sounds created by combining pairs of traditional instrument sounds. It used a larger set of subjects than previous studies (98) and specifically set out to determine if professional musicians, amateur musicians and non-musicians would rely differently on the different aspects of timbre for their dissimilarity ratings. As with other studies, the pitch, loudness and duration of all the sounds were equalized so as to force subjects to rely solely on timbre in making their dissimilarity judgments [27].

The analysis of the multidimensional scaling models produced by the 1995 experiment was more thorough than previous analyses, particularly in regard to mapping the dimensions of the models to physical properties of sound. The researchers identified two different multidimensional scaling models that fit their data well—one in six dimensions without specificities and one in three dimensions with specificities.[4] The six-dimensional model is particularly interesting because it indicates that, contrary to what previous studies had shown, listeners might be discriminating among timbres using far more than three parameters. McAdams et al. also suggest that different individual subjects, and different groups of subjects, relied more heavily on one or another aspect of timbre in making their dissimilarity ratings. In addition to more detailed multidimensional models, McAdams et al. also examined weighting schemes to see if they could determine how much subjects relied on the physical characteristics that corresponded to each dimension of their model [27].

In the three-dimensional model, all three dimensions were well correlated with physical properties of the tones. Specifically, the three dimensions correlated

[4] Specificities, in this case, meaning particular characteristics about a single test tone that differentiated it from all other test tones, but were unique to that test tone and, as such, did not map to an axis of the multidimensional scaling model. Examples of specificities recognized subjectively in this experiment include a "slightly raspy attack," a "metallic sound," and a "wobbly double attack" [27, pg. 189]

with log-attack time (the logarithm of the rise-time the tone takes to reach its full amplitude), spectral centroid, and spectral flux ("the average of the correlations between amplitude spectra in adjacent time windows" [27, pg. 188]. In the six-dimensional model, only one dimension correlated well with a single physical property of the tone, the log-attack time. That said, although there were not one-to-one correlations between the other dimensions and individual physical properties of the tones, spectral flux, spectral centroid, and measures of spectral fine structure (including spectral irregularity) each correlated with more than one of the six dimensions.

One of the other studies that figured prominently in the establishment of the MPEG-7 timbre standard was Stephen Lakatos' "A common perceptual space for harmonic and percussive timbres," from 2000 [21]. As indicated by the title, this study broadened the field of inquiry to include not only harmonic sustained tones from traditional instruments, but also a wide array of non-sustained, percussive tones, including bongo drum, steel drum, bamboo chimes and bowed vibraphone [21, pg. 1428]. Lakatos had his subjects complete similarity ratings for a grouping of just the harmonic tones, for a grouping of just the percussive tones, and then for a mixed group including some of each.

Lakatos made use of the same advanced multidimensional scaling techniques as McAdams et al. did, though he also used an "extended additive tree model" to detect “nested and overlapping groupings of timbres” [21, pg. 1427]. His results further confirmed the salience of spectral centroid and log-attack time as the primary factors that his subjects used in ordering the timbres. Interestingly, though, Lakatos was unable to correlate the third dimension of his model with a psychophysical property of sound for the harmonic grouping. However, for the group of percussive sounds, Lakatos was able to correlate the third dimension (though the correlation was far less strong than for the other two dimensions) with what he terms “timbral richness” [21, pg. 1436].

The other important point that Lakatos makes is closely related to the specificities identified by McAdams et al. He hypothesizes that trying to map all aspects of timbral dissimilarity ratings to a set a continuous values may be misleading. As McAdams et al. realized, some timbral differences between tones may be due to unique factors that set those tones apart and are simply not present in other tones [21, pg. 1437]. As an example of one possible noncontinuous factor, Lakatos cites excitation method, which is particularly relevant given the wide spectrum of excitation methods present in his sample set. As further evidence of the potential import of these non-continuous dimensions of timbre, he also cites the experiences of “students of computer music, who have attempted to synthesize electroacoustic sounds by manipulating their time-varying spectrum along dimensions similar to those suggested by MDS studies” [21, pg. 1437]. This warning is certainly worth keeping in mind, since the method for timbral synthesis proposed in this paper is a more nuanced version of that approach.

One final study that is worth noting (although it was completed after the MPEG-7 timbre specifications had been adopted) is Caclin et al.'s confirmatory study in which they specifically examined the perceptual salience of spectral flux and spectral irregularity in comparison to the salience of attack time and spectral centroid [3]. This study again confirmed spectral centroid and attack time (particularly the logarithm of attack time) as the most salient factors for listeners in making dissimilarity ratings. It also confirmed that spectral irregularity (in this case the attenuation of even harmonics, as occurs in closed tube resonators like the clarinet) is a salient, continuous dimension in dissimilarity ratings.

The results for spectral flux were more ambiguous. The researchers modeled spectral flux by varying the amplitude envelopes of different harmonics over the first 100 ms of the sound. They admit that spectral flux modeled differently—perhaps in the steady state portion of the tone—might produce different results. In three-dimensional tests, in which spectral centroid, attack time, and spectral flux varied, the contribution of changes in spectral flux was unclear, and minimal at best. Differentiation based on spectral flux was strongest between pairs of tones both having high spectral flux. Differentiation between pairs where one had high spectral flux and one had low spectral flux was less evident and differentiation between pairs where both tones had low spectral flux did not appear to take spectral flux into account. Results for two- and one-dimensional models, where spectral flux varied with only one other parameter or varied alone were clearer. In those cases, spectral flux was significantly more salient for listeners. This indicates that context matters when discerning between timbres. More perceptually salient properties can, in some sense, mask less salient ones [3].

2.3 MPEG-7

MPEG-7 is a Multimedia Content Description Interface developed and maintained by the Moving Picture Experts Group. Unlike other major MPEG standards (including MPEG-1, MPEG-2, and MPEG-4), MPEG-7 does not comprise standards for coding and compression of multimedia data, but rather, is a standardized scheme of descriptors for multimedia content to facilitate description, categorization, querying, navigation and sorting of this data. These description schemes are independent of meta-data that may accompany content, like encoding format, title or author data, or even data about the media itself (e.g. a histogram of a digital photo). Different descriptor schemes of the MPEG-7 standard are intended for use with audio, visual and multimedia content and the values of these descriptors are, in general, calculated by analysis engines specifically for an MPEG-7 description of the multimedia object [4].

MPEG-7 is highly extensible and was designed with the knowledge that our ability to analyze and categorize multimedia content will improve over time. For a descriptor to be included in the MPEG-7 standard, it must go through a rigorous process of proposal stages and challenges to prove that it is both accurate and useful in describing the content. Because of the wide variety of material that MPEG-7 seeks to describe, descriptors are organized into description schemes with specifically limited application. For audio, MPEG-7 includes both low-level descriptors that apply to all audio signals, as well as higher-level, application-specific descriptors relevant only for particular types of audio [37].

The low-level descriptors deal primarily with grouping audio events into hierarchically meaningful clusters and separating individual events so they can be analyzed as distinct entities and thus, produce accurate values that apply only to the specified event. Once these events are clustered, each audio segment can be described by low-level descriptors including temporal envelope, spectral envelope, harmonicity, spectral centroid, and fundamental frequency [37, pg. 727]. The other low-level segmenting mechanism present in MPEG-7 is a silence segment that, importantly, tells the analysis mechanism what it should not bother analyzing.

In addition to these low-level descriptors, MPEG-7 has four higher-level description schemes, each specifically suited for a particular type of audio content. The sound effects description tools are used to identify and index sound effects, even in complex sound environments. The tools for description of spoken content do include the ability to transcribe speech, but, because of the inherent difficulty of this task with sources of unknown origin and quality, these tools also include higher level descriptors to classify the stream. The tools for melody contour description quantize the fundamental frequencies of the note series and store them, along with basic rhythmic data to, for example, enable query-by-humming systems. Finally, and most importantly to the subject of this paper, the musical instrument timbre description tools use a limited number of descriptors to describe a wide range of musical sounds.

Because the relative salience (and applicability) of different physical properties of sound depends in part on the type of sound, MPEG-7 divides the timbre description scheme among four types of musical instrument sounds. These are "harmonic, sustained, coherent sounds," "nonharmonic, sustained, coherent sounds," "percussive, nonsustained sounds," and "noncoherent, sustained sounds" [37, pg. 728]. At present, timbre description schemes exist only for the first and third of these classes, though as Quackenbush and Lindsay note, those are the two classes most prevalent in musical sound [37, pg. 728].

The originators of the musical instrument timbre description scheme acknowledge the difficulties involved in quantifying timbre, but they rely on the string of perceptual experiments performed over a span of decades, the history of which is described in detail above. From among those numerous experiments, the MPEG-7 description scheme refers in particular to three experiments: those performed by Krumhansl [20], McAdams, Winsberg, Donnadieu, De Soete and Krimphoff [27], and Lakatos [21]. All three of these studies used dissimilarity ratings among pairs of tones and multidimensional scaling analyses to identify the psychophysical parameters that correspond to these dissimilarities [34].

From these three studies, the MPEG-7 standard identifies five descriptors that are applicable for harmonic sounds and three that are applicable for percussive sounds. The harmonic descriptors are derived from Krumhansl's and McAdams et al.'s studies, while the percussive descriptors are derived from Lakatos'. The harmonic descriptors are log-attack time (*lat*), harmonic spectral centroid (*hsc*), harmonic spectral spread (referred to in the MPEG-7 standard as harmonic spectral standard deviation, *hsstd*), harmonic spectral variation (*hsv*), and harmonic spectral deviation (*hsd*). Percussive sounds share the log-attack time descriptor with harmonic sounds, but, because of their non-harmonic nature, complement log-attack time with temporal centroid (*tc*) and spectral centroid (*sc*) [34, pg. 3]. These high-level descriptors, specific to the type of sound they describe, combined with low-level descriptors described above (e.g. fundamental frequency, spectral envelope, harmonicity, etc.) constitute the full description scheme for these sounds.

Before its adoption as part of the MPEG-7 standard, this timbre description scheme was tested with a series of natural sound samples. The authors tested subjects by comparing different sets of sounds to a target set of sounds and asking the subjects to indicate which set more closely resembled the target set. As they point out, "'timbre' as it is currently understood, is a relative feature. In this sense, what we are interested in is more the description of the relative positions of the sounds than their absolute position in the Timbre Space" [34, pg. 1]. As such, the validation experiments were designed to mimic real-life uses of the timbre description scheme and to make sure that sounds that clustered together in the timbre space were in fact identified by subjects as timbrally similar.

An important consequence of the descriptors' relativity is that more than one sound occupying precisely the same position in a timbre space should, by definition, sound very similar to other sounds occupying that space, by virtue of its lack of Euclidean distance in the space. However, because many different spectral and amplitude envelopes could result in the same values for the timbre descriptors, there is no guarantee that they would be perceptually indistinguishable. This is to say that occupying the same point in five- or three-dimensional space (for harmonic or percussive sounds, respectively), would not necessarily result in “auditory metamers”[5].

[5] Metamers are colors with different spectral compositions that appear identical because of the limitations of the human visual system

Chapter 3

Tools

The tools used in this version of timbral synthesis fall into two categories: mathematical and software. Because this method of timbral synthesis seeks to precisely specify timbre, it requires mathematical transformations that allow us to come up with the coefficients necessary to define spectral and temporal envelopes. Once those envelopes have been determined, the rest of the tools are used in the synthesis process, as the sound corresponding to the specified envelopes is generated.

Neither the mathematical tools, nor the software used in this implementation, are integral to the process of timbral synthesis as defined in this paper. The tools used were chosen because they fit the needs of the process. In fact, during the development of this implementation, several software packages were actually swapped out for the packages that are currently in use, with no adverse effect on the underlying project. Thus, the enumeration of these tools should not be construed as an endorsement of these tools as the only means for implementing timbral synthesis. There almost certainly are other methods by which this type of timbral synthesis could be implemented, and exploring those possibilities is a fertile area for continuing research.

3.1 Linear Optimization

The basis of timbral synthesis lies in its using analysis equations transformed into synthesis equations. These equations allow the user to define values for the perceptually significant aspects of timbre. Once these values are defined, they effectively limit the size of the solution space (i.e. the number of possible spectra that, when analyzed, would conform to the user-entered values). To find these solutions, a process must find values for the coefficients that define these spectra.

As we will see, the amplitude envelope is defined by a small number of coefficients and thus, does not require an advanced method for finding an acceptable envelope. The frequency spectrum, on the other hand, is defined by a comparatively large number of coefficients and requires more complex ways of finding the right spectrum for a set of user-defined timbral values. Thus, the challenge is twofold. First, we must have a method that allows us to solve for a set of coefficients bounded by several constraints simultaneously. Then, because most solution spaces contain more than one feasible solution, we must have a way to choose the "best" among them. Linear programming provides a way to do both[1]

By reducing the synthesis equations to a series of linear constraints, linear programming allows us to define the solution as a set of coefficients that is subject to all of the constraints. Additionally, linear programming requires that we define an objective function and an optimization direction (maximization or minimization). This suits our purposes by allowing us to define the best solution among all the feasible ones. For example, one could define a simple problem like

$$\begin{array}{lccccccc} \text{Maximize} & 5a & + & 4b & + & 3c & & \\ \text{subject to} & & & & & & & \\ & a & + & b & + & c & = & 10 \\ & 2a & + & b & & & \leq & 4 \\ & 4a & & & + & c & \geq & 6 \end{array}$$

By modeling our search as a maximization or minimization problem constrained by a number of descriptor equations (it must have this harmonic spectral centroid, this harmonic spectral spread, etc.), we can find a spectrum that meets all of our criteria and is the optimal solution among all of the feasible ones.

Specifically, if we model the problem as a maximization problem, by weighting each coefficient in the objective function differently, we can push our desired so-

[1] It is worth noting, again, that there are methods other than linear programming by which the solution space could be found and a feasible solution located. For example, machine learning, which was used by Jehan and by Le Groux in other implementations of timbral synthesis, could be utilized in a different manner to find a specific, user-defined timbre.

lution toward a particular shape. For example, by giving the first coefficient—the one representing the amplitude at the fundamental frequency—the most weight, we can force our solver toward solutions with most energy concentrated in the fundamental. In a minimization problem, we can enter the amplitudes from a known spectrum as weightings for coefficients in the objective function and thus, define the optimal solution as the one that deviates least from that spectrum (although, because absolute value, i.e. distance, is non-linear, this approach is more complicated than a maximization approach).

Having modeled the search for a desired spectrum as a linearly constrained optimization problem, there are several efficient algorithms that can solve it. The Simplex method, the original linear programming algorithm, iteratively reduces the values of slack variables to move quickly from a basic feasible solution to the optimal solution (for a fuller explanation of the Simplex method, see Cormen [6, pp. 770–821]). And although the Simplex method requires that all constraints be linear, it is more than capable of handling the optimization problems that timbral synthesis produces at very efficient speeds. Linear optimization models are often used in industrial applications with hundreds of thousands of constraints and millions of variables.[2] Timbral synthesis problems, many orders of magnitude

[2] As Franois Pachet notes [30], constraint-based methods have already been used for other musical applications, including harmonization of melodies (see [9], [29], and [32]), and spatialization (see [31]).

smaller than those problems, are solved quickly and efficiently by any software implementation of the Simplex algorithm, including the one described below.

3.2 Software

The software used for this implementation of timbral synthesis was, in large measure, selected for convenience. Because the steps in the timbral synthesis process (formulation of a linear optimization problem, solution of the problem, formulation of spectral and amplitude envelopes, and synthesis based on those envelopes) can be performed separately from each other, the current implementation is modular, and the software that performs one task can be replaced without affecting the others.

3.2.1 Java

The current implementation is written entirely in Java. Java provides a native graphical user interface toolkit (SWING), has several linear programming solver application programming interfaces (API) available for it, and is fast enough to meet the needs of the current implementation. There are also several audio synthesis packages available in Java, more than one of which meets the rather basic synthesis needs of timbral synthesis. One could easily write a similar implementation in C++ [45], a scripting language like Python [36], or even a graphical programming environment like pd [35], but Java has proven more than adequate for this proof-of-concept implementation.

3.2.2 Mosek

Mosek is an optimization software package that can handle linear optimization problems, as well as conic quadratic constraint problems and general, convex nonlinear problems. Mosek can be integrated into Java, C, and C++ applications via standard, well-documented APIs. Mosek is built to handle large-scale optimization problems, so the relatively small problems that timbral synthesis poses are solved quickly—on the order of 20 milliseconds [28].

Mosek is proprietary software, but is free for users solving problems on the scale that timbral synthesis requires. Of the linear programming packages that were examined, Computational Infrastructure for Operations Research (COIN) lacked an easy interface and Java API [5]; CPLEX was difficult to license, even for educational use [17]; Xpress-MP, when licensed for educational use, allowed access only through its proprietary interface, not via any of its callable libraries and their APIs [7]; and GLPK lacked the comprehensive documentation that would have enabled a non-expert user to implement it easily [10]. Unlike these other linear programming packages, Mosek's Java API is well-documented and easy to understand, even without a thorough background in linear programming. For this reason, Mosek was ideal for a prototype implementation. Future implementations, in order to meet scalability goals or in order to be fully free and open-source (FOS), could easily replace Mosek with one of the other FOS solvers available on the Internet.

3.2.3 JSyn

JSyn is a high-level, real-time audio synthesis engine for Java. It is based on the unit generator model, where different audio objects—like oscillators, noise generators, filters, mixers, and envelopes—are "wired" together to create complex instruments. JSyn requires little in the way of system resources and a simple additive synthesis instrument, upon being given synthesis parameters, will begin producing sound with a latency of under 0.5 seconds. Because JSyn is intended for real-time synthesis, it supports flexible read-through of amplitude envelopes, enabling timbral synthesis of continuous sounds where the full shape of the amplitude envelope is not known at the initiation of playback. Thus, for example, we can synthesize a continuous tone, modifying its spectral envelope without having to stop and restart synthesis [43].

The JSyn Software Developer Kit is freely available and supports the creation of both standalone Java applications and of Java Applets. Because of its high-level synthesis methods, JSyn proved a better option for the synthesis aspects of timbral synthesis than the Java Sound API, which provides mostly low-level support [46]. And because of its real-time synthesis methods, JSyn was more appropriate for timbral synthesis than the jMusic package [44], with which timbral synthesis was originally implemented. That said, again, JSyn could certainly be replaced by other audio synthesis tools without adversely affecting the rest of the timbral synthesis process.

Chapter 4

Timbral Synthesis

The fundamental challenge of using a timbre description scheme as a synthesis tool is one of taking the equations that define the timbral descriptors—equations that were designed to analyze existing sounds—and turning those equations into synthesis equations. This task is straightforward for some of the equations because of their one-to-one correlation with aspects of the spectral or amplitude envelope, but more complex for others. Furthermore, because the intent of the timbre description scheme is to drastically reduce the amount of data required to meaningfully describe a given timbre, using this reduced data set to define a sound becomes a problem of dimensionality.

In the current implementation of timbral synthesis, I use the MPEG-7 instrumental timbre description scheme, which in its current form defines seven descriptors to work with. The MPEG-7 scheme is a good model for timbre synthesis since its inclusion in the MPEG-7 standard ensures that it was rigorously tested and subjected to peer review. Furthermore, MPEG-7 is a good reference to use because no part of the standard is set in stone—if a better model for timbre description is proposed, that model will replace the current one and the descriptors used by timbral synthesis could updated accordingly. That said, MPEG-7 is by no means the only timbre description scheme that one could use for timbral synthesis and using descriptors particularly suited to a specific synthesis task is one area of further exploration that could prove very rewarding.

Having chosen a timbre description scheme for an implementation of timbral synthesis, the primary task is how to convert that reduced control data set into enough data to drive the synthesizer. Since additive synthesis has, as Curtis Roads puts it, "a voracious appetite for control data" [38, pg. 143], generating that control data from a severely limited set of descriptors is inherently problematic. Even if a user specifies values for all seven timbral descriptors (*lat*, *tc*, *sc*, *hsc*, *hsstd*, *hsv*, and *hsd*), along with the orthogonal control data for pitch, duration and loudness, that is only ten values. A basic additively synthesized harmonic tone might have 20 oscillators at integer multiples of the fundamental frequency, along with shaped noise (or a multitude of oscillators at closely spaced inharmonic frequencies). Controlling the 20+ elements in the additive synthesizer with values from only 10 synthesis parameters means finding an ill-defined point in a highly dimensional space.

4.1 Timbral Descriptors

Even in a model of reduced complexity, in which the frequencies of the harmonics are assumed to be exact integer multiples of the fundamental frequency and the non-harmonic portion of the spectrum is generated by a single shaped noise generator, synthesis requires amplitude values for each of the twenty harmonics, as well as a shape for the noise generator's filter, and a temporal envelope. To define the spectral envelope of the harmonic spectrum, which requires as many coefficients as there are harmonic components, only three descriptors (*hsc*, *hsstd*, and *hsd*) are relevant. To define the shape of the non-harmonic portion of the spectrum, there is only one descriptor (*sc*). And to define the temporal envelope, there are only two descriptors (*lat* and *tc*). Particularly for the harmonic spectrum, the point in three-dimensional space defined by the hsc, hsstd, and hsd values is not sufficient to specify the point in 20-dimensional (or more) space that constitutes a fully defined spectrum. Thus, the task becomes one of either reducing the dimensionality of the target, increasing the dimensions of the point specified by the descriptor values, or both.

Before addressing solutions to the dimensionality problem, it is worth examining the manner in which each analysis equation is transformed into a synthesis equation, since the process is not uniform for the different equations.

4.1.1 Log-Attack Time

The simplest transformation is that of log-attack time. Log-attack time is defined in the MPEG-7 standard as

$$lat = log_{10}(t_{max} - t_{0.02max}) \tag{4.1}$$

where t_{max} is the time at which the amplitude is maximum and $t_{0.02max}$ is the time at which the amplitude first reaches 2% of its maximum value. The inclusion of the 2% term is simply a convenient way to mark the beginning of a note and it can safely be ignored for synthesis since the amplitude envelope will precisely define the start point of the note. For a comparison of envelopes with very different log-attack times, see Figure 4.1. Thus, the synthesis equation for log-attack time is

$$t_{max} = 10^{lat} \tag{4.2}$$

where lat is a user-specified log-attack time and t_{max} is the time at which the amplitude is maximum (i.e. the end of the attack portion of the amplitude envelope). Using the log-attack time equation for synthesis is relatively simple because there is a one-to-one correlation between the value inputted and a single parameter of the synthesis process [33].

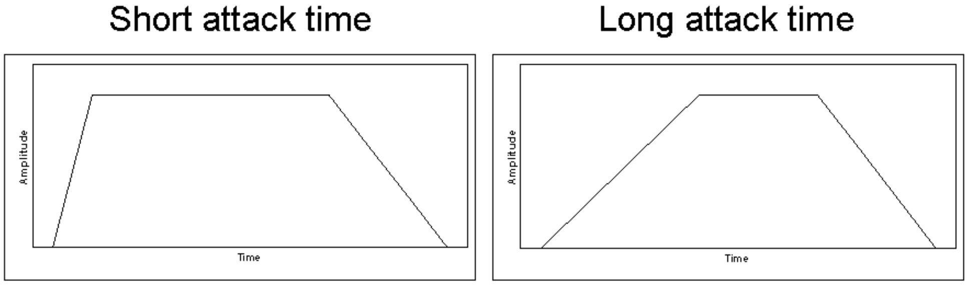


Figure 4.1: Two envelopes with different attack times

4.1.2 Temporal Centroid

The other temporal descriptor has both a more complex analysis equation and a less direct correlation with the synthesis process, rendering its conversion more difficult. Temporal centroid is defined as the weighted average of the instantaneous temporal envelope of the sound in which the weights increase with the sample number of the amplitude envelope. This is expressed as

$$tc = \frac{\sum_{n=1}^{length(signal)} n * envINST(n)}{\sum_{n=1}^{length(signal)} envINST(n)} \tag{4.3}$$

where n is the instaneous position in the temporal envelope, as measured in samples, and $envINST$ is the instantaneous value of the amplitude envelope. This analysis equation can be transformed into the synthesis equation

$$\frac{y_1 + 2y_2 + 3y_3 + \ldots ny_n}{y_1 + y_2 + y_3 + \ldots y_n} = tc \tag{4.4}$$

where tc is the user-defined temporal centroid, y_n is the value of the amplitude envelope at the nth sample and n is the sample number. For a comparison of envelopes with the same log-attack time, but different temporal centroids, see Figure 4.2.

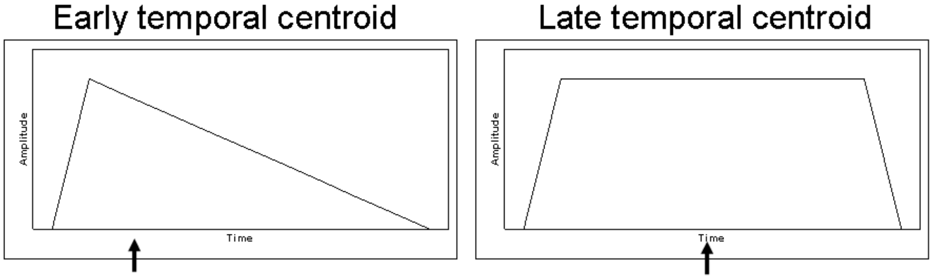


Figure 4.2: Two envelopes with different temporal centroids, as indicated by the arrows

Unlike log-attack time, temporal centroid is an average, and as such, does not correspond directly to any one parameter of the synthesis process. The temporal centroid is of course affected by the log-attack time since *lat* describes the slope and length of the attack portion of the envelope. Temporal centroid is similarly affected by the duration of the event it describes since duration defines the $length(signal)$ present in the analysis equation. As a single equation, temporal centroid can define only one variable during the synthesis process; as such, strategies for limiting the number of amplitude envelope parameters affected by temporal centroid will be discussed later [33].

4.1.3 Harmonic Spectral Centroid

Of the parameters affecting the definition of the harmonic spectrum, the most fundamental is harmonic spectral centroid. Like all of the “harmonic” descriptors, harmonic spectral centroid takes into account exclusively the harmonic partials—those located at (or nearly at) integer multiples of the fundamental frequency. Harmonic spectral centroid is a weighted average, over the entire sound duration, of the amplitudes of the harmonic components of the spectrum. In order to calculate the harmonic spectral centroid for the full sound, a running harmonic spectral centroid is calculated for each frame of a Short Time Fourier Transform. Weighting is determined by the frequency at which the component is located, such that the 10th partial has ten times the weight of the first partial (fundamental). For a comparison of spectra with different harmonic spectral centroids, see Figure 4.3.

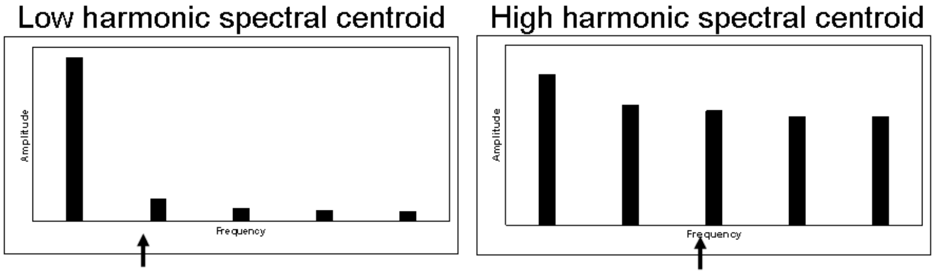


Figure 4.3: Two harmonic spectra with different harmonic spectral centroids, as indicated by the arrows

The analysis equation defining the harmonic spectral centroid of each frame is

$$ihsc(frame) = \frac{\sum_{harmo=1}^{nb\_harmo} f(frame, harmo) * A(frame, harmo)}{\sum_{harmo=1}^{nb\_harmo} A(frame, harmo)} \tag{4.5}$$

where f is the frequency of the specified harmonic in a given frame and A is the amplitude of that harmonic in that frame. Once a harmonic spectral centroid has been computed for all frames of the sound, the mean of those values is the harmonic spectral centroid of the entire sound. Equation 4.5, transformed into a synthesis equation, becomes

$$a_1 + 2a_2 + 3a_3 + \ldots na_n = \frac{A_{Total} * hsc}{f_0} \tag{4.6}$$

where A_{Total} is the user-defined sum of the amplitudes of the harmonic components of the spectrum, hsc is the user-defined harmonic spectral centroid, f_0 is the user-defined fundamental frequency of the spectrum, and a_n is the amplitude of the nth harmonic partial of the spectrum.

This expression is made simpler by the assumption that the user will define a total amplitude for the harmonic components of the spectrum. The reason for imposing this requirement will become clearer below. Equation 4.6 also assumes that the harmonic components of the spectrum lie at exact integer multiples of the fundamental frequency. If this is not the case, the coefficients of each a_n term would be adjusted accordingly to be $\frac{f_n}{f_0}$, where f_n is the frequency of the

nth partial. It is also worth noting that the synthesis equation can be rendered independent of the fundamental frequency. This is because the weightings in this equation are dependent on the components' relationships to the fundamental frequency (what integer multiple of the fundamental they represent), but are independent of what that fundamental frequency is. Thus, when the user specifies a harmonic spectral centroid, she can do so in normalized form ($\overline{hsc} = \frac{hsc}{f_0}$) or in denormalized form (as in equation 4.6) [33].

4.1.4 Harmonic Spectral Standard Deviation

The next descriptor dealing with the harmonic components of the spectrum at a specific instant is harmonic spectral standard deviation. Harmonic spectral standard deviation is computed over the duration of a sound, using the same frame-by-frame average via the Short Time Fourier Transform as the harmonic spectral centroid. It measures the amplitude-weighted standard deviation of the harmonic components of the spectrum, normalized by the harmonic spectral centroid. Thus, harmonic spectral standard deviation measures how much of the spectral energy is concentrated at the harmonic spectral centroid. A spectrum with all of its energy concentrated in one harmonic (i.e. a pure sine wave) would have no harmonic spectral standard deviation. As the energy in the spectrum becomes more evenly distributed and is thus less concentrated around the harmonic spectral centroid, the harmonic spectral standard deviation increases. For a comparison of spectra with the same harmonic spectral centroid, but different harmonic spectral standard deviations, see Figure 4.4. The analysis equation for harmonic spectral standard deviation is

$$ihsstd = \sqrt{\frac{\sum\limits_{harmo=1}^{nb\_harmo} [A(frame, harmo) * (f(frame, harmo) - ihsc(frame))]^2}{\sum\limits_{harmo=1}^{nb\_harmo} A^2(frame, harmo)}} * \frac{1}{ihsc(frame)} \tag{4.7}$$

where A is the amplitude of the given harmonic, f is the frequency of the harmonic, and $ihsc$ is the harmonic spectral centroid of the frame.

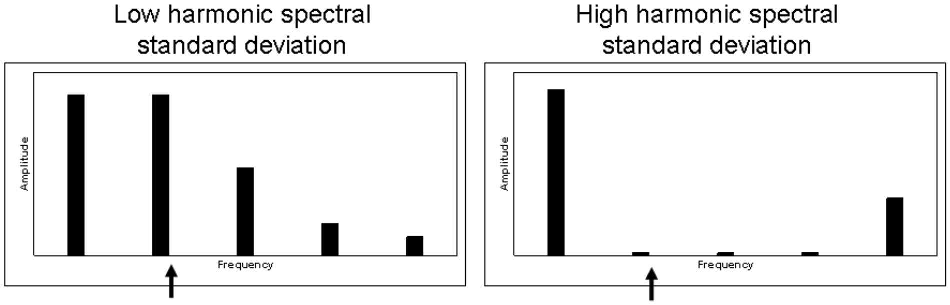


Figure 4.4: Two harmonic spectra with different harmonic spectral standard deviations, but the same harmonic spectral centroid, as indicated by the arrows

Because it is a calculation of standard deviation, harmonic spectral standard deviation includes quadratic terms.[1] Much as the terms enclosed in absolute

[1] Standard deviation is generically defined as $\sigma = \sqrt{E(X^2) - (E(X))^2}$ where $E(X)$ is the expected value of X. In our case, the expected value is the harmonic spectral centroid.

value signs in the calculation of harmonic spectral deviation prevented the further simplification of *hsd*'s synthesis equation, the quadratic terms of harmonic spectral standard deviation would make its synthesis equation somewhat complex. Without any modifications, harmonic spectral standard deviation would give the following synthesis equation

$$\frac{\left[(a_1 * (f_0 - hsc))^2 + (a_2 * (2f_0 - hsc))^2 + \ldots (a_n * (nf_0 - hsc))^2\right]}{a_1^2 + a_2^2 + \ldots a_n^2} = (hsc{*}hsstd)^2 \tag{4.8}$$

where hsc is the user-specified harmonic spectral centroid, $hsstd$ is the user-specified harmonic spectral standard deviation, f_0 is the user-specified fundamental frequency, and a_n is the amplitude of the nth harmonic. Since the user will specify harmonic spectral centroid and fundamental frequency, this equation can be simplified to

$$\frac{(C_1a_1)^2 + (C_2a_2)^2 + \ldots (C_na_n)^2}{a_1^2 + a_2^2 + \ldots a_n^2} = (hsc * hsstd)^2 \tag{4.9}$$

where C_n is a known value calculated as the number of the partial times the fundamental frequency, minus the harmonic spectral centroid [33].

Even after reworking this synthesis equation, however, there is still no guarantee that the quadratic constraints will all be convex.[2] This creates an obstacle

[2] In order to be convex, in the generic quadratic constraint $\frac{1}{2}x^TQ^kx + \sum_{j=0}^{n-1} a_{kj}x_j \leq u_k^c$, the matrix Q^k must be positive semi-definite.

because solving optimization problems with non-convex quadratic constraints is far more difficult than solving problems with strictly convex constraints. To avoid this potential problem, I modify the synthesis equation, creating a slightly different synthesis equation which defines what I label harmonic spectral spread (*hss*). First, because spread is always positive, I ensure that the weighting terms, C_1, $C_2 \dots C_n$ will be positive. This is easily accomplished by retaining the exponent such that the coefficients of the numerator are C_1^2, $C_2^2 \dots C_n^2$. I can then modify the equation by eliminating the exponents on the terms a_1, $a_2 \dots a_n$, as well as the exponent on $hsc * hsstd$. There is no question that all of these values will be positive (the amplitudes of each harmonic, the harmonic spectral centroid, and the harmonic spectral spread), so the exponents serve only to change the scale of the measurement and can safely be removed.

This results in the following synthesis equation

$$\frac{(C_1^2 a_1) + (C_2^2 a_2) + \dots (C_n^2 a_n)}{a_1 + a_2 + \dots a_n} = hsc * hss \tag{4.10}$$

In this equation, the denominator now represents amplitude, rather than energy (which is equal to the amplitude squared). And while this change will affect the scale of the standard deviation calculation, the underlying representation of how broadly spread the spectrum is, normalized by the harmonic spectral centroid, is unaffected. Finally, because $a_1 + a_2 + \dots a_n$ is equivalent to A_{Total}, the user-defined total amplitude, one more transformation gives the final synthesis equation for harmonic spectral standard deviation as

$$(C_1^2 a_1) + (C_2^2 a_2) + \dots (C_n^2 a_n) = hsc * hss * A_{Total} \tag{4.11}$$

This equation is not the exact analog of the harmonic spectral standard deviation analysis equation as defined for MPEG-7, but this new synthesis equation serves the same purpose, giving the user control over harmonic spectral spread, while rendering the implementation of the synthesis method easier and more robust.

4.1.5 Harmonic Spectral Deviation

Like harmonic spectral centroid and harmonic spectral standard deviation, harmonic spectral deviation is computed as an average over the duration of the sound. It too is calculated for each frame of a Short Time Fourier Transform and averaged. Harmonic spectral deviation represents the sum of the deviation of each harmonic's amplitude from a global spectral envelope, as determined by the two adjacent harmonics. This is to say that, for each harmonic (excluding the fundamental and the highest harmonic), an average is taken of the amplitudes of that harmonic and the two adjacent harmonics. The harmonic spectral deviation for the given harmonic is its absolute distance from that average. This is expressed as

$$ihsd(frame) = \sum_{harmo=2}^{nb\_harmo-1} \left| A(frame, harmo) - \frac{\sum_{i=-1}^{1} A(frame, harmo+i)}{3} \right| \tag{4.12}$$

where A is the amplitude of a given harmonic in that frame. For a comparison of spectra with the same harmonic spectral centroid, but different harmonic spectral deviations, see Figure 4.5.

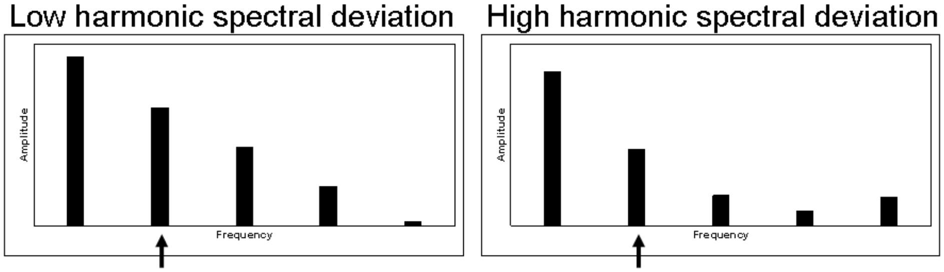


Figure 4.5: Two harmonic spectra with different harmonic spectral deviations, but the same harmonic spectral centroid, as indicated by the arrows

By virtue of being the sum of a series of absolute values, harmonic spectral deviation is non-linear.[3] This complicates its conversion into a synthesis equation. Since one does not know, *a priori*, whether the terms contained within each absolute value sign will be negative or positive, it is not possible to expand and simplify the terms as with harmonic spectral centroid. Thus, the simplest form

[3] Equations involving absolute value are usually non-linear around the origin. For example, the line $y = |x|$ is reflected around the y-axis because y always gives a positive value.

of the synthesis equation for harmonic spectral deviation is

$$\left|a_2 - \frac{a_1 + a_2 + a_3}{3}\right| + \left|a_3 - \frac{a_2 + a_3 + a_4}{3}\right| + \ldots \left|a_{n-1} - \frac{a_{n-2} + a_{n-1} + a_n}{3}\right| = hsd \tag{4.13}$$

where hsd is the user-defined harmonic spectral deviation and a_n is the amplitude of the nth harmonic. As with harmonic spectral centroid and spread, there are many different spectra (actually, an infinite number) that could all satisfy a given harmonic spectral deviation [33].

4.1.6 Spectral Centroid

The sixth descriptor is the one that defines the rest of the spectrum (i.e. the inharmonic spectrum): spectral centroid. For MPEG-7, spectral centroid is computed from a power spectrum calculated from an averaged periodogram. The MPEG-7 specification suggests that the Fourier transform used to compute the spectral centroid be calculated with a resolution providing 1024 “bins.” The calculation of spectral centroid done using the analysis equation

$$sc(frame) = \frac{\sum_{k=1}^{powerspectrum\_size/2} f(k) * S(k)}{\sum_{k=1}^{powerspectrum\_size/2} S(k)} \tag{4.14}$$

where $f(k)$ is the frequency and $S(k)$ is the power measure of the kth bin.

In transforming the spectral centroid analysis equation into a synthesis equation, it is important to note that while the spectral centroid is only calculated for 512 specific frequencies, it is expected that the centroid is accurate for the entire spectrum. Thus, when synthesizing a spectrum, it is fine to do so specifically for the 512 bins represented in the synthesis equation, but to be "realistic," these values should then be used to interpolate spectral information between those bins. For a comparison of spectra with different spectral centroids, see Figure 4.6. To obtain amplitude values for the 512 frequency bins, the analysis equation is transformed into the following synthesis equation

$$S_1 + 2S_2 + 3S_3 + \ldots 512S_{512} = \frac{sc * S_{Total}}{2f_0} \tag{4.15}$$

where sc is the user-defined spectral centroid, S_{Total} is the user-defined total power spectrum, f_0 is the user-defined fundamental frequency, and S_k is the power measurement for the kth bin [33].

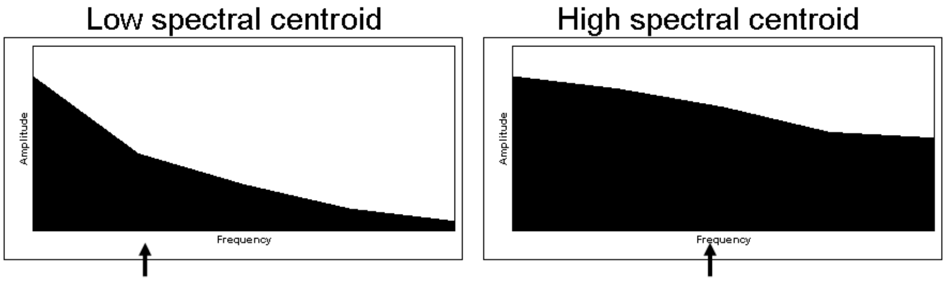


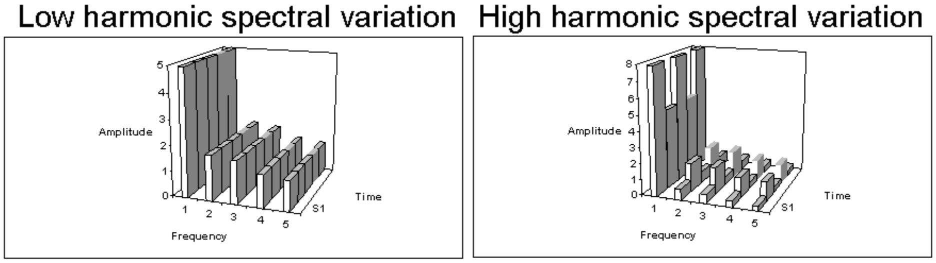
Figure 4.6: Two spectra with different spectral centroids, as indicated by the arrows

4.1.7 Harmonic Spectral Variation

The final descriptor is the only spectro-temporal one: harmonic spectral variation. Like the other harmonic spectral descriptors, harmonic spectral variation is calculated as a mean over the length of the sound. However, unlike the other descriptors, harmonic spectral variation compares each frame to the previous frame of a Short Time Fourier Transform. Harmonic spectral variation is sometimes described as measuring the "flux" or "coherence" of a sound—how the shape of the spectrum changes through time. It can also be thought of as the mean of correlations calculated for the spectra of adjacent frames. Harmonic spectral variation is defined mathematically as

$$ihsv(frame) = \frac{\sum\limits_{harmo=1}^{nb\_harmo} A(frame-1, harmo) * A(frame, harmo)}{\sqrt{\sum\limits_{harmo} A^2(frame-1, harmo)}\sqrt{\sum\limits_{harmo} A^2(frame, harmo)}} \tag{4.16}$$

where $A(frame, harmo)$ is the amplitude of the harmonic for the current frame and $A(frame-1, harmo)$ is the amplitude of the same harmonic for the previous frame. Thus, in a sound that has no flux, like the spectrum on the left in Figure 4.7, each frame will correlate perfectly with the previous one, giving a harmonic spectral variation of 1. Spectra that vary over time, like the one on the right in Figure 4.7 will have a harmonic spectral variation that is less than 1.



Low harmonic spectral variation

High harmonic spectral variation

Figure 4.7: Two harmonic spectra in time

The analysis equation for harmonic spectral variation can be rewritten as the following synthesis equation

$$\frac{\left(a_{1,1} * a_{1,2} + a_{2,1} * a_{2,2} + \ldots a_{n,1} * a_{n,2}\right)^2}{\left(a_{1,1}^2 + a_{2,1}^2 + \ldots a_{n,1}^2\right)\left(a_{1,2}^2 + a_{2,2}^2 + \ldots a_{n,2}^2\right)} = hsv^2 \tag{4.17}$$

where hsv is the user-specified harmonic spectral variation, $a_{n,1}$ is the amplitude of the nth harmonic in the previous frame's spectrum and $a_{n,2}$ is the amplitude of the nth harmonic in the current frame's spectrum. The complication with this equation, as compared to the previous synthesis equations, is that it defines two successive spectra. However, since calculations of successive spectra can take place iteratively in the synthesis process, the synthesis equation would be significantly simpler in a case where the spectrum of the previous frame was known. In this case, it would be

$$\frac{(a_1' a_1 + a_2' a_2 + \ldots a_n' a_n)^2}{A' * (a_1^2 + a_2^2 + \ldots a_n^2)} = hsv^2 \tag{4.18}$$

where a'_n is the known value of the nth harmonic in the previous frame and A' is the known sum of the squared amplitudes of the harmonics from the previous frame [33].

As with harmonic spectral standard deviation, this synthesis equation, if treated as a constraint, would be quadratic, not linear. And like the constraint from harmonic spectral standard deviation, this quadratic constraint is non-convex. Again, however, the constraint can be modified to get at the same information with a simpler process. In this case, that means modifying the original analysis equation so that the denominator of the fraction looks at amplitude, not energy. Thus, the modified analysis equation—which I distinguish from *hsv* by labeling it as harmonic spectral flux (*hsf*)—would be

$$\frac{\sum_{harmo=1}^{nb\_harmo} A(frame-1, harmo) * A(frame, harmo)}{\sum_{harmo}^{nb\_harmo} A(frame-1, harmo) \sum_{harmo}^{nb\_harmo} A(frame, harmo)} = hsf(frame) \quad (4.19)$$

After reworking, this results in the following synthesis equation

$$\frac{a'_1 a_1 + a'_2 a_2 + \ldots a'_n a_n}{A' * (a_1 + a_2 + \ldots a_n)} = hsf \quad (4.20)$$

where A' is the total amplitude of the previous frame, a'_n is the amplitude of the nth harmonic in the previous frame, and hsf is the user-specified harmonic spectral flux. Just as with harmonic spectral spread, this equation can be simplified further because $a_1 + a_2 + \ldots a_n$ is equivalent to A_{Total}. Thus, the final synthesis equation for harmonic spectral flux is

$$a'_1a_1 + a'_2a_2 + \ldots a'_na_n = hsf * A' * A_{Total} \tag{4.21}$$

These modifications change the scale on which variation is measured, but since timbral synthesis is only concerned that the equation defines variation, not with the scale on which variation is measured, this change is relatively trivial. That said, it should be noted that two successive spectra that are identical will not give a harmonic spectral flux of 1.

4.2 Pre-Synthesis Processing

Having addressed the transformation of each timbral descriptor equation from an analysis equation to a synthesis equation, I now turn to the pre-synthesis processing—the heart of timbral synthesis. (Figure 4.8 provides an overview of this process and the synthesis process that follows it.) The primary step in this process is the definition of an amplitude envelope and a spectrum for the requested note that conform to the user-defined timbral descriptor values. This is not a one-to-one transformation; for a given set of descriptor values there are many amplitude envelopes and spectra that would meet those limited criteria. Thus, the first task in the pre-synthesis processing phase is to more precisely define the tone the user seeks to synthesize.

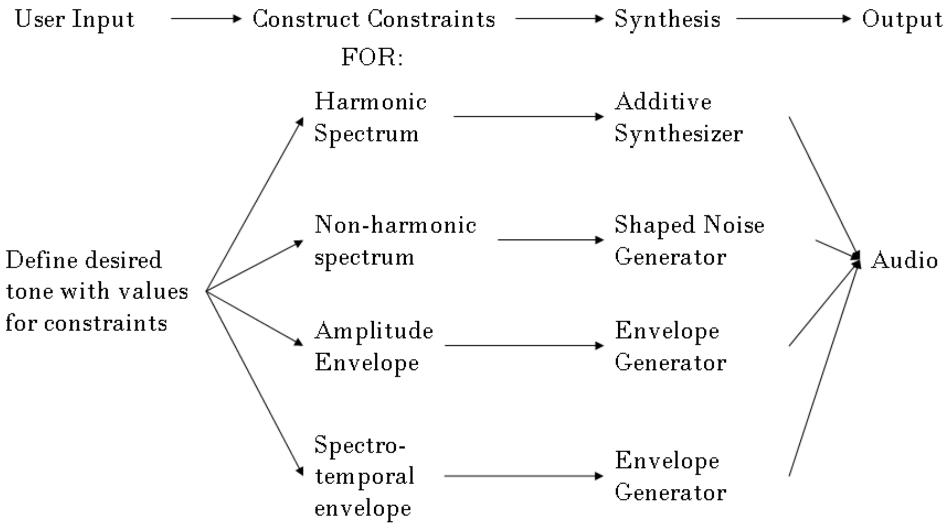


Figure 4.8: A step-by-step overview of the timbral synthesis process

Each additional piece of information about the sound to be synthesized reduces the solution space in which the desired sound can be found. This is not to say that a poorly defined solution space could not yield interesting results. One could imagine a system that required the user to specify harmonic spectral centroid but not fundamental frequency, and the broad range of pitches that acceptable tones would span. However, for the current implementation, my goal was to give the user maximum control over the sound he or she produces, while maintaining manageability. As such, in addition to specifying values for the timbral descriptors, the user is also required to set the fundamental frequency and duration of the note, the number of harmonics present, and the total amplitudes of the harmonic and non-harmonic spectra.

Furthermore, although the timbre space in MPEG-7 for harmonic, sustained tones only includes five descriptors (*lat*, *hsc*, *hsd*, *hsstd*, and *hsv*), timbral synthesis asks the user to give values for seven. The additional two descriptors (*sc* and *tc*) are helpful since spectral centroid helps define the shape of the non-harmonic components of the spectrum and temporal centroid helps give shape to the amplitude envelope that would otherwise be defined only by log-attack time and duration. Thus, in total, timbral synthesis asks the user to define twelve values for each tone desired. Because the values for these descriptors affect and constrain the possible values for other descriptors, timbral synthesis incorporates each in sequence, roughly in their order of perceptual salience, as defined by the timbre studies described in section 2.2. This process is explained in more detail below.

4.2.1 The Amplitude Envelope

The tasks of defining an amplitude envelope and spectrum are independent of each other. In order to define the amplitude envelope of a note, timbral synthesis relies on the user-specified log-attack time, temporal centroid, and duration. With these three equations that describe the envelope, timbral synthesis can solve for up to three variables. This implementation of timbral synthesis uses an attack-sustain-release (ASR) envelope in which the variables solved for are the respective lengths of the three portions of the envelope.

None of these variables is wholly independent of the others. For example, the log-attack time must be shorter than the duration. Similarly, for a given duration and log-attack time, only some temporal centroids are possible. As such, I prioritize the treatment of these interdependent variables. For the amplitude envelope, log-attack time, being the most salient psychoacoustic parameter, is given the highest priority. Then, because it is a timbral parameter (and this is a timbral synthesizer), temporal centroid is given second priority. This means that if a user-entered parameter must be modified to accommodate the other parameters, this implementation first modifies duration and then, if necessary, temporal centroid.

Since log-attack time directly describes the length of the attack portion of the amplitude envelope, the first step in generating an envelope is calculating this length. The initial portion of the envelope is then constructed as a line defined by the equation $y = \frac{x}{10^{lat}}$ for $0 \leq x \leq 10^{lat}$. The next step is to calculate the minimum temporal centroid (in which the attack is followed immediately by the beginning of the release, with no sustain) and the maximum temporal centroid (in which the attack is followed by a long sustain and a sudden release at the very end of the duration) for the attack time and duration requested by the user. The minimum temporal centroid is defined as

$$\min tc = \frac{\displaystyle\sum_{x=0}^{duration} x * y}{\displaystyle\sum_{x=0}^{duration} y}$$

where $y = \dfrac{x}{10^{lat}}$ for $0 \leq x \leq 10^{lat}$ and $y = \dfrac{duration - x}{duration - 10^{lat}}$ for $10^{lat} \leq x \leq duration$. The maximum temporal centroid is defined as

$$\max tc = \frac{\displaystyle\sum_{x=0}^{duration} x * y}{\displaystyle\sum_{x=0}^{duration} y}$$

where $y = \dfrac{x}{10^{lat}}$ for $0 \leq x \leq 10^{lat}$, $y = 1$ for $10^{lat} \leq x < duration$, and $y = 0$ for $x = duration$.

Assuming the requested temporal centroid falls within these bounds, the next step is to calculate the appropriate length for the sustain portion of the envelope such that the temporal centroid is equal to the one requested by the user. The remaining portion of the envelope is allocated to the release and is a linear drop-off from the amplitude in the sustain portion of the envelope to 0. Since the relationship between temporal centroid and sustain time is linear given a fixed log-attack time, timbral synthesis can calculate the sustain time that corresponds with the requested temporal centroid as

$$sustain = \frac{(tc - \min tc)\left(duration - 10^{lat}\right)}{\max tc - \min tc} \tag{4.22}$$

With the lengths of the sustain and release portions of the envelope computed, and the length of the attack portion of the envelope already derived, the complete ASR envelope is defined.

In cases in which the requested temporal centroid falls outside the bounds computed using the requested duration and log-attack time, this implementation modifies duration. As explained above, I privilege log-attack time and temporal centroid over duration and so the current implementation extends or shortens the requested duration—doing so as little as possible—until the requested log-attack time and temporal centroid are possible. Although, for the reasons stated, I privilege log-attack time and temporal centroid, changing the implementation such that duration would be privileged over temporal centroid and/or log-attack time would be simple.

4.2.2 The Average Harmonic Spectrum

In order to define the average spectrum of the desired sound, timbral synthesis relies on the fundamental frequency, the total amplitude of the harmonic spectrum, the number of harmonics, the harmonic spectral centroid, the harmonic spectral deviation, and the harmonic spectral spread (derived from harmonic spectral standard deviation above)—all defined by the user. The first two values do not affect the spectral shape. Fundamental frequency simply defines by how much to translate the harmonic portion of the spectrum, while total amplitude is a scalar by which the amplitude of the harmonic components of a normalized spectrum are multiplied. As such, for spectral shape, timbral synthesis relies entirely on three descriptors and the number of harmonics requested. Among the harmonic spectral descriptors, timbral synthesis privileges harmonic spectral centroid, since psychoacoustic studies have shown spectral centroid and log-attack time to be the most salient physical properties of sound.

Feasibility

The first step in creating a harmonic spectrum is making sure that a given set of descriptors define a realizable spectrum. Since the range of realizable harmonic spectral centroids is defined by the number of harmonics present, the first check is of the viability of this pair of parameters. The smallest theoretical normalized harmonic spectral centroid is always 1.0 (a sine wave at the fundamental) and the largest theoretical normalized harmonic spectral centroid is always equal to the number of harmonics present.[4] If the requested normalized harmonic spectral centroid is too high, the number of harmonics is increased; if it is too low, it is raised to 1.0.

[4] Although a spectrum with all its energy in the 10th harmonic and none in the fundamental would, in reality, be heard as a note having a fundamental ten times the "fundamental frequency" and a normalized harmonic spectral centroid of 1.0, rather than as a note at the fundamental frequency with a normalized harmonic spectral centroid of 10.0.

The range of realizable harmonic spectral centroids is also bounded by the amount of harmonic spectral spread available, so the second check compares these two descriptors. Harmonic spectral spread (the modified version of harmonic spectral standard deviation) defines how concentrated the energy of the spectrum is around the harmonic spectral centroid. A spectrum with no spread would have all of its energy concentrated at the harmonic spectral centroid. Thus, this check ensures that the user-specified harmonic spectral spread is sufficient to allow a spectrum with the requested harmonic spectral centroid to be generated. If the combination of the two descriptors is infeasible, the spread is adjusted until a spectrum is feasible.

The next feasibility check is of harmonic spectral deviation, which constrains the range of realizable harmonic spectral centroids and harmonic spectral spreads. Harmonic spectral deviation defines the non-linearity of the harmonic spectral components. As harmonic spectral deviation increases, the minimum achievable harmonic spectral centroid decreases and the maximum achievable harmonic spectral centroid increases. Because the harmonic spectral deviation synthesis equation does not distinguish where in a spectrum the deviation actually occurs, when calculating minima and maxima, timbral synthesis strategically places all of the deviation between the first and second harmonics for minima and between the penultimate and ultimate harmonics for maxima.[5]

The linear relationship between the amount of total harmonic spectral deviation and the size of the range of achievable harmonic spectral centroids continues until the harmonic spectral deviation is equal to one third of the total amplitude in the harmonic spectrum. When harmonic spectral deviation is equal to one third of the total amplitude, both the minimum and maximum harmonic spectral centroids—1.0 and the number of harmonics, respectively—are possible. We can see this because a spectrum with all of its energy in the first harmonic would have a harmonic spectral deviation equal to $\left|0 - \dfrac{totalAmp + 0 + 0}{3}\right|$, while a spectrum with all of its energy in the last harmonic would have a harmonic spectral deviation of $\left|0 - \dfrac{0 + 0 + totalAmp}{3}\right|$. For infeasible combinations of harmonic spectral centroid, harmonic spectral spread, and harmonic spectral deviation, the harmonic spectral deviation is increased to the exact amount needed to obtain the requested harmonic spectral centroid and harmonic spectral spread. In this way, the parameters of the sound produced are as close as possible to those requested by the user.

[5] This is because, for a given amount of harmonic spectral deviation, the spectrum with all of its deviation between the first and second harmonics will result in the lowest possible harmonic spectral centroid. Conversely, placing all of the deviation between the last two harmonics will result in the highest possible harmonic spectral centroid.

Constraint Problem

Once the input values have been checked for feasibility, the next step is to generate a spectrum. In some cases, the process of assessing feasibility may restrict the solution space. In fact, in the most extreme cases, the solution space may be reduced to a single point, in which case, formulating a constraint problem is unnecessary, since the spectrum has already been defined. However, in cases where the feasibility assessment leaves more than one spectrum that would meet the inputted values, the synthesis equations are used to draw up a linear constraint problem to identify the viable spectra.

Because in most cases there are many spectra that would meet all of the user-entered criteria, the constraint problem must not only define the bounds of the solution space, but also identify which of these spectra would be most "desirable." To do this, the linear constraint problem is formulated as a linearly constrained maximization problem in which the objective function (that which the solution works to maximize), serves to "point" the solver towards desirability. Thus, the objective function that timbral synthesis uses affects which of the many possible spectra is chosen as best (assuming there are many), but it will never cause the constraint problem to choose a spectrum that does not meet all of the user-specified criteria. The objective function can be thought of as "tuning" the spectrum that timbral synthesis creates. If, for example, a user sought the spectrum with the most energy in the fundamental that also met all of her timbral criteria, tuning the objective function so that the most weight is placed on the first term and little weight is placed on the other terms in the function the solver seeks to maximize would allow her to obtain this spectrum.

In this linear constraint problem, one of the constraints is generated using the user-specified total harmonic amplitude, one is generated using the user-specified harmonic spectral centroid, one is generated from the user-specified harmonic spectral spread, and a series of constraints are generated using harmonic spectral deviation. Because the synthesis equation for harmonic spectral deviation contains $n - 2$ non-reducible terms (see equation 4.8 above), where n is the number of harmonics present in the spectrum, each of those terms is formulated as its own linear constraint. The distribution of the total harmonic spectral deviation, *hsd*, among these terms, HSD_1, HSD_2, etc., is determined according to feasibility. Each of these *hsd* constraints governs three adjacent harmonics, ignoring all the others. In total, this gives $n+1$ constraints ($n-2$ constraints corresponding to *hsd*, one corresponding to total amplitude, one to *hsc*, and one to *hss*). Additionally, the constraint problem requires that all of the resulting amplitudes be positive, since negative amplitudes act the same as positive ones acoustically.

Thus, to calculate the spectrum, timbral synthesis solves the following optimization problem.

$$\begin{array}{lccccccccccc}
\text{Maximize} & na_1 & + & (n-1)a_2 & + & (n-2)a_3 & + & \ldots & a_n & & \\
\text{subject to} & a_1 & + & a_2 & + & a_3 & + & \ldots & a_n & = & totalAmp \\
 & a_1 & + & 2a_2 & + & 3a_3 & + & \ldots & na_n & = & \overline{hsc} * totalAmp \\
 & C_1^2 a_1 & + & C_2^2 a_2 & + & C_3^2 a_3 & + & \ldots & C_n^2 a_n & = & \overline{hsc} * hss * totalAmp \\
 & \frac{a_1}{3} & + & \frac{a_2}{3} & + & \frac{a_3}{3} & & & & = & HSD_1 \\
 & & & \frac{a_2}{3} & + & \frac{a_3}{3} & + & \ldots & & = & HSD_2 \\
 & \vdots & + & \vdots & + & \vdots & + & \ldots & \vdots & = & \vdots \\
 & & & & & & & \ldots & \frac{a_n}{3} & = & HSD_{n-2}
\end{array}$$

where

$$a_0, a_1, a_2 \ldots a_n \geq 0,$$

$$C_n = (nf_0 - \overline{hsc}),$$

$$\text{and} \quad HSD_1 + HSD_2 + \ldots HSD_{n-2} = hsd$$

4.2.3 The Non-Harmonic Spectrum

Having dealt with the harmonic spectrum, timbral synthesis can use the user-specified spectral centroid to calculate the rest of the spectrum. It is important that the harmonic spectrum be calculated before the non-harmonic spectrum because the non-harmonic spectrum must preserve the amplitudes at frequencies that are integer multiples of the fundamental. This is because any energy that the non-harmonic spectrum added to the harmonic spectrum would change the harmonic spectrum rendering it incompatible with the user-inputted harmonic descriptor values.

To generate the non-harmonic spectrum, timbral synthesis asks the user for two values: spectral centroid and total amplitude of the non-harmonic spectrum. The non-harmonic spectrum is represented as broadband noise, which is then shaped by filters according to the user-entered parameters. The broadband noise is assumed to roll off linearly and the spectral centroid is used to determine the slope of the rolloff. Thus, for example, a spectral centroid directly in the middle of the spectrum would mean that the broadband filter applied to the noise would have a slope of 0. A lower spectral centroid would necessitate a filter with a steeper, negatively sloped rolloff, while a higher spectral centroid would mean a filter with a steeper, positively sloped rolloff.

The other parameter, total amplitude, determines the overall level of the non-harmonic spectrum. Also, because the range of the broadband noise is not determined by spectral centroid, the total amplitude descriptor is used to calculate the point at which the filter has fully filtered the noise. Thus, the spectral centroid is used to determine the general shape of the non-harmonic spectrum and the total amplitude is used to scale that shaped noise.

Once the overall non-harmonic spectrum has been calculated, the final step in creating the overall spectrum is to account for the already-determined harmonic spectrum. The amount of energy that the non-harmonic spectrum contributes at each of the harmonics is minimized using a comb filter tuned to the fundamental frequency of the harmonic spectrum. In this manner, the actual amplitude at each of the harmonics is in line with the value that was calculated for the harmonic spectrum and thus, those values will fulfill the harmonic spectral descriptor values inputted by the user.

4.2.4 Spectral Flux

Like the creation of the non-harmonic spectrum, the incorporation of harmonic spectral flux is dependent on the creation of the average harmonic spectrum first. In order to incorporate flux, timbral synthesis uses two more linear optimization problems that use the calculated average spectrum as the weighting for the objective function. Both these linear optimization problems retain the constraints created from the other descriptors. In addition, they each incorporate a constraint representing harmonic spectral flux. In this way, timbral synthesis creates two new spectra—one "above" the average spectrum and one "below" it—that are the appropriate distance from the average spectrum[6]. When these two spectra are averaged, they result in the original spectrum.

[6] This distance is determined by the amount of harmonic spectral flux requested. More flux means they will be further from the original average spectrum

The two new constraints devised to calculate the varied spectra each use half of the requested variation. Because two perfectly correlated spectra no longer give a harmonic spectral variation of 1 (see section 4.1.7), the first step in incorporating flux is determining what value the average spectrum would give when correlated with itself. Once this value is determined, I calculate the requested flux's distance from that equilibrium as $dist = |flux - equil|$. Then, I divide that absolute distance between the two new constraints, keeping one positive and negating the other. Thus, the two flux values are $var + dist$ and $var - dist$, respectively.

From these values, I construct the two new constraint equations. The first is

$$a'_1 a_1 + a'_2 a_2 + \ldots a'_n a_n = \left(flux + \frac{dist}{2}\right)(A' * A_{Total})$$

and the second is

$$a'_1 a_1 + a'_2 a_2 + \ldots a'_n a_n = \left(flux - \frac{dist}{2}\right)(A' * A_{Total})$$

where $flux$ is the user-specified harmonic spectral flux, $dist$ is the absolute distance of that flux from the equilibrium for the specific average harmonic spectrum (as calculated above), A' is total amplitude of the average harmonic spectrum, A_{Total} is the total amplitude of the new spectrum being sought, and a'_n is the amplitude of the nth harmonic in the average harmonic spectrum. With these two new constraints, timbral synthesis can calculate the two new spectra needed to implement the spectral flux. To do this, I create one optimization problem that includes all of the previous constraints, as well as the first new flux constraint. Then I repeat that process, replacing the first flux constraint with the second new flux constraint. These two new spectra serve as the endpoints between which the spectral amplitudes fluctuate.

There remains a question of variation speed that is not well addressed by this procedure. Although changing the speed with which the harmonic spectrum varies between the two new spectra has no effect on the harmonic spectral descriptors, it does affect the value of the spectro-temporal analysis descriptor, harmonic spectral variation. This is because, no matter the speed of oscillation, the average values will remain the same. However, because harmonic spectral variation is calculated frame to frame, the method described above assumes a full period of oscillation every new frame. A slower oscillation frequency would result in less than the requested variation, while a faster oscillation speed would result in more than the requested variation. Giving the user more control over the oscillation speed, either directly or through the harmonic spectral flux descriptor, is an area that merits further research.

4.3 Synthesis

Having completed all the preceding calculations, timbral synthesis then moves into its final step: the synthesis process, which is outlined in Figure 4.9. The synthesis process is relatively simple, and is similar to the methods described by Serra [39]. The harmonic spectral components are reproduced by a series of sinusoidal oscillators. The amplitudes of these oscillators are varied between the two spectra calculated using the harmonic spectral variation. As noted above, the speed with which the amplitudes vary between their two values is not defined precisely in the current method, since it is dependent on the "frame rate." However, it should be noted that using a low frequency oscillator, at a sub-audio frequency, is preferable, since varying the oscillator amplitudes more quickly could introduce unwanted effects of amplitude modulation.

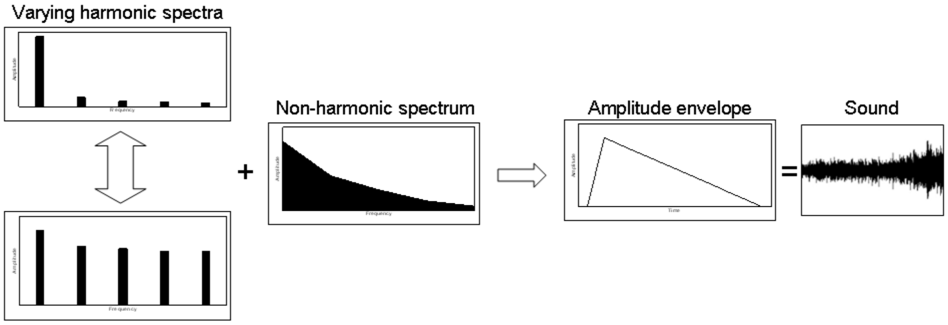


Figure 4.9: Overview of the sound synthesis process

The non-harmonic spectrum is generated using a white noise, broadband noise generator. This noise generator is then run through a filter (or a series of filters) to achieve the shape defined by the spectral centroid and total amplitude for the non-harmonic spectrum and is comb filtered to remove the energy at harmonic frequencies. The output of this shaped noise is added to the output of the bank of oscillators producing the harmonic spectrum. Finally, this complete spectrum is controlled by the amplitude envelope created using the duration, temporal centroid, and log-attack time values inputted by the user. This amplitude envelope scales the outputs of the harmonic and inharmonic spectra to produce, finally, the synthesized note requested by the user.

Chapter 5

Implementation

The current implementation of timbral synthesis, like the theory behind the process, is modular. The software is currently very simple, but by replacing one or more modules, the functionality of the software could be greatly expanded and tailored to a user's specific purposes. This modularity extends not only to the code, but also to the software packages and APIs used in this version. As mentioned in the Tools chapter, the JSyn synthesis engine, Mosek linear optimizer, or even the whole Java framework could be replaced without fundamentally changing the nature of the software.

The current version of the software is intended as proof of the concept of timbral synthesis. This means that it implements the theory described above and gives interesting results, but lacks advanced features that would be useful for more extensive applications. Some of these features are discussed in the future work section below, but suffice to say that, because timbral synthesis could be implemented for one of several different purposes, and each of these purposes would have different needs, a fuller implementation would need to take the specific goals of the project into account to best make use of timbral synthesis' abilities.

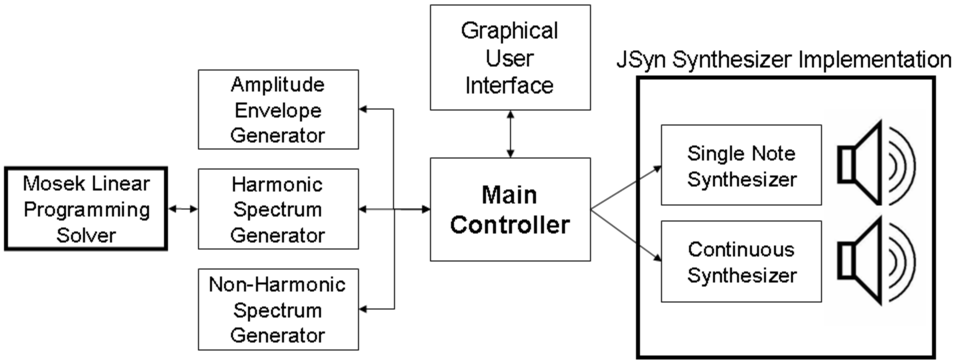

Graphical User Interface
JSyn Synthesizer Implementation
Amplitude Envelope Generator
Mosek Linear Programming Solver
Harmonic Spectrum Generator
Main Controller
Single Note Synthesizer
Continuous Synthesizer
Non-Harmonic Spectrum Generator

Figure 5.1: Modular implementation of timbral synthesis

The software generally conforms to the same structure outlined in the Timbral Synthesis chapter and modeled in Figure 5.1. A main controller directs the interaction of the different modules. It takes user-entered values from a graphical user interface, and passes those values to the modules that calculate the amplitude envelope and the harmonic and non-harmonic spectra. The harmonic spectrum module makes use of the Mosek linear programming solver to deal with the linear constraint optimization problems it generates. The main controller then receives the spectra and envelopes back from the generators and passes them to the synthesizer, which produces the sound output using the JSyn synthesis modules. At the same time, the main controller passes data back to the GUI showing the descriptor values of the current output. This is so that the user can see what, if any, modifications were made to the requested values to solve problems of infeasibility.

5.1 Main Controller

The main controller is responsible for all traffic control in the software. The current implementation operates in two modes, Single Note and Continuous. In Single Note mode, the values for a note are entered and the note is produced. Until the note has finished sounding, the main controller will not begin the process of creating the next note. In Continuous mode, on the other hand, the main controller begins playing a note matching the initial values entered by the user, but continues to play that note until the user modifies or stops it. The user can modify as many values as she wants before sending the new values to the main controller, which then calculates the new spectrum and changes the sound to match the new values. This modification process can happen an unlimited number of times before the user stops playback. Also, when in Continuous mode, the main controller ignores the values for temporal centroid and duration (since the attack portion of the envelope is retained, but then the sustain is used until the user requests the note stop).

5.2 Envelope and Spectrum Generators

These generators constitute the heart of any timbral synthesis software. The amplitude envelope is generated by a distinct set of classes that checks the user-inputted values for feasibility, adjusts them if the envelope requested is infeasible, and then generates the amplitude envelope. As noted in the Timbral Synthesis chapter, the envelope produced is an Attack-Sustain-Release envelope, which is represented as an array of length 8 consisting of four time-amplitude pairs.

The spectrum generator classes are of two types. The first type is responsible for creating the initial, average spectrum, while the second type incorporate harmonic spectral variation to find the endpoint spectra between which the tone should vary. In the same manner as the amplitude envelope generators, the spectrum generators prioritize descriptors, checking for feasibility with the addition of each new descriptor and modifying those descriptor values as necessary to keep the solution space from shrinking to zero. Specifically, this means prioritizing harmonic spectral centroid, followed by the number of partials, followed by harmonic spectral spread, followed by harmonic spectral deviation.

The spectrum generators that incorporate harmonic spectral variation only need to check the feasibility of the variation constraint itself, since the full combination of the other constraints has already been checked during the generation of the original, average spectrum. All of the feasibility checks during the generation of the spectra can happen very efficiently because of the linear solvers. These solvers already can handle optimization problems on the order that timbral synthesis requires very quickly, but their feasibility checks happen even more quickly, since infeasible problems are identified by a pre-solver. This means that feasibility checks and the subsequent adjustments to the optimization problems to render them feasible can happen iteratively without introducing a major source of latency.

5.3 Synthesizer

As outlined above, the current software operates in two modes. Each of these modes has its own synthesizer. The Single Note synthesizer receives all of the information it needs to synthesize a note, including the initial harmonic spectrum, the endpoints spectra between which the harmonic spectrum will vary, the non-harmonic spectrum, and the full amplitude envelope, before it makes a sound. The Continuous synthesizer receives the initial harmonic spectrum, the endpoint spectra, the non-harmonic spectrum and the attack portion of the envelope. It then begins synthesis by reading through the attack envelope and remaining in the sustain portion of the envelope. This continues until the user stops synthesis.

Any modifications the user requests are implemented by changing the shape of the spectra while continuing to remain in the sustain portion of the envelope. The Continuous synthesizer has one other feature—a "playing" flag—that tells the main controller whether a sound is already playing so that the main controller knows whether a new spectrum should be implemented as the start of a new continuous tone or as the continuation of an already existing one.

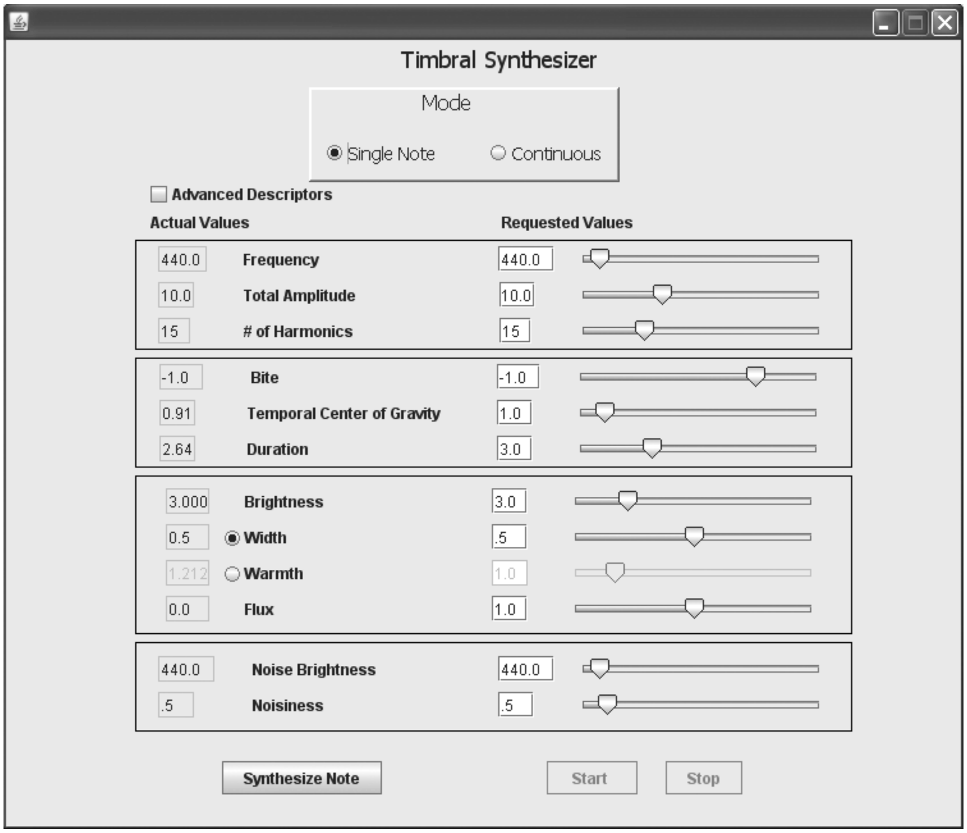
5.4 Graphical User Interface

The graphical user interface, programmed using Java Swing components, allows the user to specify values for the descriptors using sliders, or by entering the values directly into the text fields in the column in the middle of the GUI. The column of text boxes to the left shows the actual descriptor values for the last sound created (or the current sound in Continuous mode). Thus, a mismatch between the text boxes in the different columns alerts the user that her requested sound was infeasible and shows her what modifications were made to make the sound feasible. The GUI functions almost identically in both Single Note and Continuous modes. The only differences are that the available control buttons switch depending on the mode selected and the temporal centroid and duration controls are inactive in Continuous mode.

5.5 User Parameter Mapping

The labels on the GUI's descriptor controls are switchable. The "advanced descriptors" (those used throughout this paper) and the "beginner" descriptors can be used alternately, as controlled by the Advanced Descriptors checkbox. Figure 5.2 shows the GUI with the beginner descriptors, while Figure 5.3 shows the GUI with the advanced descriptors in place. This feature is helpful because the intended user of this software is unknown. Although timbre researchers would likely be comfortable with the technical descriptor names (e.g. harmonic spectral centroid, log-attack time, etc.), a composer likely would not be. To make this implementation as accessible as possible to a broad range of users, adjectives that correspond to each of the non-self-evident descriptors can be substituted in place of the technical terms. As noted above, though, different versions of timbral synthesis software can be modified to suit the needs of their intended audiences. Thus, a version intended solely for use by timbral researchers might use the technical terms exclusively.

The beginner descriptors used in this implementation's GUI were drawn from a number of sources. The use of two—brightness and flux—is generally accepted and can be found extensively throughout the literature. The correspondence of two others—bite and warmth—with their respective physical characteristics, is



Timbral Synthesizer

Mode

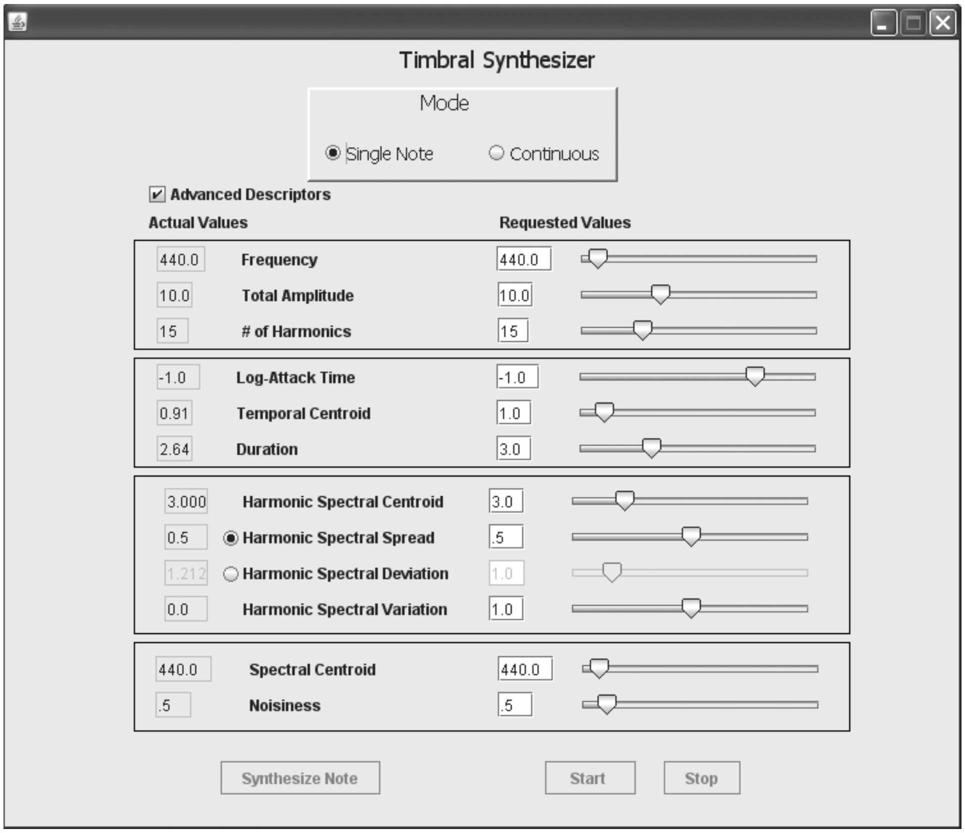
● Single Note ○ Continuous

□ Advanced Descriptors

Actual Values		Requested Values
440.0	Frequency	440.0
10.0	Total Amplitude	10.0
15	# of Harmonics	15
-1.0	Bite	-1.0
0.91	Temporal Center of Gravity	1.0
2.64	Duration	3.0
3.000	Brightness	3.0
0.5	● Width	.5
1.212	○ Warmth	1.0
0.0	Flux	1.0
440.0	Noise Brightness	440.0
.5	Noisiness	.5

Figure 5.2: Graphical user interface, beginner descriptors

suggested by Beauchamp [1, pg. 1]. The remaining two—width and center of gravity—were assigned based on my general impression of their influence on the sound. These assignments are sufficient for now, but, precisely because timbral synthesis allows users to modify one parameter of a sound while holding the others constant, more generalized adjectival descriptors could easily be assigned. A simple experiment to this end could ask several subjects to compare sounds with high and low values for each descriptor and to describe the dimension on which the sounds differed.



Timbral Synthesizer

Mode

● Single Note ○ Continuous

☑ Advanced Descriptors

Actual Values		Requested Values
440.0	Frequency	440.0
10.0	Total Amplitude	10.0
15	# of Harmonics	15
-1.0	Log-Attack Time	-1.0
0.91	Temporal Centroid	1.0
2.64	Duration	3.0
3.000	Harmonic Spectral Centroid	3.0
0.5	● Harmonic Spectral Spread	.5
1.212	○ Harmonic Spectral Deviation	1.0
0.0	Harmonic Spectral Variation	1.0
440.0	Spectral Centroid	440.0
.5	Noisiness	.5

Figure 5.3: Graphical user interface, advanced descriptors

Chapter 6

Conclusions

6.1 Applications

A fully developed timbral synthesizer is a versatile tool that could be of use to composers as well as psychoacoustic and signal processing researchers. Because timbral synthesis provides manageable, precise controls over the additive synthesis process, it is a powerful tool for composers looking for new, perceptually meaningful synthesis methods. Because it allows the precise specification of points in a timbral space, timbral synthesis could be a valuable tool for researchers exploring the psychoacoustics of timbre. And because timbral synthesis relies on a very reduced set of control instructions, it could prove useful to those working to resynthesize, morph and compress existing sounds.

6.1.1 Artistic Applications

The original impetus behind the creation of this new method of timbral synthesis was artistic. Despite the many years of work synthesizing sound with analog synthesizers and, more recently, with computers, the landscape of synthesis methods available to composers still leaves much to be desired. As outlined in the Related Work chapter, each type of synthesis currently available to composers has drawbacks. And while the palette of sounds that a composer can easily create is far broader than it used to be, timbral synthesis, by virtue of its manageable, perceptually relevant control interface, could be a helpful addition to many composers' array of techniques.

One advantage that timbral synthesis offers is the possibility of creating families of sounds. Like physical modeling, timbral synthesis is ideal for creating a group of similar, related sounds or even a continuum of sounds that vary on one or more perceptible sonic dimensions. In physical modeling, this grouping is accomplished by holding a particular parameter, like the vibrating material, constant. In timbral synthesis, grouping might be accomplished by differentiating between different timbres on one particularly salient dimension, like log-attack time. In this manner, sounds with quick attack times would remain grouped in the mind of the listener, even while individual sounds within the group might vary on other dimensions.

Another area in which timbral synthesis excels is the close relationship of its controls with the resultant sound. Because the controls are very intentionally based on perceptual models, the sounds produced from those controls vary in predictable ways. This means that tweaking a sound in timbral synthesis produces predictable results. Not surprisingly, this property is important to composers because it means that given a sound that's almost right, the composer can make a small change and know he is going to get a sound that is only slightly changed.

Finally, although these improvements and potential areas for future work will be discussed more fully in the next chapter, it is worth mentioning a few additions to the current software that would make it a much more functional synthesis tool for composers. The first is a score entry mode. The current modes are sufficient for experimentation, but creating a piece, or even a series of note events, is tedious without a way to notate multiple events at once. The second is the addition of recording capability. This improvement would be relatively simple given the current framework, since JSyn supports writing synthesized tones to disk. However, without this ability, any timbres created via timbral synthesis are necessarily ephemeral.

6.1.2 Research Applications

The biggest advantage that timbral synthesis offers to researchers in psychoacoustics is the ability to precisely locate a sound in a given timbre space. Unlike the machine learning-based timbral synthesis methods discussed in chapter 2, the timbral synthesis method proposed in this paper provides the same precision that the analysis of an existing sound does. This method is also easily adaptable to different timbre spaces, meaning that researchers can restrict or expand the number of descriptors at play for their particular experimental needs.

This precision also means that researchers can hold all but one value constant. The isolation of a single variable by this method would mean that researchers could investigate the relative contributions of different descriptors, furthering our knowledge of the relative salience of the different aspects of timbre. Furthermore, since timbral synthesis can produce many different sounds for a given group of values by varying the weightings in the objective function, researchers could use timbral synthesis to explore the differences among these sounds. This is a particularly important area to research because these sounds would occupy the same point in the timbre space. If they don't all sound identical (and preliminary experiments already indicate they don't), comparing these sounds might reveal further dimensions that we use to differentiate between timbres.

Another application that timbral synthesis could be used for by timbre researchers is a timbre matching experiment. In this type of experiment, which has been used extensively in researching our perception of color (another multidimensional percept), subjects would be given a fixed target sound and an adjustable sound. Then, they would be asked to move several parameters of their adjustable sound, which in this case would be the control values for a timbral synthesizer, until their sound matched the target sound. In color research, these experiments have shed new light on researchers' understanding of our perception and one could imagine similar results for timbre, especially in exploring the relative salience of different parameters.

6.1.3 Resynthesis, Morphing and Compression Applications

The third possible area of use for timbral synthesis is in several digital signal processing contexts. These applications are probably further down the road and would require more refined timbral synthesis tools, but none is too hard to imagine. One of these possible applications is in the resynthesis of existing tones. Although the limited number of descriptors in timbral synthesis probably means that a precise recreation of a particular sound won't be possible, creating a sound of a specific type is well within the capabilities of timbral synthesis. And further refinements of our understanding of timbre may well render resynthesis applications more feasible. A more precise understanding of exactly which elements are salient in making a violin sound like a violin would further our ability to discard those elements that are not salient and resynthesize a tone from the remaining parameters.

Another digital signal processing application would be in morphing or shaping existing sounds. This could mean combining two existing sound by locating them in a timbre space and then synthesizing sounds whose timbres fall along the line between the two original sounds. It could also mean taking an existing sound and resynthesizing a similar tone with one or more timbral property modified. Thus, for example, a one could create a super-bright trumpet by analyzing a trumpet tone and resynthesizing its spectrum with the brightness (spectral centroid) boosted. One could even do this for each of a series of analysis frames so that detailed spectro-temporal information is retained. By brightening the spectrum of each frame, but holding all the other timbral descriptor values constant, all aspects of the trumpet's character would be retained except for its brightness.

Finally, timbral synthesis could play a role in compressing the amount of information needed to represent a particular sound. Since timbral synthesis is so intimately concerned with issues of perception, it provides a useful tool for determining exactly how much and which information is needed for a perceptually lossless representation of timbre. Though not likely to play a role in the compression of audio signals, timbral synthesis could be helpful in finding reduced sets of control data that, despite their compressed character, sufficiently specify how the desired sound should sound.

6.2 Future Work

Because timbral synthesis of the type proposed in this paper is a largely novel procedure, there are myriad avenues that might be explored in the near future. These explorations could further both the practical application of the ideas behind timbral synthesis and the theory itself. One clear area that could use further work is the prototype software implementation.

6.2.1 Software

Although there are too many ways to list in which the software might be improved, a couple that leap immediately to mind are enumerated here. As noted in the previous chapter, the ability to specify a series of tones, rather than a single discrete tone or a single continuous tone, is a must if timbral synthesis is going to be useful to create music. Specifying multiple tones that will sound simultaneously is another obvious area for improvement. Like the earliest synthesizers, the current timbral synthesis software is monophonic, but this could and should be remedied so that polyphonic timbral synthesis is available.

The other suggestion from the previous chapter, that the synthesizer offer a recording mode, is also an easy and necessary improvement. A simple set of transport controls, as well as the ability to save control data and the audio data produced in a session, would quickly move the software from its current proof-of-concept state to a serious tool that could be put to use by musicians.

Other than these areas, work making the computer code that underlies the software more robust and scalable would be worthwhile. Since the software was implemented only as a prototype, it makes extensive use of static methods that might hinder its expansion. Also, as the current implementation relies on two external libraries (Mosek and JSyn), users are currently required to download and install those packages before the timbral synthesis software will function. Finding replacements that can easily be bundled with the timbral synthesis software itself would ease distribution.

Finally, the software could be reworked so that it can be distributed as free and open source software. Doing so would require replacing the external libraries—neither of which is open source—with open source alternatives, but because of the modularity of the software, this shouldn't pose a major problem. Especially because timbral synthesis is a new idea, releasing it as free and open source software would play an important role in hastening its adoption and encouraging the collaboration of others who could adapt it to meet their needs, exploring new uses and extending the original ideas as they did so.

6.2.2 Theory

While the theory behind timbral synthesis is more developed than the software, there are still numerous areas for further exploration. For example, while some of the descriptors were easily converted into synthesis equations, the conversion of others was less straightforward. Exploring alternate methods for dealing with these more problematic descriptors might yield different, interesting results. Specifically, the current implementation applies the harmonic spectral deviation constraint to each set of three adjacent harmonics. This means making somewhat arbitrary decisions about dividing up the amount of *hsd* the user requests among those constraints. Finding ways to treat the deviation of the spectrum more holistically would give timbral synthesis more freedom in “optimizing” the spectra it finds.

Similarly, this implementation makes decisions for the user about the way that harmonic spectral variation will be incorporated. During analysis, *hsv* is calculated on a frame-by-frame basis, but is then averaged to give an overall value. For synthesis, the software must decide how to distribute this average—

whether to incorporate the full variation between each set of adjacent frames or whether to stretch a larger variation over many frames. How much space to leave between frames is also an open question that has a profound effect on how *hsv* works. For MPEG-7 analysis, the short-time Fourier transform is generally done on 24 millisecond frames, with a 12 millisecond hop size.[1] However, this custom adopted from timbral analysis need not constrain the size of the frames in timbral synthesis.

In addition to improving the handling of the existing descriptors, researchers might also find it useful to explore other descriptors that could be added to the timbral synthesis process. The addition of a limited number of descriptors might give users more refined control over the timbres they produce without making the interface unmanageable. Further time might also be spent working to normalize the effects of changes across descriptors. While the current setup ensures a good fit between changes made to a single parameter and the corresponding change in the sound produced, significant time was not devoted to scaling the controls so that the sonic difference resulting from a change made to one parameter would be similar to the difference resulting from a similar amount of change in another parameter.

[1] Hop size is the time advance from one frame to the next in a short-time Fourier transform.

Finally, while the timbre description scheme from MPEG-7 is the basis of the current implementation, other timbre models may be better suited to timbral synthesis, especially for particular classes of sounds. As the MPEG-7 standard specifies one timbre space for sustained, harmonic sounds and another for non-sustained, inharmonic sounds, a timbral synthesizer might need different modes for different types of sounds—each mode based on a different underlying set of descriptors. These ideas are only a small sampling of the directions that further research into timbral synthesis could take and further exploration will doubtless lead to new, unimagined directions. But hopefully, these suggestions will provide an initial direction for the next phase of work.

6.3 Final Thoughts

Timbral synthesis is a promising addition to the current palette of sound synthesis methods. Given the prominent role that timbre often plays in electro-acoustic compositions, it seems only natural that composers should be able to manipulate timbre directly. Timbral synthesis is well-suited to this task because of the broad range of timbres that the method can produce, because of the manageable interface it offers users, and because of its basis in our perception of timbre. This combination of range, manageability, and ease of use is certainly not ideal for every situation, but for many applications, timbral synthesis could surpass any of the existing methods.

Compared to previous methods for timbral synthesis, this method provides some clear advantages. Unlike previous implementations that have had to simplify timbre, this new type of timbral synthesis tackles timbre with as complex an approach as our understanding allows, yet it effectively hides that complexity from the user. But unlike other implementations that rely on machine learning to hide timbre's complexity, this new type of timbral synthesis is precise and exacting in the timbres it produces.

Although the theory of timbral synthesis proposed here is still in its infancy, there are already opportunities to put it to work with software not much more complex than the current implementation. Hopefully, near-term applications will spark interest in the exciting possibilities timbral synthesis offers. And with many related avenues to further explore, timbral synthesis should continue to be a fertile ground for research in the years to come.

Bibliography

[1] James W. Beauchamp. *The sound of music : analysis, synthesis, and perception of musical sounds*, chapter Analysis and Synthesis of Musical Instrument Sounds, pages 1–89. Springer, New York, 2007.

[2] Kenneth W. Berger. Some factors in the recognition of timbre. *The Journal of the Acoustical Society of America*, 36(10):1888–1891, October 1964.

[3] Anne Caclin, Stephen McAdams, Bennett K. Smith, and Suzanne Winsberg. Acoustic correlates of timbre space dimensions:a confirmatory study using synthetic tones. *Journal of the Acoustical Society of America*, 118(1):471–482, July 2005.

[4] Shih-Fu Chang, Thomas Sikora, and Atul Puri. Overview of the mpeg-7 standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(6):688–695, June 2001.

[5] COIN-OR. Computational Infrastructure for Operations Research Home Page [online]. June 2007. Available from: `http://www.coin-or.org/`.

[6] Thomas H. Cormen. *Introduction to algorithms*. MIT Press, Cambridge, Mass., 2nd ed edition, 2001.

[7] Dash Optimization. Xpress-mp [online]. June 2007. Available from: `http://www.dashoptimization.com/`.

[8] Dictionary.com. Timbre [online]. May 2007. Available from: `http://dictionary.reference.com/browse/Timbre` [cited May 27, 2007].

[9] Kemal Ebcioğlu. An expert system for harmonizing four-part chorales. *Computer Music Journal*, 12(3):43–51, 1988.

[10] GNU Project. GNU Linear Programming Kit [online]. June 2007. Available from: `http://www.gnu.org/software/glpk/`.

[11] Alex Gounaropoulos and Colin Johnson. Synthesising timbres and timbre-changes from adjectives/adverbs. In F. Rothlauf et al., editor, *Applications of Evolutionary Computing*, volume 3907 of *Lecture Notes in Computer Science*, pages 664–675. Springer-Verlag, April 2006.

[12] John M. Grey. Multidimensional perceptual scaling of musical timbres. *Journal of the Acoustical Society of America*, 61(5):1270–1277, May 1977.

[13] John M. Hajda. *The sound of music : analysis, synthesis, and perception of musical sounds*, chapter The Effect of Dynamic Acoustical Features on Musical Timbre, pages 250–271. Springer, New York, 2007.

[14] John M. Hajda, Roger A. Kendall, Edward C. Carterette, and Michael L. Harshberger. *Perception and cognition of music*, chapter Methodological issues in timbre research, pages 253–306. Psychology Press, Hove, East Sussex, 1997. Available from: `http://www.loc.gov/catdir/enhancements/fy0652/97202027-d.html`.

[15] Lippold Haken, Kelly Fitz, and Paul Christensen. *The sound of music : analysis, synthesis, and perception of musical sounds*, chapter Beyond Traditional Sampling Synthesis: Real-Time Timbre Morphing, pages 122–144. Springer, New York, 2007.

[16] Christophe Hourdin, Gérard Charbonneau, and Tarek Moussa. A sound-synthesis technique based on multidimensional scaling of spectra. *Computer Music Journal*, 21(2):40–55, 1997.

[17] ILOG. ILOG CPLEX [online]. June 2007. Available from: `http://www.ilog.com/products/cplex/`.

[18] Tristan Jehan. Perceptual synthesis engine: An audio-driven timbre generator. Master's thesis, Massachusetts Institute of Technology, Cambridge, Mass., September 2001.

[19] Colin G. Johnson and Alex Gounaropoulos. Timbre interfaces using adjectives and adverbs. In *NIME '06: Proceedings of the 2006 conference on New interfaces for musical expression*, pages 101–102, Paris, France, 2006. IRCAM &#8212; Centre Pompidou.

[20] Carol L Krumhansl. Why is musical timbre so hard to understand? In Sören Nielzén and Olle Olsson, editors, *Structure and Perception of Electroacoustic Sound and Music*, number 60, pages 43–54. Royal Swedish Academy of Music, Excerpta Medica, 1989.

[21] Stephen Lakatos. A common perceptual space for harmonic and percussive timbres. *Perception & Psychophysics*, 62(7):1426–1439, 2000.

[22] Sylvain Le Groux. *Mapping High-Level Sonic Percepts to Sound Generation*. PhD thesis, Universitat Pompeu Fabra, Barcelona, Spain, August 2006.

[23] J.C.R. Licklider. *Handbook of experimental psychology*, chapter Basic Correlates of the Auditory Stimulus. A Wiley publication in psychology. Wiley, New York, 1951.

[24] Max V. Mathews. The digital computers as a musical instrument. *Science*, 142:553–557, November 1963.

[25] Max V Mathews. *The Technology of Computer Music*. M.I.T. Press, Cambridge, Mass., 1969.

[26] Stephen McAdams, James W. Beauchamp, and Suzanna Meneguzzi. Discrimination of musical instrument sounds resynthesized with simplified spectrotemporal parameters. *Journal of the Acoustical Society of America*, 105(2):882–897, February 1999.

[27] Stephen McAdams, Suzanne Winsberg, Sophie Donnadieu, Geert De Soete, and Jochen Krimphoff. Perceptual scaling of synthesized musical timbres: common dimensions, specificities and latent subject classes. *Psychological Research*, 58:177–192, 1995.

[28] Mosek ApS. Mosek ApS optimization software [online]. May 2007. Available from: `http://www.mosek.com/`.

[29] Russell Ovans and Rod Davison. An interactive constraint-based expert assistant for music composition. In *Proceedings: Ninth Canadian Conference on Artificial Intelligence*, pages 76–81, 1992.

[30] François Pachet. Constraints and multimedia. *Practical Applications of Constraint Logic Programming*, pages 3–13, March 1999.

[31] François Pachet, Olivier Delerue, and Peter Hanappe. Musicspace goes audio. In Curtis Roads, editor, *Sound in Space*, Santa Barbara, CA, 2000. CREATE.

[32] François Pachet and Pierre Roy. Musical harmonization with constraints: A survey. *Constraints*, 6(1):7–19, January 2001.

[33] Geoffroy Peeters, Perfecto Herrera, and Xavier Amatriain. Audio CE for instrument description (timbre similarity). Proposal ISO/IEC JTC1/SC29/WG11, International Organisation for Standardisation: Coding of Moving Pictures and Audio, November 1999.

[34] Geoffroy Peeters, Stephen McAdams, and Perfecto Herrera. Instrument sound description in the context of mpeg-7. In *Proceeding of the ICMC2000*. International Computer Music Conference, 2000.

[35] Miller Puckette. pd [online]. May 2007. Available from: `http://crca.ucsd.edu/~msp/Pd_documentation/index.htm` [cited May 21, 2007].

[36] Python Software Foundation. Python [online]. May 2007. Available from: `http://www.python.org/` [cited May 21, 2007].

[37] Schuyler Quackenbush and Adam Lindsay. Overview of mpeg-7 audio. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(6):725–729, June 2001.

[38] Curtis Roads. *The computer music tutorial*. MIT Press, Cambridge, Mass., 1996.

[39] Xavier Serra. *Musical Signal Processing*, chapter Musical sound modeling with sinusoids plus noise, pages 91–122. Studies on New Music Research. Swets & Zeitlinger Publishers, 1997.

[40] Julius O. Smith. Physical modeling synthesis update. *Computer Music Journal*, 20(2):44–56, 1996.

[41] Julius O. Smith. Sinusoidal modulation of sinusoids [online]. December 2005. Available from: `http://ccrma.stanford.edu/~jos/rbeats/rbeats.pdf`.

[42] Julius O. Smith and Xavier Serra. Parshl: An analysis/synthesis program for non-harmonic sounds based on a sinusoidal representation. In *Proceedings of the 1987 International Computer Music Conference*, San Francisco, 1987. Computer Music Association.

[43] SoftSynth. JSyn [online]. May 2007. Available from: `http://www.softsynth.com/jsyn/` [cited May 21, 2007].

[44] Andrew Sorensen and Andrew Brown. jmusic: Music composition in java [online]. May 2007. Available from: `http://jmusic.ci.qut.edu.au/` [cited May 21, 2007].

[45] Bjarne Stroustrup. Stroustrup:C++ [online]. May 2007. Available from: `http://www.research.att.com/~bs/C++.html` [cited May 21, 2007].

[46] Sun Developer Network. Java Sound API [online]. May 2007. Available from: `http://java.sun.com/products/java-media/sound/` [cited May 21, 2007].

[47] David L. Wessel. Timbre space as a musical control structure. *Computer Music Journal*, 3(2):45–52, 1979.