TIMBRAL MIGRATION: STOCHASTIC PROCESSES FOR THE CONTROL OF SMOOTH SPECTRAL TRANSFORMATION

John MacCallum, Andrew Schmeder, David Wessel

Center for New Music and Audio Technologies (CNMAT)
Department of Music
University of California, Berkeley
{john,andy,wessel}@cnmat.berkeley.edu

ABSTRACT

We present an additive synthesis engine that can produce smooth transitions between timbres without *glissandi* or audible transients. With this algorithm, one can explore the musically rich area of hybrid timbres that arise as one timbre "migrates" into another. This method operates in real-time using live or pre-recorded sinusoidal analyses as input. Parameterization of the migration behavior enables interactive control. We present the history of this engine that begins with work by David Wessel at IRCAM in the 1970s. We discuss two implementations: a statistical method based on sampling from a discrete probability mass function derived from a sparse input spectrum, and a stochastic local-update method using parallel Markov Chain Monte Carlo processes, which enables control over spectral flux, spectral smearing and migration rate. We conclude with a summary of notable musical uses in contemporary electroacoustic music.

1. INTRODUCTION

Additive synthesis is inherently problematic—once the oscillators have been turned on, their state may only be changed by first turning them off, or allowing them to *gliss* to their new destination. Unfortunately, not many natural phenomena exhibit the timbral stability of an oscillator bank. Nor do many natural sounds gliss from one point to another, and the ones that to rarely do it with the linear precision of a computer. For these reason, additive synthesis, unless very cleverly done, tends to have a classic sound that one can only describe as "synthesized." The *migrator* algorithm represents an attempt approximate a spectrum, and to move smoothly from one timbre to another without any added transients or *glissandi*. The basic method involves continually updating one oscillator at a time; when a new timbre is input to the system, the oscillators are gradually retuned to frequencies contained in the new timbre. A fortuitous byproduct of this process is that the system is constantly in flux even when the input is stable, which results in a timbre that has a more natural sounding inner life than a typical sound constructed with basic additive synthesis.

2. BACKGROUND

Shortly after Wessel's arrival at IRCAM in July of 1976 he began working with Giuseppe Di Giugno, who had designed and built a digital oscillator bank called the 4A that he brought with him from Italy. This was the first machine in the 4 series that would, over the subsequent years, evolve to the 4B, then 4C, culminating in the 4X. At a sampling rate of 32kHz, the 4A was capable of generating 128 oscillators in real-time. It was interfaced to the Unibus of a PDP 11/40 laboratory computer that ran a real-time OS and Fortran programs that sent parameter values to the the 4A. The frequency, amplitude, and phase of each oscillator were controlled independently and the wave table could handle an arbitrary periodic function.

With the assistance of Di Giugno, Wessel developed software that provided for user control of the migratory behavior of the 4A oscillators that became the basis for Wessel's *Antony*[5]. Completed in October of 1977, the musical aesthetic of *Antony* was influenced by György Ligeti's *Lux Aeterna* and by the French spectral composers especially, Gérard Grisey. In *Antony*, the harmonic flow was to be continuous in character with no discernible changes—no beginnings, no ends, just gradual evolution. The migration metaphor was well suited to these goals. In the cluster of 128 oscillators the frequencies of each were moved one by one to new positions. The control structure used in *Antony* afforded exploration of transitions in real time. The user stayed one step ahead and set up new frequency maps towards which the oscillators would migrate once they had reached a current frequency map. After considerable experimentation the work was performed in four layers of 128 oscillators each. *Antony* was first presented at the 1977 ICMC at the University of California San Diego.

In 2003 Wessel received a request to perform a new live electronic work at the San Francisco Electronic Music Festival. In the intervening years since the 4A, personal com-

puters had become quite powerful; a laptop could produce more than ten times the number of oscillators as the 4A in real time. Musical controllers had evolved as well, primarily due to the development of the MIDI standard. Using a very efficient oscillator bank developed as an external for Max/MSP by Adrian Freed (oscillators~[1]), and a migration inspired control structure, Wessel composed and performed *Singularities*, the first in a new series of works. As in *Antony*, the emphasis was on slowly evolving spectra. Using Don Buchla's touch-sensitive controller, *Thunder*, Wessel assigned different frequency maps to positions along the touch strips for each of his fingers and mapped the pressure applied to each strip to intensity. One hundred oscillators were used for each of the eight fingers and each of the fingers could interpolate along the strip containing four frequency maps. The *migrator* Max/MSP patch in *Singularities* found a number of other applications in pieces by Edmund Campion, John MacCallum, and others described later in this paper.

3. METHOD

The *migrator* algorithm works by updating each component of an oscillator bank (typically containing 100 oscillators) in three stages:

1. Fade out the oscillator to be updated
2. Select a new frequency from the target spectrum
3. Fade in the oscillator

By limiting the number of oscillators being updated (usually one at a time), the rate of change of the output spectrum is controlled so that an audible transient never occurs. When the synthesizer is implemented by a parallel bank of oscillators (e.g., the oscillators~ object in Max/MSP), the *migrator* manages the birth and death of each partial explicitly. If a granular synthesizer using a sinusoidal wavetable is used, *migrator* is required to explicitly renew partials that are not being updated.

3.1. Analysis and Resynthesis

Early in the development of the *migrator*, Wessel realized that it could be a useful tool for resynthesizing a spectrum that had been decomposed using some analysis tool. In Max/MSP, this can be done in real-time by using objects such as fiddle~[3] and iana~[4]. Of course, the analysis can also be done "offline" with tools like SPEAR and AudioSculpt. The entire analysis resynthesis process can be seen in figure 1. The ability to do this timbral transformation "live" allows one to explore complex timbre spaces in an intuitive and improvisational manner.

[1] oscillators~ and all other CNMAT objects can be downloaded from `http://cnmat.berkeley.edu/downloads`

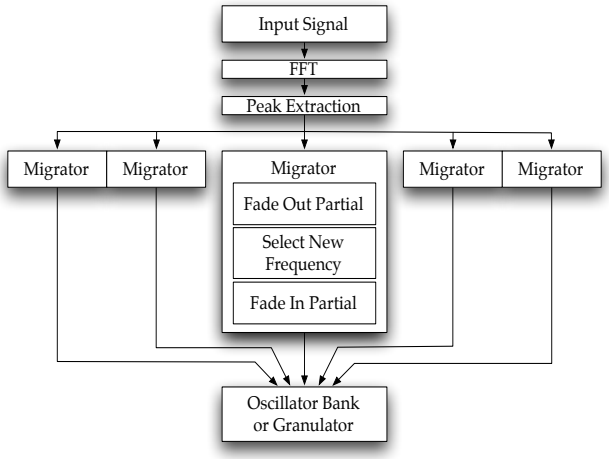

Figure 1. Analysis/resynthesis with the *migrator*

4. IMPLEMENTATION

When an oscillator is updated, the method by which the frequency is chosen can have an effect on the character of the synthesized sound. Here, we present two methods for selecting new frequencies, and we look at the ways in which the different parameters of the *migrator's* update algorithms can be tuned for expressive purposes.

4.1. Spectral Profile as PMF

In the first method, we treat the input spectrum as a Probability Mass Function (PMF), convert it to a Cumulative Distribution Function (CDF) and draw samples from it. In the original Max/MSP implementation, this was done using the table object which has, as a built-in feature, the ability to draw a sample from the data set contained within it. After the sample has been drawn, "Gaussian blur" can be added by selecting a frequency from a Gaussian distribution centered around the new frequency with a user-controlled standard deviation. When fixed at a small value, the standard deviation causes the spectrum to blur resulting in a more natural sounding timbre. When controlled by the performer, this parameter can be an expressive way of adding noisiness to the spectrum.

This method works particularly well for spectra that have been pruned of all but the most salient partials. Because all oscillators are given the same amplitude, partials with very low amplitudes in the target spectrum can be given disproportionate weight in the synthesized spectrum unless a substantial number of oscillators is used.

Another problem can arise when a partial that exists in a region of low spectral energy (i.e. a partial that has a low probability of occurrence) is chosen. When this happens, especially if it is in the upper register above the rest of the spectrum, one's attention is often drawn to the entrance of

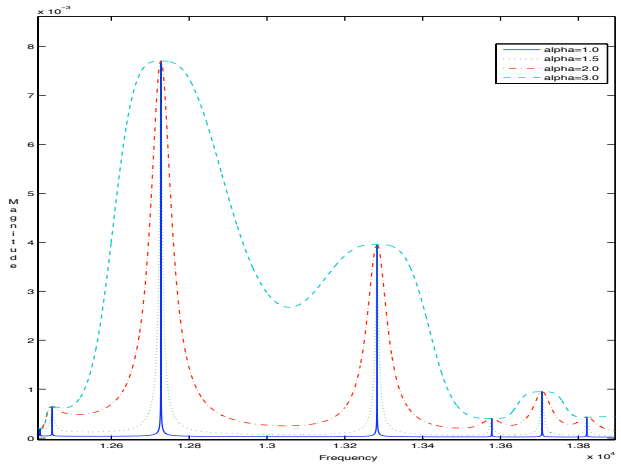

Figure 2. The width of the peaks can be adjusted by altering the interpolation algorithm.

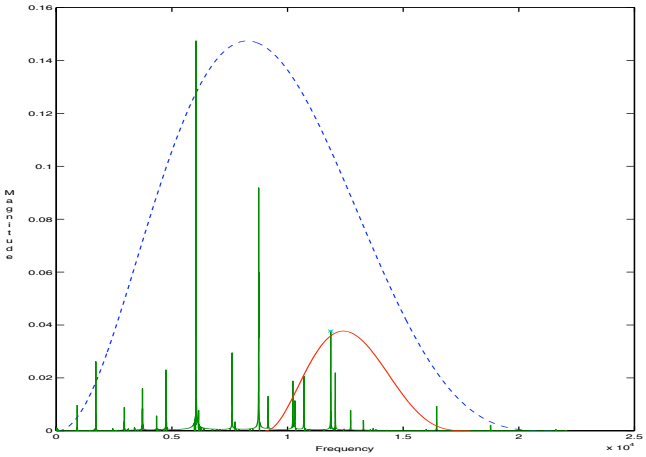

Figure 3. The proposal distribution created by scaling the beta distribution that was fitted to the input spectrum.

that partial. Our second implementation below addresses this.

4.2. Interpolation

The algorithm presented in section 4.1 uses no interpolation between points of the input spectrum—if the "Gaussian blur" is set to 0, the output will be composed only of frequencies present in the input. In the method presented in the following section, we use a simple interpolator[2] covering all points of the input spectrum and two additional points on either side that bound the function:

$$S(x;\{(x_i,y_i)\}) = \frac{\sum_{i=1}^{N} w_i y_i/|x-x_i|^{\alpha}}{\sum_{i=1}^{N} w_i/|x-x_i|^{\alpha}} \tag{1}$$

where w_i is a vector of weights

$$w_i = \begin{cases} \frac{\sum_{0}^{N-1} a_i}{w0} & \text{for } i = 0, N+1 \\ 1 & \text{for } i = 1 \ldots N \end{cases} \tag{2}$$

and $w0$ is the average of the interpolation function which should be some value smaller than the lowest amplitude of the input spectrum and is calculated automatically when input is received. The function is bounded on both sides at

$$x_0 = \begin{cases} m_1 f_{min} & \text{for } m_1 f_{min} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

$$x_{N+1} = \begin{cases} m_2 f_{max} & \text{for } m_2 f_{max} < f_{nyquist} \\ f_{nyquist} & \text{otherwise} \end{cases} \tag{4}$$

α is set by the user and affects the interpolation function as seen in figure 2. We can also replace α with a vector α_i to have local control over the *noisiness* or *temperature* in different regions of the spectrum. Typical values for m_1 and m_2 are 0.9 and 1.1 respectively.

4.3. Markov Chain Monte Carlo

The second method implements a version of the Metropolis-Hastings algorithm[1]. We start by taking a random step away from the current frequency of the oscillator being updated. The step size is a random value sampled from a *proposal density* function $Q(x_t|x_{t+1})$. For this, we use the beta distribution fitted to the envelope of the input spectrum (figure 3). We then scale the distribution to a width defined by the user and center its mean around the current frequency of the target oscillator. A sample is then drawn from the proposal density function and accepted with probability

$$r < \frac{P(x_{t+1})Q(x_t|x_{t+1})}{P(x_t)Q(x_{t+1}|x_t)} \tag{5}$$

where r is a uniform random variate drawn from $U(0,1)$, $P(x_t)$ is the amplitude of the current frequency and $P(x_{t+1})$ is the amplitude of the proposed frequency. If the proposal is rejected, we re-try until a proposal is accepted, or the maximum number of tries is exhausted at which point we can either draw a sample from the input spectrum's CDF as described in section 4.1, or leave the oscillator's state unchanged.

This method only allows partials to move to frequencies that are nearby, thereby mediating problem of a stray partial appearing out of nowhere. However, unless the width of the proposal density function is carefully controlled, oscillators can become "stuck" in a narrow band around a peak in the input spectrum (see the bottom plot in figure 4). This is not necessarily a problem as it may indicate that the algorithm has settled into a stable representation of the input. However, we can encourage the partials to change their state by adjusting the α parameter in our interpolation function (see

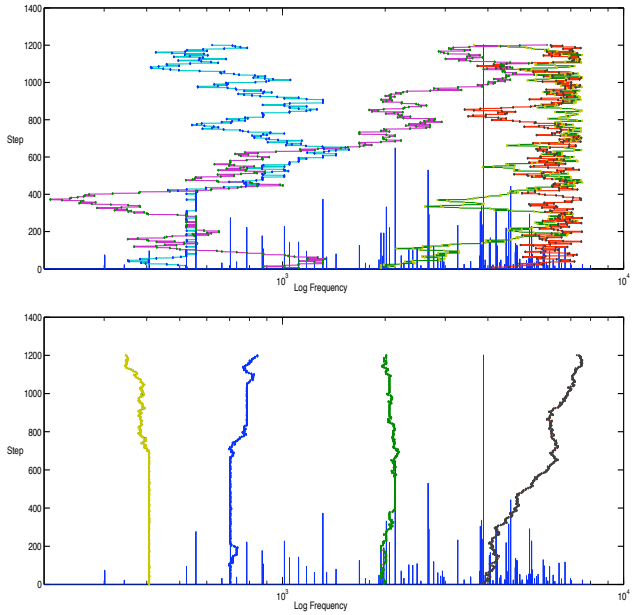

Figure 4. Evolution of four partials over 300 steps. $\alpha = 0.5$ and the proposal density width is $0.5 f_c$ (top) and $0.05 f_c$ (bottom)

eq. 1 and figure 2). Raising α can also smear the spectrum in a way that is similar to increasing the standard deviation in the algorithm presented in 4.1.

5. NOTABLE USES

5.1. Timbre Space

With the Buchla touch-sensitive controller, *Thunder*, Wessel assigned different spectral PMFs to locations along the one dimensional pressure sensitive strips with pressure mapped to intensity. Using the recently developed SLABS controller[2] we lay out eight or more spectral PMFs in a two dimensional space for each touch pad, providing for a natural interpolation among spectral profiles. Additional strategies for control of the structure of the spectral PMFs are under study, most notably procedures that broaden or sharpen the spectral peaks.

5.2. Prolongation of Orchestral Texture

In 2005 Kent Nagano asked Wessel and Ali Momeni to provide electronic interludes to a large scale operatic production arranged for the occasion of the 10th anniversary of Toro Takemitsu's death. The work, entitled *My Way of Life*, was a sequence of orchestral and chamber works composed by Takemitsu. Momeni used the migration technique to provide a number of transitions between pieces in real-time. A single microphone was placed before the orchestra and spectral PMFs were estimated and applied to the migration procedure providing for an extension of the orchestral texture and an evolving transition between the pieces.[3]

In his work for string orchestra and electronics *Hold that Thought*[4], Edmund Campion makes efficient use of the *migrator* to both prolong and reinforce the texture. An important contribution Campion made to the *migrator* was to update all of the oscillators at once so that an abrupt change in harmony could occur.

MacCallum's work for full orchestra and electronics *Alloy* uses the *migrator* as a real-time prolongational device similar to Campion's work cited above, and provides a microtonal haze for certain passages. The piece uses complex microtonal harmonies derived from the analysis of Tibetan singing bowls. Strong partials from the bowls were adjusted to equal temperament and used in the orchestral parts, while the *migrator* provided the characteristic spectral smearing.

6. CONCLUSION

The *migrator* brings its own character to the music created with it. In that sense, it is more like an instrument than a tool for faithfully reconstructing a sound from an analysis. Additionally, due to certain limitations inherent in the algorithm (the update rate, for example), the tool itself suggests ways in which it should be used.

7. ACKNOWLEDGEMENTS

We wish to thank Adrian Freed for his invaluable advice, as well as those people who, through their use of the *migrator*, have advanced its development, especially Edmund Campion, Ali Momeni, and Aaron Einbond.

8. REFERENCES

[1] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, p. 97, 1970.

[2] M. Lombardi, "Interpolation and smoothing," *Astronomy & Astrophysics*, vol. 395, p. 733, 2002.

[3] M. Puckette and T. Apel, "Real-time audio analysis tools for Pd and MSP," in *Proceedings of the International Computer Music Conference*, 1998.

[4] T. Todoroff, E. Daubresse, and J. Fineberg, "Iana~ (a real-time environment[...])," in *Proceedings of the International Computer Music Conference*, 1995.

[5] D. Wessel, "Antony," Wergo WER 2030-2032, 1980.

[2] http://cnmat.berkeley.edu/user/david_wessel/blog

[3] http://alimomeni.net/takemitsu-my-way-life

[4] www.edmundcampion.com/project_holdthatthought/index.html