An Overview of Criteria for Evaluating Synthesis and Processing Techniques

DAVID A. JAFFE
295 Pudue Ave.
Kensington, CA 94708
david@jaffe.com

Abstract

A wide variety of synthesis and processing techniques have been invented. The question may arise as to which is best. It turns out there is no simple answer. To help clarify the issues, we present a list of ten criteria for evaluating synthesis and processing techniques and give examples of well-known techniques that succeed especially well or fail especially poorly in each area.

This paper is forthcoming (in expanded form) in Computer Music Journal.

Introduction

A *synthesis technique* is a strategy for generating sound samples, based on some control information called *parameters*. Parameters generally change at a rate that is significantly slower than the audio sampling rate. Also, while sound samples are usually produced at a constant rate, parameter-setting messages are often sporadic and irregular. For example, in the familiar Yamaha DX7 synthesizer (Chowning and Bristow, 1986), the synthesis technique is FM (frequency modulation) synthesis, which produces a steady stream of audio samples, while MIDI pitch bend is a parameter that changes only when the performer moves the modwheel or foot pedal. For a historical summary of synthesis techniques, see, for example, (Smith 1991.)

A *processing technique* is similar to a synthesis technique, except that it includes an additional input of one or more audio sample streams at the audio sampling rate. An example of a common processing technique is reverberation. Here, the parameters are controls such as the delay before the first reflections, the balance between the reverberant and dry signals and the decay rate of the echoes.

A wide variety of synthesis and processing techniques have been invented. The question may arise as to which technique is best. It turns out there is no simple answer--the best technique depends on the priorities of the user and the problem to be solved.

To help clarify the issues, we present a list of ten criteria for evaluating synthesis and processing techniques and give examples of well-known techniques that succeed especially well or fail especially poorly in each area. The first four criteria are concerned with the usability of the parameters: Are they intuitive? Does a change in a parameter have a perceptible effect? Do they map to physical attributes of musical instruments or other physical sound-producing mechanisms? Are they "well-behaved" or wildly non-linear? Other criteria deal with the sounds produced: Do they retain their identity in the context of variation? Can all classes of sounds be produced? Are there techniques for deriving parameters for real-world models? The remaining criteria are concerned with efficiency and implementation. How efficient is the technique? Does it have an undesirable unavoidable latency? How sparse is its parameter control stream?

The intention here is not to survey all known techniques, but to outline the criteria that make a given technique suitable or unsuitable to a specific purpose. A technique that is weak in a certain area can often be made stronger by combining it with another technique to produce a hybrid.

How Intuitive are the Parameters?

This topic is concerned with whether the parameters map in an intuitive manner to musical attributes such as musical dynamics and articulation, or whether they are mere mathematical variables with very little correlation to real-world perceptual or musical experience.

In the old analog synthesizer technique of low-pass filtering a complex waveform, the bandwidth of the low-pass filter affects both the brightness and the amplitude of the sound. This control is very much like a musical "dynamics" parameter. In a real-world instrument, loud notes tend to have more significant higher

partials than soft notes, due to physical non-linearities becoming more prominent as the instrument is played louder. In the synthesis simulation, low values of the filter bandwidth correspond to a dynamic of pianissimo while high values correspond to fortissimo. This correlation is especially effective if the filter bandwidth is adjusted on a note-by-note basis so that a requested fundamental amplitude is obtained uniformly regardless of frequency, thus decoupling distance (largely a function of amplitude) from dynamics (primarily a function of brightness) (Jaffe and Smith 1983.)

In contrast, parameters of non-linear synthesis methods, such as complex FM (Schottstaedt 1977), can be quite non-intuitive. Although an extremely useful technique, it has the property that changing the FM index of a cascade modulator slightly can cause a drastic and difficult to predict change in tone quality, enough of a change to turn a drum into a woodblock. This situation may be merely annoying to a composer, who can take the time to find the proper value for his or her application, but it can drive a performer crazy who is trying to control such a parameter from an instrument in front of a live audience.

How Perceptible are Changes in Parameters?

Changing a parameter by a significant amount should have an obvious audible effect. We call such parameters *strong* or *powerful*, in contrast to *weak* parameters whose effect is barely audible. Typically, the more parameters a technique has, the weaker each parameter is. Parameters that are too weak leave the composer to wander in the dark, setting parameters to arbitrary values, since it's not possible to hear any difference. On the other hand, parameters that are too strong can also cause problems. If a tiny change causes a huge effect, a performer may have difficulty controlling the technique.

Again, the filter bandwidth of a low-pass filtered complex waveform is a good example of a reasonably strong parameter. A single parameter controls a clear effect of brightness. In contrast, additive synthesis, in which a tone is produced by a weighted sum of sinusoidal components, is an example of a technique with weak parameters. If you change the amplitude envelope of a less-important harmonic, the change can be completely inaudible.

A technique with weak parameters can be transformed into a related technique with more powerful parameters, often using the same underlying algorithm. For example, an additive synthesis brightness parameter can be defined that behaves similar to the low-pass filter brightness parameter. Such a parameter can be defined as a *meta-parameter* that simply scales a set of the existing additive synthesis parameters. We would define the user-supplied partial amplitudes and frequencies as the arrays *Amps[i]* and *Freqs[i]*, respectively, where '*i*' is the partial index. In normal additive synthesis, the *Amps* values are used directly. With the new *brightness* parameter, the actual value applied could be defined as:

```
ActualAmps[i] = Amps[i] * MAX(brightness,0.001) ^
                          log2(MAX(Freqs[i],100.0)/50.0)
```

Here, a *brightness* of 1.0 represents no modification and a *brightness* of .001 represents maximum low-pass filtering. (*log2()* is the *log* function, base 2 and *MAX()* returns the maximum of its arguments.) This technique can be viewed as a modified form of FFT filtering, where the additive synthesis is used in a manner similar to an inverse Fourier transform. (In standard FFT filtering, a sound is analyzed via the Fourier transform, the analysis data is multiplied by the frequency response of an FIR filter, and the result is inverse-transformed.)

A particularly rich example of the use of meta-parameters in conjunction with additive synthesis allows the user to describe the sound as a path through a multi-dimensional space of timbres. Each dimension of a line segment function describes the scaling of a particular timbre. These values are added and the results are used to scale the additive synthesis harmonics, producing complex interpolations (McNabb, 1981.)

A technique with even weaker parameters than additive synthesis is digital sampling, where each sample value can be viewed as a parameter. In fact, sampling can be considered the "identity synthesis technique," where the parameters are the samples themselves. Certainly, with the exception of introducing or removing a click, adjusting a single parameter (sample) does not produce a very noticeable effect. In practice, most samplers actually use a hybrid technique that includes filters, amplitude envelopes, and other more reasonably-parameterized processing modules.

How Physical are the Parameters?

We call a *physical parameter* one that not only mimics the behavior of a real-world instrument, but actually controls a synthetic instrument in the same manner as its real-world counterpart. Physical parameters are ideal for creating complex behaviors that arise during unstable moments in a tone's evolution, such as during instrument attacks and transitions between notes. The beauty of techniques with physical parameters is that they don't require a formal analytical model or data base describing the behavior of the instrument under all circumstances. For example, neither the player/composer nor the implementer of the technique itself need understand exactly what happens in the output waveform during an attack or transition--the physical nature of the parameter causes the correct result.

One of the strongest arguments for physical modeling synthesis is that the parameters are exactly those used by a human instrumentalist. A waveguide violin model (Smith 1993) (one type of physical model) has bow pressure and bow velocity parameters; a clarinet model has a mouth breath pressure parameter (Cook 1988). Of course, playing a real-world violin or clarinet well is no trivial matter and requires years of training and practice. This same training may be needed, though in the virtual domain, to play the physical models (Chafe 1985.) For musicians who are not violinists or clarinetists, non-physical controls that are still intuitive may be more appropriate.

An example of a non-physical parameter is the amplitude of an additive synthesis overtone. Changing the amplitude of the third harmonic is not a very relevant control when attempting to synthesize a violin. While meta-parameters could in principal be defined to map pseudo-physical parameters to any underlying technique, we often don't have the acoustical knowledge necessary to adequately define this mapping.

How Well-Behaved are the Parameters?

Ideally, a change in a parameter produces a proportional change in the sound. If a small parametric change produces a wild unpredictable sonic result, the parameter is called "poorly-behaved" and can be a nightmare for composers and performers alike.

It is commonly thought that linear techniques, such as additive and subtractive synthesis, always have well-behaved parameters. However, such systems are linear only when the parameters change at slow rates relative to the sampling rate. When parameters change rapidly, energy is injected into the system, allowing the possibility of unpredictable or unexpected results. For example, consider a simple amplitude scaling that is allowed to be changed at an audio rate. This situation is equivalent to amplitude modulation (AM) synthesis. If the modulation is sinusoidal, two sidebands are produced for each component in the original signal, a non-linear effect. These sidebands are at frequencies corresponding to the frequency of each component plus or minus the modulating frequency. If the modulation is non-sinusoidal, the effect is even more complex. Especially dangerous are filter structures that can become unstable during transitions from one set of coefficients to another.

An example of a technique that exhibits excellent behavior during transitions is the waveguide-based vocal synthesis model of Perry Cook (Cook, 1990), which models the vocal tract as a series of waveguide filters, each representing a cross-section of the tract. Changing vowels has a physical interpretation--one or several sections shrink or increase in diameter. Interpolating between vowels corresponds directly to real-world spatial interpolation and all intermediate values have an intuitive, physical interpretation and are well-behaved.

Chaotic behavior can arise in non-linear feedback techniques (McIntyre and Woodhouse 1983), often used in physical models of wind instruments. In fact, this "extreme sensitivity to initial conditions" is often given as the very definition of a chaotic system. Still, with sufficient care, such a model can be made to operate in regions of its state space where it behaves predictably and effectively.

How Robust is the Sound's Identity?

Here we are concerned with how well the sound retains its identity in the context of variation. This is the author's personal favorite criterion. To do an effective musical instrument simulation--one that is more than a mere snapshot of a moment--it is essential to be able to synthesize "expression," which we define as "great variety in the context of a particular perceived source." For example, a violinist can make many changes in his or her sound, but the sound is still clearly a violin--it doesn't suddenly turn into a trumpet. Many synthesis techniques can approach a particular note on a particular instrument very closely, but fail miserably in making the family of perceptually-related sounds that is required for a true expressive instrument simulation.

Physical models do exceptionally well in this area. For example, the extended Karplus-Strong plucked string (Karplus and Strong 1983, Jaffe and Smith 1983), is a partially-physical model based on waveguides. It has the property that it sounds like a plucked or struck string no matter what you to do to it. You can vary such parameters as pick position, string flexibility, string thickness, dynamics and decay characteristics to a great degree, providing a rich expressive vocabulary while never leaving the realm of string-like identity.

On the other end of the spectrum, pure sampling is a notorious offender in this area. A single trombone sample, on first hearing, is impressive because it sounds exactly like a trombone. But there is a strong tendency for a composer to assume it is an actual trombone, rather than a single snap-shot of a trombone. He or she may

be tempted to compose a variety of articulations, durations, dynamics and timbres only to discover that his or her supposed trombone is a two-dimensional cardboard cut-out of the real thing that falls over as soon as it is pushed lightly in any direction.

The power and flexibility of sampling can be greatly enhanced by using filters for dynamics and linear interpolation for timbral variety. For example, you can record a quiet trombone note and a loud trombone note and, if care is taken to match the pitch and phase exactly, do linear interpolation between the two to get a range of dynamics. If a huge amount of memory is available, another solution is to use a large library of samples that represent the trombone in all of its guises, including various attacks, dynamic contours and vibratos. Nevertheless, despite its seductive realism, sampling presents both a challenge and an opportunity to the composer attempting to use it in an expressive manner.

How Efficient is the Algorithm?

Efficiency is an extremely important criterion. In a real-time context, it determines the number of voices that are possible on a given piece of hardware. In non-real time contexts, it determines the amount of time a composer must wait before hearing the results of a computation. Given a fixed amount of time to complete a piece, an extremely long turn around time translates into fewer iterations, resulting in a less refined result.

Determining the efficiency of an algorithm is more complicated than it might first appear. It is not merely a matter of comparing processing benchmarks. Numerous aspects of a technique and its implementation come into play. We divide these into three categories: memory requirements, processing details and control stream attributes.

Memory

It is a well-known axiom of computer programming that you can often trade off memory against processing power. For example, when doing wavetable synthesis (Mathews 1969), in which a single period of a waveform is stored in memory, you can store a huge table and use a non-interpolating ("drop-sample") oscillator. Alternatively, you can store a smaller table and use a more expensive oscillator that interpolates between samples in the table. For that matter, using a wavetable oscillator at all is a memory optimization. A wavetable oscillator multiplied by an amplitude envelope can be replaced by a pre-computed version of the entire resultant waveform; the result is less real-time computation at a much greater memory cost. . On the other end of the spectrum, the waveform can be computed analytically using any of a number of techniques. For example, if the wavetable is a sine wave, three alternatives to wavetable- based oscillators are marginally stable two-pole filters, evaluation of a complex phasor (Gordon and Smith 1985) and waveguides (Smith and Cook 1992.)

Some techniques have a memory requirement that changes with the parameter values. The Karplus-Strong plucked string, as well as several other waveguide-based modeling technique requires more memory for lower pitches than for higher pitches.

Processing

The issue of processing power itself is quite complex and depends to some degree on the details of the processor architecture. Some traditionally expensive techniques, such as finite element modeling (numerical integration of the difference equations that describe masses and springs), are well-suited to parallel architectures such as array processors. As another example, an algorithm with an active code size that fits within the cache of a RISC chip will run many times faster than one that overflows the cache (Freed, Rodet and Depalle 1993.) Techniques with minimum changes in program flow are well-supported by DSPs and other heavily pipe lined architectures. Of course, special-purpose hardware can increase enormously the efficiency of a technique.

Just as memory usage can be dependent on parameter values, some techniques have a processing requirement that changes with the parameter values. The time-domain implementation of the Chant FOF (Rodet 1984), which does voice synthesis (as well as synthesis of other resonant systems) by adding up overlapping vocal tract impulse responses, becomes more expensive as the frequency rises--there are more pitch periods per second, and thus more additions and table-lookups per output sample.

Numerical characteristics also come into play. Can a technique be implemented in fixed point or does it require a large dynamic range? An algorithm that requires floating point has a higher cost associated with it on most systems. For example, some filtering structures are easier to implement in a floating point environment where overflow of intermediate values need not be a concern.

A related but different issue is how many bits of precision are required? It is important not to confuse

precision and dynamic range. When an algorithm uses floating point, it is trading off precision for dynamic range, assuming a constant word size. For example, 32-bit unsigned floating point typically has 24 bits of precision, while unsigned 32-bit integers have a full 32 bits of precision. Thus, on a 32-bit machine, integers actually have more precision than floating point numbers.

Some algorithms change their behavior depending on the word precision. This is especially likely with non-linear recursive systems, such as non-linear steady state oscillations in physical models, where small deviations in either calculation accuracy or initial conditions can completely alter the path and final state of such systems. In fact, it may make the difference between oscillating at all or not oscillating.

Another example of word precision affecting algorithm behavior is in regard to "limit cycles," annoying oscillations that arise at the end of an exponential decay and continue forever. If rounding is used, even the convergent rounding used in the DSP56001, which guarantees there will be no bias accumulated by the rounding, it is likely that limit cycles will arise. Since limit cycles tend to be confined to the lower bits, the more word precision, the less objectionable the limit cycles. One way around this is to implement truncation toward zero instead of rounding. But this tends to make exponential decays faster. For example, an exponential decay degenerates to linear in such systems; an exponential ratio constant of 0.99999999 just causes a 1 to be subtracted from the magnitude each sample, yielding a linear decay (Cook 1993.)

Control stream

The third efficiency consideration is the heaviness of the required control stream. Even an inexpensive algorithm, if it has a very dense control stream, may be difficult to implement in real time. Many systems use two processors, one dedicated to sound computation and the other handling control data. The difficulty comes from the cost of getting the control data to the sound processor. Even if there is only one processor, the control stream may consist of many mega-bytes of data, more than can fit in RAM, so disk space and disk bandwidth become scarce resources. Compression may be of use here.

When evaluating an algorithm's control stream requirements, it is important to differentiate between techniques that require dense bursts of parameter update messages as opposed to those with relatively steady streams of messages. A well-spaced fairly-dense control stream may be easily managed by some systems that fail when the same volume of control data is clumped in large bursts. For example, if an amplitude envelope changes with each note, it may actually be more efficient to feed the break-points to the processor one point at a time, rather than sending the entire envelope in a burst at the beginning of each note. This leads to a closely related issue...

How Sparse is the Control Stream?

In designing, implementing or using a synthesis technique, it is important to keep in mind where the actual work is being done. Is it the synthesis method or the control data that is actually doing the work? This is an often-overlooked distinction. Some synthesis techniques require such a bulk of control data that it actually exceeds the number of samples synthesized. This state of affairs is not necessarily bad, because the control data may be in a more useful parameterized form, as in the case of phase vocoder-based additive synthesis (Dolson 1986, Portnoff 1977.) Nevertheless, it is important to keep this criterion in mind. In fact, many synthesis techniques can be made nearly equivalent when fed enough data. A single sine wave that is frequency-modulated very quickly with a complex signal can produce intelligible speech. We can easily imagine absurd synthesis techniques where all the information is in the control stream. An example is the "modulated constant technique," defined as the number 1.0 multiplied by an "appropriate" control stream!

It is hard to come up with an example that ideally optimizes this criterion because the amount of control information that is necessary (a negative characteristic) is often directly proportional to the amount of control that is possible (a positive characteristic.) The best technique here is one where everything is pre-determined, a situation that is, by definition, non-interactive. Once interactivity is removed from the picture, very little information is required. For example, there is an old joke where a sailor says "I know two songs. One is Yankee Doodle. And the other isn't." Thus, with one bit you can represent both Beethoven's Ninth Symphony and the entire Ring Cycle of Wagner. Similarly, most MIDI synthesizers are designed for a minimum control stream because of the limitations of MIDI itself. This is not necessarily an advantage, since it limits the expressive bandwidth between the player and the synthesis. Still, especially in real-time implementations, control stream density must be monitored carefully or it can become a major bottleneck. Control stream optimizations, such as suppressing duplicate parameter-setting messages, lazy garbage collection and shared resource management can significantly reduce the density (Jaffe 1990.) Meta-parameters, such as the additive brightness parameter described above, can be defined. These are then expanded into lower

level parameters on the processor itself, thus reducing dramatically the control bandwidth. This is actually one form of control stream compression.

Perhaps the best way to look at the criterion of control stream density is to hold the amount of possible control constant, and proceed to minimize the control stream density for that amount of possible control. Physical models excel in this regard, since they tend to have a small number of powerful parameters.

Many control stream optimizations are specific to particular techniques. Pure phase vocoder-based synthesis (oscillator bank resynthesis), as well as short-time inverse Fourier synthesis (inverse FFT resynthesis) (Rodet and Depalle 1992), are extremely data-intensive techniques. Yet, there are ways this cost can be reduced. First, if an appropriate window is chosen, the window may skip ahead in the overlap-add process. This produces a dramatic decrease in the amount of control data. Further decreases come from the field of psycho-acoustics. In frequency domain techniques such as MPEG audio encoding (MPEG 1991), partials may be omitted if they are of low enough amplitude that they are entirely masked by neighboring partials. Similar techniques make possible the DCC and mini-disk consumer audio formats. However, such data reduction should be avoided if post-processing is planned, since a partial that was originally masked may be exposed during the post-processing phase. Finally, an efficient alternative to the phase vocoder has recently been developed (Serra and Smith 1990), that separates the analyzed sound into deterministic and stochastic components. It then models the deterministic portion using traditional phase vocoder methods, while modeling the stochastic portion with filtered random noise. For a wide class of real-world sounds, this results in a dramatic reduction not only in the control stream density, but also in the synthesis computation itself.

What Classes of Sounds can be Represented?

Is there a class of sounds that is impossible to produce with a given synthesis technique? While some techniques can produce any sound, given appropriate control data, others are less general. For example, inharmonic sounds are impossible with pure wavetable synthesis. However, once you add frequency or amplitude variation, which is almost always used to some degree, inharmonic sounds become possible. If you have an amplitude or frequency envelope with a rapid attack, you are actually producing inharmonic partials momentarily during the attack portion of the sound. For some synthesis techniques, the lack of flexibility can actually be viewed as an asset. A particular physical model, say a brass instrument, may synthesize only brass sounds. But this is exactly the property of "identity in the context of diversity" that was lauded as a great advantage in the fifth criterion above ("How Robust is the Sound's Identity?").

What is the Smallest Possible Latency?

Some techniques have an inherent and unavoidable latency that may cause problems in an interactive situation, where a sound must be computed and played immediately in response to an asynchronous event. A typical example is a MIDI keyboard triggering a sound that is synthesized on the fly. The situation also arises in the processing of live sound. Clearly, the sound cannot be computed in advance because it has not happened yet. If the technique has an inherent latency, there will be no way of avoiding a latency between the onset of the incoming sound and the resultant output sound.

Techniques with minimum latency are those that can consume and produce samples one at a time, with the first output sample emerging as soon as the triggering event has occurred or the input sound has begun. Many of the techniques mentioned in this paper, including FM, waveguide-based physical modeling, additive synthesis and wavetable synthesis have this property. Sound processing techniques such as reverberation (as it is ordinarily implemented), flanging, and equalization similarly follow this rule.

Other techniques may a require significant latencies. FFT-based sound processing methods with large block sizes have a latency on the order of the FFT size. Other techniques are simply impossible in real time, regardless of the processor speed. An example of such a technique is non-causal filtering. You can't make something laugh before you tickle it. Another example is playing a sound backwards. You have to wait for the sound to finish before you can even start playing it. The latency is the length of the sound itself.

Do Analysis Tools Exist?

When computer and electronic music was first invented, both the technical literature and the popular press were full of claims that now, finally, we could "make any sound." However, as composers started exploring the space of parameter values, they discovered that most of that landscape produced sounds of undistinguished character. For example, although all excerpts of white noise have different sample values, they sounds very much the same. It is not enough to know in theory that any sound can be produced. You need tools to derive the proper parameter values from a specification of a desired result. This process can be manual

or automated, as well as intuitive or analytical.

The question we ask of our synthesis technique, then, is whether well-understood analysis techniques exist for deriving parameters from a real-world description? While all synthesis is certainly not directed at imitating real-world sounds, this class of sounds is of great interest to many composers. Coming up with the right parameters to induce a technique to make the sound that in theory it is supposed to be able to make can be tricky, to say the least.

It is generally easiest to come up with an analysis technique for a synthesis technique that is capable of exactly reproducing any sound. The phase vocoder analysis/synthesis system is an example of such a technique. It derives additive synthesis parameters from the samples of a recorded waveform, then reproduces exactly those samples. We call this the "identity property."

Another popular analysis technique is linear prediction (LPC) (Markel and Grey 1976), which designs a series of recursive filters for subsequent resynthesis. Like the phase vocoder, it starts from a recorded waveform, but it can produce only an approximation of some input sounds. As an example, nasal vocal sounds are not well-represented by linear prediction. And while there are other analysis techniques such as Prony's method (Prony 1795) that attempt to take this into account, all are approximations--none has the identity property of the phase vocoder.

Yet, the identity property is only part of the picture. Unless mere data compression is the goal, composers want not only to derive parameters from real-world examples, but also to modify these parameters to create related but different sounds. One of the most basic transformations is to change the pitch of a sound. In this area, LPC has a significant advantage over the phase vocoder. While transposing a phase vocoded note also distorts its spectrum, LPC allows the pitch of the sound to be easily modified with no change to the spectral envelope. To do the same thing to phase vocoded data requires first fitting a spectral envelope to the data, then computing a new spectrum under that envelope--a much more computationally-intensive process. The reason LPC is superior in this case is the fact that the fundamental frequency is decoupled from the spectrum, while with the phase vocoder, the two are merged.

Samples of a waveform represent only one possible input to an analysis system. One can imagine techniques for deriving synthesis parameters and algorithms from other sorts of descriptions, such as a spatial representations of physical sound-producing mechanisms.

Conclusions

It's safe to assume that all existing synthesis/processing techniques have some benefit, otherwise they would have died out from lack of use. The plethora of hybrids now commonly in use shows that various techniques can be combined to maximize strengths and minimize weaknesses from several areas. This hybridization typically shows up after a technique has been around for some time and its characteristics have been extensively explored. FM is a case in point. Even the DX7 featured multi-carrier FM, which is actually a hybrid between FM and additive synthesis. This model turned out to be particularly useful for synthesis of formant structure in the human voice (Chowning 1989.) The NeXT Music Kit (Smith, Jaffe and Boynton 1989) includes a hybrid wavetable/FM synthesis in which the carrier and modulators can have arbitrary waveforms. Thus the ability to match a given periodic spectrum, which is provided by wavetable synthesis, is combined with the dynamic quality of FM. A similar capability is provided by Waveshaping synthesis (Le Brun 1979.)

In the final analysis, the appropriateness of a given technique depends on the task at hand. To quote the cellist who fruitlessly tries to teach loser Fielding Melish in the Woody Allen film "Take the Money and Run":

"He has no conception of the instrument--he blows into it!"

The moral is to avoid wasting time blowing into a cello--get a bassoon. On the other hand, blowing into a cello can occasionally have its own peculiar rewards and lead to discovery of unforeseen sound treasures.

References

(Chafe, 1989) Chris Chafe. "Simulating Performance on a Bowed Instrument." *Current Directions in Computer Music*, M. Mathews, ed., MIT Press, Cambridge, MA.

(Chowning, 1989) John Chowning. "Frequency Modulation Synthesis of the Singing Voice." *Some Current Directions in Computer Music Research.*, Cambridge MA, MIT Press, pp. 57-63.

(Cook, 1993) Perry Cook. Personal communications.

(Cook, 1988) Perry Cook. "Implementation of Single Reed Instruments with Arbitrary Bore Shapes Using Digital Waveguide Filters." *CCRMA Dept. of Music Report* No. STAN-M-50.
(Cook, 1990) Perry Cook. "Identification of Control Parameters in an Articulatory Vocal Tract Model, with Applications to the Synthesis of Singing." Ph. D. Dissertation, Elec. Eng. Dept., Stanford University.
(Chowning and Bristow, 1986) John Chowning and David Bristow. *FM Theory and Applications*. YAMAHA Music Foundation, Tokyo.
(Dolson, 1986) Mark Dolson. "The Phase Vocoder: A Tutorial." *Computer Music Journal* 10(4):14-27.
(Freed, Rodet and Depalle, 1993) Adrian Freed, X. Rodet and P. Depalle. 1993. "Synthesis and control of hundreds of sinusoidal partials on a desktop computer without custom hardware." *Proceedings of the International Conference on Signal Processing Applications and Technology.*
(Gordon, 1985) John Gordon and J. O.Smith. 1985. "A Sine Generation Algorithm for VLSI Applications." *Proceedings of the 1985 International Computer Music Conference.*
(Jaffe, 1990) David Jaffe. "Efficient Dynamic Resource Management on Multiple DSPs, as Implemented in the NeXT Music Kit." *Proceedings of the 1990 International Computer Music Conference*, Glasgow, Scotland, Computer Music Association, pp. 188-190.
(Jaffe and Smith, 1983) David Jaffe and Julius Smith. "Extensions of the Karplus-Strong Plucked String Algorithm." *Computer Music Journal* 7(2):56-69. Reprinted in *The Music Machine,* Roads, C., ed., MIT Press, 1989.
(Karplus and Strong, 1983) Kevin Karplus and Alex Strong. "Digital Synthesis of Plucked String and Drum Timbres." *Computer Music Journal* 7(2):43-55. Reprinted in *The Music Machine,* Roads, C., ed., MIT Press, 1989.
(Le Brun, 1979) Marc LeBrun. "Digital Waveshaping Synthesis". *Journal of the Audio Engineering Society* 18(2):250-266.
(Markel and Gray, 1976) J. D. Markel and A. H. Gray. *Linear Prediction of Speech*, Springer-Verlag, Berlin Heidelberg.
(Mathews, 1969) Max Mathews. *The Technology of Computer Music*. MIT Press, Cambridge MA.
(McIntyre and Woodhouse. 1983) McIntyre and Woodhouse. "On the Oscillations of Musical Instruments," Journal of the Acoustical Society of America 63(3):816-8253.
(McNabb, 1981) Michael Mcnabb. "Dreamsong: the Composition." *Computer Music Journal* 5(4):36-53.
(ISO-IEC, 1991) International Standards Organization ISO-IEC. 1991. "Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to About 1.5 Mbits/s." *Motion Picture Experts Group Audio Specification* ISO-IEC JTC1/SC29/WG11, Documents 3-11171, 3-11172.
(Portnoff, 1977) M.R. Portnoff. "Implementation of the Digital Phase Vocoder Using the Fast Fourier Transform." *IEEE Trans. on Acoustics, Speech, and Signal Processing* ASSP-25 pp. 235-238.
(Prony, 1795) R. Prony. "Essai experimental et Analytique sur les lois de la dilatabilit'e des fluides 'elastiques et sur celles de la force expansive de la vapeur de l'eau et de la vapeur de l'alcool, 'a diff'erentes temperatures." *J. Ecole Polytech*, Paris, 1:24-76.
(Rodet, 1984) Xavier Rodet. "Time Domain Formant Wave Function Synthesis." *Computer Music Journal* 8(3):9-14.
(Rodet and Depalle, 1992) Xavier Rodet and P. Depalle. "Spectral Envelopes and Inverse FFT Synthesis," *Proceedings of the Audio Engineering Society*, San Francisco.
(Schottstaedt, 1977) William Schottstaedt. "The Simulation of Natural Instrument Tones using Frequency Modulation with a Complex Modulating Wave." *Computer Music Journal* 1(4):46-50.
(Serra and Smith, 1990) Xavier Serra and Julius O. Smith. "Spectral Modeling Synthesis: A Sound Analysis/Synthesis System Based on a Deterministic plus Stochastic Decomposition," *Computer Music Journal* 14(4):12-24.
(Smith, 1993) Julius O. Smith "Physical Modeling using Digital Waveguides." *Computer Music Journal* 16(4):74-87.
(Smith, 1991) Julis O. Smith. "Viewpoints on the History of Digital Synthesis." *Proceedings of the International Computer Music Conference*, Montreal, pp. 1-10.
(Smith, Jaffe and Boynton, 1989) Julius Smith, David Jaffe and Lee Boynton. "Music System Architecture on the NeXT Computer." *Proceedings of the Audio Engineering Society Conference*, Los Angeles.
(Smith and Cook, 1993) Julius O. Smith, Perry. R. Cook, "The Second-Order Waveguide Oscillator." P*roceedings of the International Computer Music Conference*, San Jose, California, pp. 150-153.

Acknowledgements

We have attempted to present a balanced view of the synthesis techniques discussed. Nevertheless, our bias toward physical modeling techniques can not be hidden. Also, we apologize to those researchers and techniques not explicitly cited in this article--the choice was biased by the techniques with which the author has the most familiarity, having implemented them himself or used them in musical compositions. Thanks to Julius Smith for consultations and suggestions.