A sound, general-purpose sound-synthesizer must be capable of supporting a variety of potentially
complex synthesis algorithms in real time. This
requirement is normally met by high-speed,
high-cost, programmable hardware. The technique
described here is capable of emulating many of the
popular synthesis algorithms, yet only a
microcomputer and some relatively simple dedicated
hardware are required for effective real-time
operation.

Sets of single cycles of audio waveforms are
stored in a digital memory. A waveform of fixed
harmonic content can be produced by cycling
through one of these. Under the control of a host
microcomputer, however, the hardware can switch
between waveforms instantaneously. This enables
signals having rapidly-changing, dynamic spectra to
be synthesized. The need for interpolation in
cross-fading hardware (to prevent glitching in the
countdown) is avoided, since it is arranged
that the amplitude and phase of the waveform are
concerned over the loops. The only discontinuity
arises from the change in spectral content, and it
is small that provided this is not gross (which is
easy to arrange in practice), the perceptual
effect is of a smooth change in timbre.

The waveform cycles stored in memory can be
regarded as a multi-dimensional array of musical
tones. Figure 1 illustrates this concept for the
dimensionality normally, when spectral
evolution depends on function of time, frequency and amplitude (as example). If we
regard these as forming a basis set of vectors,
time the evolution of the spectra is determined by
a vector (call it the t-vector) that can have an
arbitrary functional dependence on each of the
given vectors. Higher dimensionality of the
'time-home' gives greater flexibility, at the
expense of increased memory requirements.
Fortunately, the number of dimensions required to
produce naturally interesting sounds is usually
quite small: a one-dimensional space is
sufficient to emulate additive, subharmonic or
wave-shaping synthesis, for example. Performance
gestures may be used to exercise
control in two ways. Firstly, a base-vector may
be assigned to a gestural input device, the reading
from which determines the component of the
t-vector. In the one-dimensional case, it would
be possible to emulate the sweeping of an analogue
filter by a joystick, for example. The second
method of performance control is in the
redetermination of the multi-dimensional
cycles. The host microcomputer, guided by the user, can
synthesize new waveforms and transfer these to the
appropriate hardware memory locations (points in
time-space). The time delay before hearing the
effect of a gestural act can be made acceptably
short, since only one cycle of the waveform needs
to be generated.

My implementation of the technique employs
pitching and time-various sound-synthesizing
techniques to provide sixteen channels at a sample
rate of 2400. Sound examples will be presented
that demonstrate the ability of the system to
evaluate and generalize upon some popular
synthesis algorithms.

1. SPECTRAL WAVEFORM SYNTHESIS 1.1 MUSICAL
EXAMPLES

1.1 Introduction

In 1932, work started on the development of a
uni-digital, real-time, polyphonic sound synthesis
system (described elsewhere in these proceedings;
Greenough, Raver and Norris [1932]). A
general-purpose, microprogrammable architecture
for the synthesis system was ruled out on cost
grounds. Therefore, a synthesis algorithm had to
be chosen that would be easy to control, not
require expensive hardware, but yet be
capable of generating a wide range of sounds.
Many synthesis algorithms were considered, but all were found to suffer from defects in some respects. For example, it is easy to implement and control, but the range of sounds available with the algorithm was not felt to be sufficiently diverse. Additive synthesis, on the other hand, is capable of synthesizing my possible sound and is quite easy to control in an intuitive way. However, complex soundware is required to implement the algorithm, and supplying parameters would place a heavy strain on the bandwidth of the link with the host computer.

Sennel and Smith (1980) have described a tone-reduction scheme for additive synthesis that overcomes some of these difficulties. Rather than specifying the amplitude envelopes of partials directly, an index pointer into a file of spectra is used. This index selects the spectrum that is to be used at any instant, while interpolating between spectra used to "smooth the resulting amplitude function". While their technique largely overcomes the parameter supply problems, the authors concede that speed requirements for the hardware are "aweome" as a larger number of sinusoids must be generated and their velocities by the amplitude functions. Spinal synthesis, described by Nads (1978), was the next algorithm to be considered, though in this case, the technique did not seem to present control as difficult as a large number of parameters are required, and complicated interpolation hardware is needed. These difficulties are largely overcome in the technique we describe. The technique differs substantially from spinal synthesis in many respects, and demands a new name: I have called it Switched Waveform Synthesis, or SW.  

1. Switched waveform synthesis 
In switched waveform synthesis, a sequence of N samples is stored in a digital memory. These samples form one of the segments of an additive waveform. By addressing the memory with a modulo-N phase accumulator, a sequence of samples is generated at the output from the memory, which may be processed further or fed to a Digital-to-Analog Converter. The resulting waveform is one of fixed spectrum, with components whose amplitudes and phases are in fixed integer ratios (often forming a harmonic series). The frequency of the waveform is determined by a phase increment supplied to the phase accumulator. While the frequency of the waveform is very easy to implement, the sounds produced generally are not as flexible as desired. In Switched Waveform Synthesis, the hardware can effect an instantaneous transition between waveforms and another, while maintaining dynamically varying spectra to be produced and makes the algorithm very flexible. Several waveforms of N samples are stored in a digital memory, and the lower-order address bits are supplied by a modulo-N phase accumulator (as in fixed waveform synthesis). The higher-order address bits are programmed directly by a host microcomputer. By modifying these higher-order address bits, the microcomputer can effect an instantaneous transition between one waveform (call it the source) and another (the destination), thus changing the output.

In general, this abrupt change of waveform can give rise to one of four perceptually distinct effects: "glimitching" (modulate clicks, sounding similar to scratches on a gramophone record), abrupt change of timbre, or a gradual evolution of timbre. Glimitching is generally held to be undesirable in an audio context, while abrupt changes of timbre may be of occasional musical use, gradual timbral evolutions are most often what we require. Therefore it is necessary to understand what gives rise to these effects, and how glitching may be avoided.

Early experiments were conducted using a microcomputer system fitted with analog-to-digital and digital-to-analog converter peripherals. Sounds from both natural and synthetic sources were sampled and stored in the microcomputer memory. New sounds were then synthesized by rolling through a portion of the memory (corresponding to a cycle of the stored waveform) and switching to a different portion of the memory (corresponding to a different waveform) periodically. Glimitching was found to occur over the transitions when:

a) Phase was not continuous,
b) The period of the waveform was different,
c) The amplitude of the waveforms differed greatly,
d) Some changes in the spectrum were involved.

Of these, the first two (a and b) were found to be the most important contributory factors to glitching; to avoid glitching, it was typically necessary to keep both phase and frequency constant to better than 14 over the switch, while peak-to-peak amplitude changes of around 20 dB could be tolerated. The glitching was relatively independent of the quantization limits, and the values used in the experiments (255, 32767, 34000) were all quite satisfactory. Also, in the longer term, the effects of phase distortion were important, and the time for which the source waveform was stored before the switch (this was measured entirely by the pulse-amplitude of spectral components, or what are called "notches", in the spectrum) was very important, though the psychoacoustic data available on temporal effects with complex waveforms is rather limited.

To devise some statistical envelopes of fit test to predict the existence of glitches over a switch would therefore require some detailed psychoacoustical experiments to be performed, which would be time consuming (though probably very worthwhile). Any such test would require information on the spectral components (in order to perform multi-order fits) and, in any case, information would not be known in advance. The goodness of fit test would therefore have to either run in real time (requiring complex hardware) or assume worst-case values for masking (predicting glitches where none occur in practice). An automatic "fit-glimitch" would therefore be of limited use, and was considered to be beyond the scope of the present work. Instead, it was decided to use an interactive "trial-and-error" approach.
Glitches can be avoided by the use of intermediate waveforms. These are produced by mixing together the source and destination waveforms in some proportion, and carrying out the switch from source to destination in two or more stages, going via the intermediate waveforms. The change in operation of the waveform at the point of each stage is reduced, and, in practice, it was found that rather few intermediate waveforms were generally required to avoid glitches. The precise manner in which this situation can be controlled by the user, who is unmodified by the need to avoid glitches and the size of the available waveform memory. In alternative use of intermediate waveforms would be to use a "dynamico interpolation technique between source and destination waveforms (as in granular synthesis), but this would require complicated interpolation hardware and increased the required bandwidth from the waveform memory. Revealing the nature of the description of 2, 3, we can now how glitches can be avoided when using the technique. Changes in period of great frequency occur over switches, as all waveforms are stored in 8 samples and the period is determined only by the phase accumulator, which is unaffected by the switch. Phase changes can be avoided by ensuring that each frequency component has the same phase in each waveform; a convenient and sufficient condition for this is to generate the waveforms that are to be stored using sine waves only. Large amplitude changes over a period can be avoided simply by normalising the waveforms in the memory. This has the added advantage that amplitude can be controlled accurately (by subsequent envelope and gain controllers). By storing additional waveforms that are mixtures of the desired source and destination waveform, the final output of the waveform is obtained, and it is found that by using a sufficient number which usually quite small, the intermediate waveforms, a gradual evolution between any two waveforms can be achieved.

1. Waveform Generation

There are many ways in which the individual waveform components may be altered, as this can be done by the host microcomputer, which in principle can generate a waveform according to an algorithm. One possibility is to use an additive synthesis technique in which, the harmonic amplitudes of the waveforms are set to fade. The microcomputer generates a sequence of samples and feeds this into the waveform memory. The process takes place in the microcomputer, giving close to real-time control. The process can be speeded up by an order of magnitude or more by the addition of a dedicated microcomputer system. Filters, non-linear distortion elements etc. may also be implemented by the microcomputer, giving it the ability to create waveforms. The result can be effective in an analysis/synthesis role, provided the signals to be analysed can be well described by a set of harmonics having time-varying amplitudes. In 5,000, a sound is sampled and the analysis described by Hoppen, Grey and Brown (1979) is applied, giving the variation of spectral-amplitude amplitudes with time.

At points along the time axis (typically separated by a few milliseconds) the amplitudes of the waveforms are determined and used to generate one cycle of a waveform. The resulting set of waveforms can be summed sequentially so as to approximate the original spectral evolution. This is illustrated in figure 2, which shows the desired evolution of a harmonics, and the SMS approximation to it. The resulting sounds are rarely indistinguishable from the original because of the need to use a strictly harmonic set of partials. Nevertheless, it is often possible to recognise which instrument is being synthesised, and the sound is usually quite firm. It is easy to modify the spectral evolution of the sounds by modifying the waveform opies in a different order, and changing the rate at which they are selected. This completes the description of the basic SMS technique. The waveforms in 5,000 SMS may be used at a synthesis technique in its own right as well as being used as a technique. The waveforms in 5,000 SMS may be used at a synthesis technique in its own right as well as being used as a technique.

Figure 2. Illustrating a) the desired evolution of a partial with time, and b) the SMS approximation to it.

2. TIME SPACE AND THE CONTROL OF SMS

2.1 Controlling SMS

If essentially evolving, dynamically are to be synthesised with SMS, it is necessary to switch between waveforms every few milliseconds. The data for controlling the switching could come from a pre-prepared 'waveform', through large quantities of waveforms could be too large to be specified and stored. This control technique was used in my first implementation of the SMS technique on a 320-based microcomputer system. While this technique is feasible, supplying the necessary data proves tedious and non-intuitive, and it was soon realised that a real-time method of control was required.

2.2 Update

In the context of Senn and Senn (1980), described earlier, a one-dimensional array of
spectra (or, equivalently, of timbres) is addressed by an index pointer. This makes real-time control easy, but makes the technique rather inflexible, since the spectra depend only on a single control parameter. Intuitively, this means that a major limitation, though it is one that is shared by many synthesis techniques that are suitable for real-time synthesis in waveshaping synthesis. For example, the spectra depend only on the amplitude of the waveform, while in some implementations of additive and subtractive synthesis, the spectra depend only upon one or frequency, respectively. The technical details are still to be addressed by several orthogonal indices. We can regard these indices as lying along a set of basic vectors that define a multi-dimensional fluidic space. It seems reasonable to call this the timbre space, but since this space has previously been used in a psychoacoustic context (e.g., Dennis (1979), Schindler (1991), I shall normally call it the space. The spectra that is produced at any instant is determined by taking the vector sum of the indices that generate a new vector (call it the vector) that locates a point in space. Each point in space has a corresponding spectrum that is represented (in 3D) by a wavefront cycle. The evolution of this index thus constrains the path of the vector, and hence the evolution of the space. The indices can be assigned (by the user) to any available parameters—pre-defined functions, stochastic function generator, and functions obtained from external input devices—so all relevant candidates, for example. It is true that my sound cannot be synthesized simply by specifying a parameter at a particular time, but the use of a multi-dimensional space gives a far better representation of the perceived space than the simple index used for an individual sound. 2.3 Simulating other algorithms The concept of t-space is illustrated for the case of three dimensions in figure 1. The figure shows a view of a space (given by amplitude (in frequency space) located at points within the 3D space. The three space vectors are time, frequency and amplitude, while the indices are used to get at the general. This is a particularly interesting case, as we find that in the t-space in case of describing the important algorithms, waveshaping, additive and subtractive synthesis. In waveshaping, for example, the spectra depend only upon the index of the space. To describe waveshaping, we therefore define a one-dimensional t-space with the index depending on a linear function of amplitude. In particular, the t-space, we place spectra that correspond to the spectra that would be produced by a waveshaping synthesizer with the corresponding value of the amplitude function. The only difference between the sound produced by waveshaping and that produced by the same function is that due to the coarse quantization of the 3D space. This can be made perceptually insignificant by defining a sufficient number of steps within the 3D space, so in conjunction with the t-space.

formalism, is therefore capable of producing all the sounds that are available with waveshaping, while similar arguments show that additive and subtractive synthesis may also be extended (it is difficult to produce "harmonic" spectra with 3D, however, this is not a fundamental limitation of the t-space formalism. T-space could be used in conjunction with additive synthesis to produce any kind of spectrum. With a one-dimensional array of indices addressed by several orthogonal indices, we can regard these indices as lying along a set of basic vectors that define a multi-dimensional fluidic space. It seems reasonable to call this the timbre space, but since this space has previously been used in a psychoacoustic context (e.g., Dennis (1979), Schindler (1991), I shall normally call it the space. The spectra that is produced at any instant is determined by taking the vector sum of the indices that generate a new vector (call it the vector) that locates a point in space. Each point in space has a corresponding spectrum that is represented (in 3D) by a wavefront cycle. The evolution of this index thus constrains the path of the vector, and hence the evolution of the space. The indices can be assigned (by the user) to any available parameters—pre-defined functions, stochastic function generator, and functions obtained from external input devices—so all relevant candidates, for example. It is true that my sound cannot be synthesized simply by specifying a parameter at a particular time, but the use of a multi-dimensional space gives a far better representation of the perceived space than the simple index used for an individual sound. 2.3 Simulating other algorithms The concept of t-space is illustrated for the case of three dimensions in figure 1. The figure shows a view of a space (given by amplitude (in frequency space) located at points within the 3D space. The three space vectors are time, frequency and amplitude, while the indices are used to get at the general. This is a particularly interesting case, as we find that in the t-space in case of describing the important algorithms, waveshaping, additive and subtractive synthesis. In waveshaping, for example, the spectra depend only upon the index of the space. To describe waveshaping, we therefore define a one-dimensional t-space with the index depending on a linear function of amplitude. In particular, the t-space, we place spectra that correspond to the spectra that would be produced by a waveshaping synthesizer with the corresponding value of the amplitude function. The only difference between the sound produced by waveshaping and that produced by the same function is that due to the coarse quantization of the 3D space. This can be made perceptually insignificant by defining a sufficient number of steps within the 3D space, so in conjunction with the t-space.
assigned to as many gestural input sources as we want (or have, or are able to control). Furniture owned from these input sources determine components of the t-vector, and alone these are orthogonal, a given gestural act can be made to have a distinct effect on the timbre. Having learnt the effect of each gesture individually, it is easy for the performer to combine gestures in an intuitive way to produce the required sounds.

3.3. Alternatives for Future Work

I hope that I have shown that the t-space formalism is a very powerful one that yields up many new possibilities for sound-synthesis. Because of the shortcomings of input devices, however, my implementation of the technique has been a rather restricted one. In the hope that others may be encouraged to try the work further, I offer a few suggestions for advancements.

An n-dimensional t-space that requires n points to be defined along each dimension requires a total of M^n (M raised to the power N) points to be defined and stored. In SAW, M is larger than it would be if interpolation techniques were to be used (so as to avoid glitches), and the total number of points required soon becomes very large. Some forms of interpolation would clearly be desirable, but the waveform interpolation technique described by Achorre (1985) would probably be impractical for large N, as the number of multi-dimensional waveforms required is M^n. Fortunately, the interpolation technique used by Neat and Heath (1981) does not suffer from this defect: they use a digital filter to "smooth out" abrupt changes in spectral amplitudes supplied to an additive synthesizer. While this is not exactly what is required, the effect is similar. The overall effect may not always be what was desired: if the SAW filter is too low and the points occur frequently, the filter may not respond quickly enough and will attempt to change the spectrum. If switches between spectra occur frequently, however, the effect may be of abrupt changes of timbre, rather than the gradual evolution that may have been required. These effects would be eliminated by using a 'noise vector' (i.e., a random vector with components having mean zero and some distribution along each dimension) to the t-vector.

Effects could be enhanced by adding a 'noise vector' (i.e., a random vector with components having mean zero and some distribution along each dimension) to the t-vector. The idea is illustrated in Figure 3 for one case of a two-dimensional t-space. If the noise vector was not present, the sound would always be the closest to the head of the t-vector (in this case, that labeled ST). With noise added, however, the spectrum chosen is that nearest to the sum of the t-vector and the noise vector. This is most likely to be S (in Figure 3), but C and D will also be chosen occasionally, the probabilities depending on the distance from the t-vector to the noise. If the noise calculations are rapid, and mixing use of the averaging effect of the spectral amplitude filter, it should be possible to achieve a gradual evolution between any two spectra. I estimate that using byte of memory for storage of spectral amplitudes, t-space of up to seven or eight dimensions could usefully be implemented using this technique.

4. CONCLUSIONS

Switched A/D Synthesis is new way to implement digitally, as only a phase accumulation and a waveform storage memory are required. Nevertheless, a wide range of interesting sounds can be produced by the algorithm, the t-space formalism is a very general one for describing sounds, and can be used both to extend existing algorithms, and to explore new ones in a fertile and intuitive way. My own implementation has shown both the SAW and t-space techniques to be viable, and I have made some suggestions for future implementations.

References


Figure 3. Illustration of the effect of adding a noise vector to the t-vector. In two dimensions, the m vectors points that have been defined in the space. The contours represent the probability distribution of the noise vectors' positions near the head of the t-vector have the highest probability. Also shown are two particular noise vectors, N and R. In the case of N, point u is selected, whereas for R (which is less probable) point S is selected.

