A Method for Envelope Warping in Digital Audio Synthesis*

ROBERT C. MAHER, *AES Member*

Department of Electrical Engineering, University of Nebraska–Lincoln, Lincoln, NE 68588-0511, USA

The use of stored envelope functions is common to many audio signal synthesis techniques. This table lookup method is susceptible to audible defects in the synthesized output if care is not taken in the design of the synthesis algorithm, particularly when the rate at which values from the envelope table are accessed is to vary with time. A useful strategy is presented for altering the effective lookup rate for a stored envelope function in order to minimize audible discontinuities between envelope segments with differing lookup rates. The approach creates a smooth warping of the stored envelope function by forcing the envelope lookup index (the envelope phase) to vary according to a cubic polynomial function with matched transitions from envelope segment to segment.

0 INTRODUCTION

Digital audio synthesis methods must strike a balance between computation speed and memory requirements. For example, repetitive operations such as generating a fixed waveform are typically accomplished using an incremented address pointer into a stored array of numerical values (a wave table) which corresponds to a sampled version of the desired function. This table lookup approach is based on the assumption that less time is required to obtain a value from the wave table and update the table pointer than to calculate the desired function explicitly. The constant-amplitude waveform generated by the wave table is then multiplied by an amplitude envelope function to produce the desired time-varying amplitude characteristic for the signal.

In the case of envelopes, that is, functions used to modulate signals, it is often desirable to produce a continuum of envelopes differing in certain parameters, such as rate of attack or maximum value. One such approach uses a template envelope perhaps obtained by analyzing a recorded acoustic musical instrument tone as a point of departure: instead of attempting to record (and store) every possible combination of performance parameters (such as loudness, duration, musical pitch, intensity), a small collection of representative envelopes is determined and modified in such a way as to simulate the desired behavior of the instrument for a particular musical context specified by the composer or determined by the performer. The template approach can also be used in additive synthesis algorithms where the amplitude history of each partial is determined. For example, a target musical note 1 s in duration can be analyzed to determine its amplitude envelopes for use as a template. In order to simulate a note with a duration of 2 s, the template information can be warped in order to increase its duration. *Since the time warping is applied only to the amplitude envelope and the signal from the controlled wave table is not altered, the result is a time-scale change without changing the musical pitch of the signal.* This useful property of the envelope concept should be kept in mind throughout the discussion that follows.

However, simply time stretching the entire envelope in a linear fashion is unlikely to produce the desired effect in general. Instead, one may choose to subdivide the 1-s template envelope into several sections and warp them separately, as depicted in Fig. 1. In this example, the envelope is arbitrarily divided by hand into three segments: an *attack* portion, a relatively constant *sustain* portion, and a short *release* portion. By choosing to stretch the sustain portion by a greater factor than the attack and release portions, the perceptual

* Manuscript received 1991 January 24; revised 1991 June 27.

quality of the synthesized signal may be improved in comparison to the simple linear stretch approach.

It should be noted that the use of envelope functions in digital synthesis is not, of course, limited to amplitude modulation tasks. Depending on the needs of the synthesis algorithm, an envelope function may be used to control a wide range of signal parameters such as frequency, filter bandwidth, or FM modulation index.

In practice the division of the template envelopes into segments has been performed manually. The process is often accompanied by a data-reduction step in which the shape of the envelope is approximated by a simple set of straight-line segments, as shown in Fig. 2. The abrupt discontinuities in slope associated with the straight-line segments do not cause a discontinuity in the signal itself, but must lead mathematically to a corresponding *spectral* discontinuity. For envelopes which change slowly compared to the sample rate (the usual situation for amplitude envelopes) this distortion has been found to be inaudible in most situations [1].

However, time warping a sampled envelope function (or its line-segment approximation) may cause audible discontinuities, particularly in cases where the envelope is followed by additional stages of processing or filtering. This may lead to audible changes at each envelope break point. Furthermore, the use of line-segment approximations or some other parametric description of the envelope shape is impractical in situations where the details of the envelope shape are important. Even in the more common situation in which no interruption is heard at the segment transition, the result may be an unnatural or noticeably altered timbral characteristic when compared to the original unwarped envelope. Thus we desire a means to use the efficient table lookup operation for warped envelope generation while avoiding audible defects and timbral alteration.

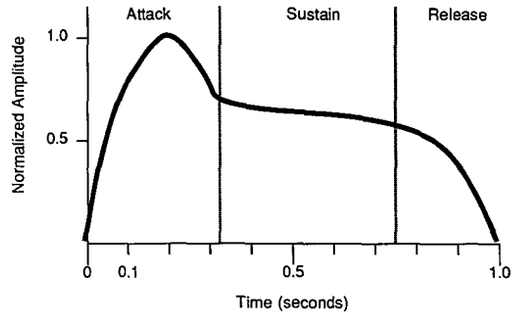


Fig. 1. Simple envelope function manually divided into three segments: attack, sustain, and release.

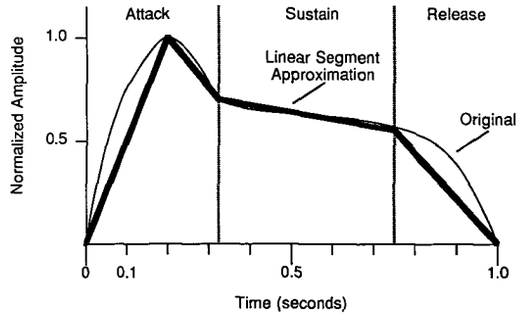


Fig. 2. Simple envelope function of Fig. 1 represented by several linear segments. This representation usually requires less storage and is often aurally indistinguishable from the original shape.

The research reported here treats the audible effects of envelope time warping by considering a nonlinear lookup function. Rather than abruptly changing the time-warping factor from one envelope segment to the next, the nonlinear lookup function gradually changes the warping at segment boundaries while retaining the efficiency of the prestored template envelope function. Calculation of the nonlinear lookup function coefficients can be performed at the beginning of the envelope and the lookup function itself can be implemented efficiently using the well-known method of forward differences. This procedure is particularly suited to the warping of stored envelope shapes of such complexity that simple smoothing of the envelope itself is infeasible.

It must be reemphasized that envelopes with discontinuous slopes are used frequently and with complete success in audio synthesis. While the deleterious effects caused by time warping envelope segments are usually negligible in practice, situations do arise in which abrupt changes in the envelope warping are unacceptable, as shown in Fig. 3. The unwarped envelope [Fig. 3(a)] was generated by a table lookup followed by a filter stage. The envelope with individually warped segments [Fig. 3(b)] suffers from an accentuation of the slope discontinuities by the filter stage. The improved result incorporating the nonlinear lookup function to avoid the abrupt warping changes is shown in Fig. 3(c). Additional examples involving timbral issues are considered in Sec. 5.

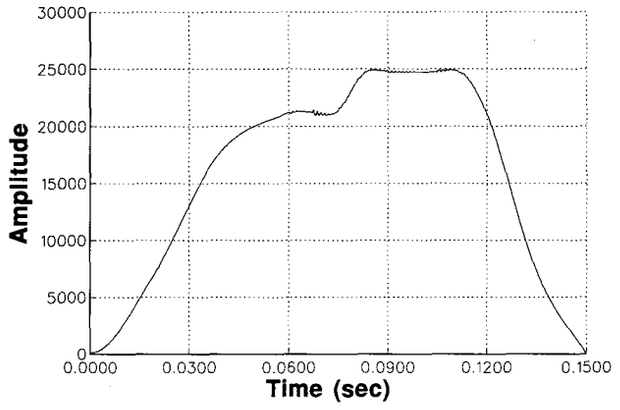
Sec. 1 of this paper begins with an elementary description of the strategies commonly used for envelope table lookup in computer music systems. Next, the problem of envelope lookup with time warping is exposed, including some empirical results. The third section contains a description of the new envelope warping scheme using cubic functions for the lookup index function, followed by a discussion of the method and its implementation.

1 BASIC ENVELOPE TABLE LOOKUP

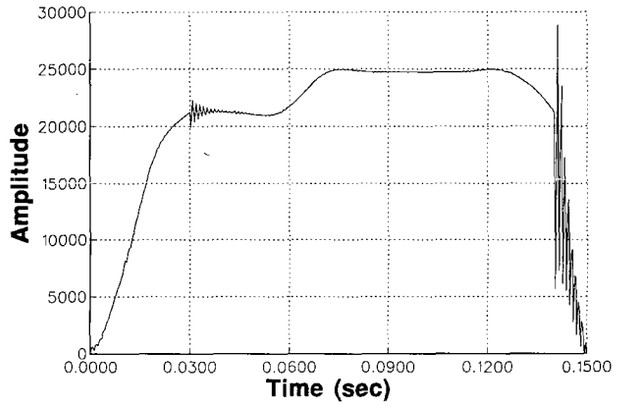
Digital synthesis algorithms are commonly described as a collection of unit generators [2], [3]. Each generator performs a specific subfunction within the particular synthesis model. Typical unit generators include audio signal sources (audio frequency oscillators), control sources (envelope generators, subaudio frequency oscillators), signal processors (modulators, filters, reverberators), and signal summers (audio mixers). Fig. 4 shows a block diagram of a simple signal source constructed from several unit generators.

The implementation of a synthesis algorithm using unit generators may be done entirely in hardware, entirely in software, or in some combination. The im-

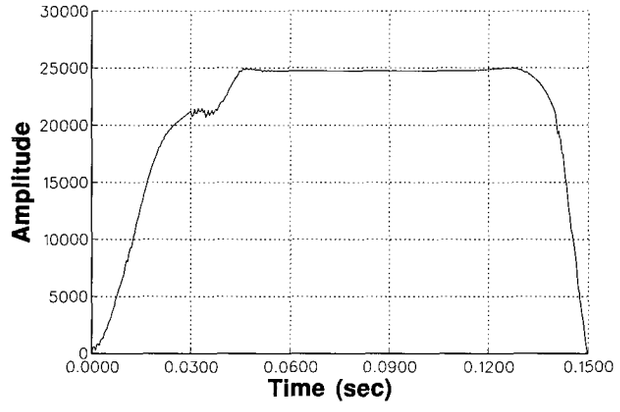
plementation may take advantage of particular features of the algorithm in order to reduce complexity or computation time. For example, the RAMP generator in Fig. 4 could be implemented as a counter: the *next* value depends solely on the *current* value and the time interval between observations. While most unit generators are assumed to produce or receive new data at the required signal sample rate, some functions may be computed less frequently, perhaps only at the beginning of a note or at some divisor of the signal sample rate. In situations where non-real-time operation is acceptable, the various unit generator operations may actually occur consecutively rather than in parallel. This type of delayed-performance computer music is found in the MUSIC *X* computer languages running on general-purpose mainframe and mini computers [4], although recent developments in signal processing hardware are beginning to allow for real-time software synthesis using microcomputer-based systems [5], [6].



(a)



(b)



(c)

Fig. 3. Example of envelope function with time warping. (a) Original (unwarped) envelope generated by table lookup followed by filtering. (b) Same envelope with independent warping of its segments using a conventional linear lookup function. (Note fluctuations caused by abrupt change in lookup rate.) (c) Same envelope with independent warping of its segments using proposed nonlinear lookup function method. [Note absence of fluctuations in comparison to (b).]

As mentioned in the Introduction, one of the basic operations of many typical unit generators is table lookup. The unit generator "plays" the stored function by sequentially reading the stored values from the table. A repetitive waveform can be generated by reading the table over and over, that is, once the end of the function is reached, the unit generator begins reading from the beginning of the table, becoming a fixed-waveform oscillator. For envelopes, on the other hand, the unit generator is typically designed simply to stop reading the lookup table once the end of the table has been reached, perhaps sustaining the last envelope value.

Since the amount of memory available for lookup tables is usually limited in practice, the size and number of tables must be chosen with some planning. Moreover, because the lookup tables contain sampled functions of time, issues relating to the time density of samples and the use of interpolation must also be considered. For example, we may wish to represent a slowly varying envelope function using a lookup table, say, 512 samples long, representing a total duration of 2 s. With an actual audio sample rate probably 20 kHz (for computer music) or greater per channel, the unit generator must either repeat each envelope value from the table some 40 or more times (zero-order hold), or perform some form of interpolation. Fig. 5 depicts a simple linear interpolation scheme.

A single template envelope can be used for different musical contexts by changing the amplitude scaling or time scaling of the envelope lookup process. Careful

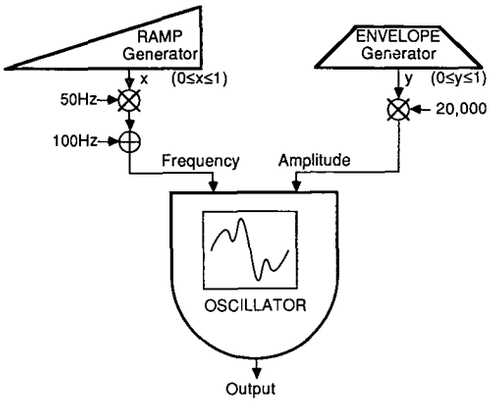


Fig. 4. Block diagram of simple synthesis algorithm: a fixed-waveform oscillator is amplitude modulated by an envelope generator and frequency modulated by a linear ramp function. Maximum output amplitude is 20 000 units; frequency varies from 100 to 150 Hz.

manual segmentation of the stored envelope table by a skilled synthesist can result in a flexible representation for muscial modification by individually time scaling each segment. However, many envelope shapes of interest are not easily divided into independent sections. Fig. 6 shows an attempt to compress a template envelope from its original 3-s duration to 2 s by scaling the sustain portion of the lookup table. The slope discontinuities present in the compressed envelope may cause noticeable timbral artifacts in the output signal—the sound is not only shortened, its tone quality may be altered as well.

2 ENVELOPE TABLE LOOKUP WITH TIME WARPING

Because the goal of time warping a template envelope is to simulate the expected behavior of the modeled instrument when playing notes of longer or shorter duration, the presence of audible timbral distortion could be unacceptable. Attempts to simply "smooth out" the discontinuities of the warped envelope can be successful, but this technique usually requires significant overhead in adjusting the envelope values obtained from the table, which is essentially what the envelope lookup procedure was designed to avoid in the first place.

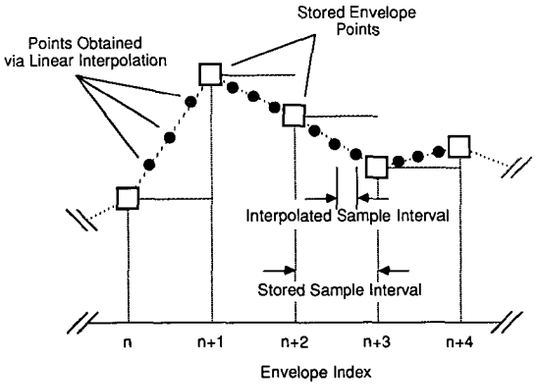


Fig. 5. Linear interpolation to increase effective time resolution of stored envelope function.

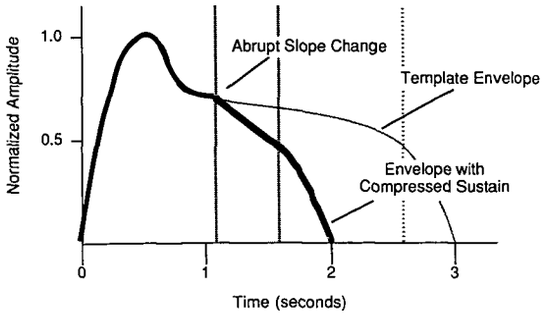


Fig. 6. Time compression of envelope function. In this example, sustain (middle) portion of envelope has been compressed while attack and release have been left unchanged. Note abrupt slope discontinuity introduced between attack and sustain segments.

Slope discontinuities in the warped envelope are caused by the abrupt change in time scale (the implicit envelope sample rate) from one segment of the envelope to the next. This observation can be better understood by considering the lookup phase function, denoted by $\phi(t)$, as shown in Fig. 7, where the abscissa value is time and the ordinate is the lookup address, or index.[1] For the unwarped envelope (the template), the lookup phase is a linear function with unity slope, that is, the kth value of the envelope is read from the table at envelope time $t = k\Delta$, where Δ is the time interval between the stored envelope samples. If a segment of the envelope is warped to expand the total envelope duration, the phase function during that segment has a slope less than unity, implying that a greater time span is required to traverse a given range of lookup addresses. Similarly, an envelope segment scanned faster than the template corresponds to a phase slope greater than unity. For continuous time functions the instantaneous slope of the phase curve $d\phi/dt$ is the lookup angular frequency, while for discrete-time envelope functions the phase slope $\Delta\phi/\Delta t$ is better described as a lookup sample increment.

In order to reduce the lookup function slope discontinuity effect while still retaining the computation efficiency of the lookup operation, it is necessary to modify the table lookup mechanism so that the sample increment (phase slope) is varied smoothly from one segment of the envelope to the next. We would like to determine a smooth lookup phase function constrained by the desired duration of each frame, thereby maintaining the expected duration of the entire envelope. In other words, we need to choose a smooth function with specific envelope phase value (the table lookup index) and phase slope (table sample increment) at the time corresponding to the envelope segment boundaries. This pair of constraints implies at least a third-order function, so a cubic polynomial, or spline, was chosen for the lookup function [7].

[1] Note that the *phase* referred to is the phase of the envelope function, not the waveform or signal being modulated.

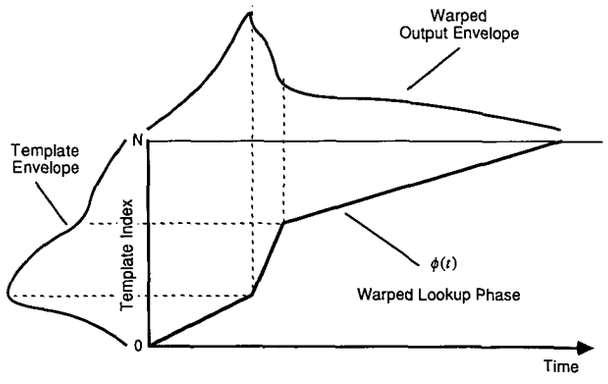


Fig. 7. Time-warped envelope produced from template envelope using nonlinear lookup function, denoted by $\phi(t)$. Unwarped envelope would be generated if $\phi(t)$ were a straight line with slope 1.

3 ENVELOPE TABLE LOOKUP USING CUBIC FUNCTIONS

Consider a simple envelope table (template) with S segments, as shown in Fig. 8. Each segment s of the template has a particular duration τ_s, resulting in a total envelope duration of $(\tau_1 + \tau_2 + \cdots + \tau_S) = T$. If a separate time modification factor μ_s is applied to each segment, the total duration becomes $(\mu_1\tau_1 + \mu_2\tau_2 + \cdots + \mu_S\tau_S)$, or $(\tilde{\tau}_1 + \tilde{\tau}_2 + \cdots + \tilde{\tau}_S) = \tilde{T}$, where $\tilde{\tau}_s$ is the warped duration of segment s, and $\tilde{T}$ is the warped envelope duration. The table lookup operation again can be described in terms of a lookup phase function, as shown in Fig. 9. The broken line in the figure represents a lookup phase function with abrupt changes in slope. This three-segment example function can be described by Eq. (1), using a continuous time representation for convenience. The function is a collection of linear segments:

$$\phi_{\text{lookup}}(t) = \begin{cases} \dfrac{1}{\mu_1} t\ , & 0 \leq t < \tilde{\tau}_1 \\ \dfrac{1}{\mu_2} t + \left(1 - \dfrac{\mu_1}{\mu_2}\right)\tau_1\ , & \tilde{\tau}_1 \leq t < (\tilde{\tau}_1 + \tilde{\tau}_2) \\ \dfrac{1}{\mu_3} t + \left(1 - \dfrac{\mu_1}{\mu_3}\right)\tau_1 + \left(1 - \dfrac{\mu_2}{\mu_3}\right)\tau_2\ , & (\tilde{\tau}_1 + \tilde{\tau}_2) \leq t < \tilde{T}\ . \end{cases} \tag{1}$$

3.1 Specification of the Cubic Functions

In order to develop the desired smooth lookup phase function, each of the linear segments is replaced by a cubic spline (solid line in Fig. 9) under the following guidelines:

1) The lookup phase function for the table consists of continuous, concatenated cubic functions (one function per envelope segment) with continuous first and second derivatives at the break points between segments. The segment break points occur at the same phase coordinates (time, envelope phase) as the linear-segment case of Eq. (1).

2) The slope of the phase function at a break point between segments within the table is chosen to be equal to the slope of a line connected between the two adjacent break points. For example, the slope chosen for the break point between segment s and segment $s + 1$ of a simple envelope (Fig. 10) is given by

$$\text{slope} = \frac{\tau_s + \tau_{s+1}}{\mu_s\tau_s + \mu_{s+1}\tau_{s+1}} \quad \text{or} \quad \frac{\tau_s + \tau_{s+1}}{\tilde{\tau}_s + \tilde{\tau}_{s+1}}\ . \tag{2}$$

This choice balances the length and slope of the individual segments to help confine the cubic functions to the vicinity of the linear connecting segments. The slope at the start of segment s is denoted by $m_{s,0}$, while the slope at the end of the segment is denoted by $m_{s,1}$. Note that $m_{s,1}$ must equal $m_{s+1,0}$ since they refer to the same break point.

3) A cubic polynomial is selected for each envelope segment. Again using a continuous time function for convenience, the cubic equation for each segment s is of the form $A_st^3 + B_st^2 + C_st + D_s$, and the slope at any point is given by $3A_st^2 + 2B_st + C_s$. If we define a local coordinate system such that the coordinates and slope at the first point ($t = 0$) of the function are (0, 0) and $m_{s,0}$, respectively, and at the last point ($t = \tilde{\tau}_s$) of the segment, ($\tilde{\tau}_s$, τ_s) and $m_{s,1}$, it is apparent that

$$\begin{aligned} C_s &= m_{s,0} \\ D_s &= 0\ . \end{aligned} \tag{3}$$

The remaining coefficients, A_s and B_s, can be determined using the coordinates and slope at the other end of the

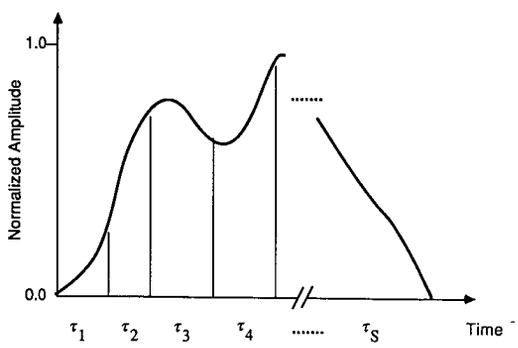


Fig. 8. Envelope function divided into S nonoverlapping sections. Each segment has a particular duration, denoted by τ_s.

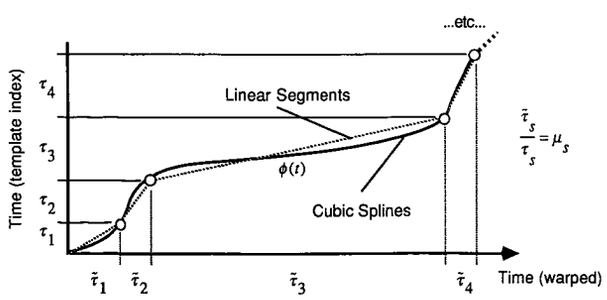


Fig. 9. Envelope lookup phase function represented as a concatenated set of smooth cubic-spline functions with matched slopes at each segment endpoint.

segment, namely,

$$\begin{aligned} \tau_s &= A_s\tilde{\tau}_s^3 + B_s\tilde{\tau}_s^2 + m_{s,0}\tilde{\tau}_s \\ m_{s,1} &= 3A_s\tilde{\tau}_s^2 + 2B_s\tilde{\tau}_s + m_{s,0} \ . \end{aligned} \tag{4}$$

Solution for the coefficients gives

$$\begin{aligned} A_s &= \frac{1}{\tilde{\tau}_s^2}\left(m_{s,0} + m_{s,1} - \frac{2\tau_s}{\tilde{\tau}_s}\right) \\ &= \frac{1}{\tilde{\tau}_s^2}\left(m_{s,0} + m_{s,1} - \frac{2}{\mu_s}\right) \\ B_s &= \frac{1}{\tilde{\tau}_s}\left(\frac{3\tau_s}{\tilde{\tau}_s} - 2m_{s,0} - m_{s,1}\right) \\ &= \frac{1}{\tilde{\tau}_s}\left(\frac{3}{\mu_s} - 2m_{s,0} - m_{s,1}\right) \ . \end{aligned} \tag{5}$$

4) The slope of the phase function at the beginning point and ending point of the envelope cannot be determined using the adjacent break-point formula given previously for the segment boundaries within the table. These slopes can be set to an arbitrary value, such as unity or zero, or some other choice can be made. We initially chose to select the initial (final) slope in order to minimize the root-mean-squared curvature of the first (last) lookup function segment. The curvature at any point is defined to be $6A_st + 2B_s$ (the second derivative of the cubic lookup phase function). In other words, the initial and final slopes can be chosen such that the lookup phase function is as nearly linear as possible during those segments. Since the minimum of the rms curvature occurs for the same value of initial (or final) slope as the integrated squared curvature, minimization for the initial segment can be performed using

$$\min\left\{\int_0^{\tilde{\tau}_s} (6A_st + 2B_s)^2 \, dt\right\} \tag{6}$$

with respect to $m_{1,0}$. (Recall that A_s and B_s are functions fo $m_{1,0}$ for the first segment.) Solution of this minimization gives the result

$$m_{1,0} = \frac{1}{2}\left(\frac{3\tau_1}{\tilde{\tau}_1} - m_{1,1}\right) = \frac{1}{2}\left(\frac{3}{\mu_1} - m_{1,1}\right) \ . \tag{7}$$

This causes the quadratic coefficient B_1 to be zero. A similar result can be obtained for $m_{s,1}$ on the final segment. The equations defining the lookup function slope at each segment boundary could be chosen using some other rationale. In particular, the slopes could be selected in order to allow a desired lookup function shape (such as linear) during particular envelope segments. This strategy would allow any distortion of the envelope shape caused by nonlinear lookup functions to be confined to certain envelope segments where the distortion would be less noticeable. For example, the attack portion of the envelope (where the amplitude often changes rapidly) is usually of great perceptual importance. By restricting the lookup function to be linear during the attack, the correct timbral character can be preserved at the expense of some other envelope segment with more constant amplitude. The examples considered in Sec. 5 were generated using a linear lookup segment during the attack portion of the envelope.

5) A significant characteristic of the cubic-spline formulation is, clearly, that restrictions are placed only on the value and slope of the curve at the break points, not on the shape or extent of the cubic function between break points. The result is that certain combinations of segment durations and extreme time-warping factors can result in cubic lookup functions containing negative slopes or excursions outside the desired envelope segment, as depicted in Fig. 11. This occurs only when adjacent-segment time warps differ by a large factor, such as an unwarped attack segment with a significantly stretched sustain. However, in these situations the expected quality improvement of a smooth lookup phase function compared to a simple linear lookup segment

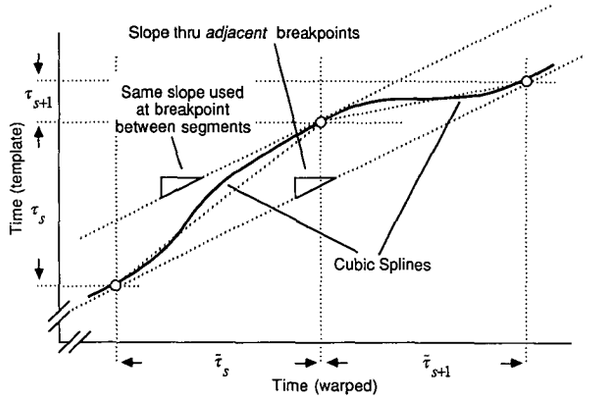


Fig. 10. Slope of phase function at segment boundaries within table can be chosen parallel to a line connecting two adjacent break points. This helps confine behavior of cubic splines between break points.

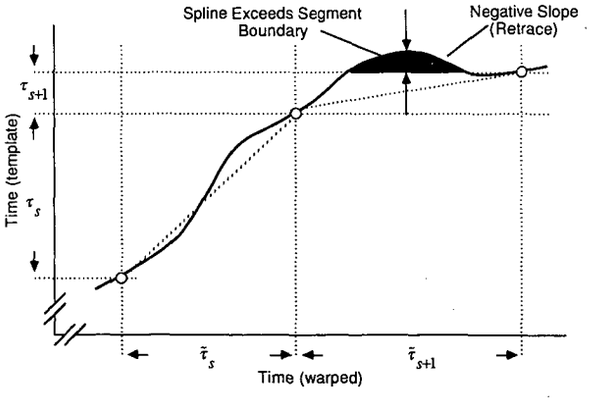


Fig. 11. Certain combinations of segment durations and time-warping factors may generate unacceptable lookup function which goes beyond end of desired envelope segment or retraces part of template. This situation must be detected and corrected.

is marginal anyway because the lookup phase slope must change abruptly in the vicinity of the segment boundary, no matter whether line segments or splines are used. Thus a sensible kludge to this problem (during the coefficient calculations) has been simply to check the slope of the lookup function evaluated at the extremum of the curvature (second derivative): if this minimum slope is negative, the cubic function is abandoned and a linear lookup segment is used instead.

It should be noted that the spline formulation could also be constrained by subdividing the segment into two or more smaller segments with additional constraints added for each of the subsegments. This approach was not incorporated into the implementation described here for the reasons of computational efficiency and simplicity described previously.

3.2 Calculation of the Coefficients

The calculation of the cubic polynomial lookup phase functions can be summarized as follows:

1) The durations of each envelope segment $\tilde{\tau}_s$, are chosen according to the particular synthesis context.

2) The coordinates $(\tilde{\tau}_s, \tau_s)$ are determined, and the slopes $m_{s,1}$ $(=m_{s+1,0})$ at each break point *within* the table warping function are calculated using Eq. (2).

3) The initial slope $m_{1,0}$ and the final slope $m_{s,1}$ are calculated using Eq. (7) or some other criteria.

4) Finally, the cubic lookup phase coefficients A_s, B_s, C_s, and D_s for each segment are calculated using Eqs. (3) and (5). The resulting function is checked—a linear lookup function is substituted if the cubic function contains any portions with negative slope.

4 IMPLEMENTATION OF TABLE LOOKUP VIA FORWARD DIFFERENCES

Computation of the cubic-spline coefficients for each segment of the envelope lookup phase function occurs once at the start of the note. The value of the lookup function must be calculated for every output sample from the envelope, however, so explicit evaluation of the cubic polynomial is likely to be an expensive computational decision.

Instead we chose to use the widely known forward-differences strategy to generate the polynomial values recursively [8], [9]. Representing the *next* value of the polynomial by adding an increment to the *current* value, the polynomial computation becomes a series of additions. For example, to calculate a polynomial function f,

$$f_{i+1} = f_i + \Delta f_i \tag{8}$$

where Δf_i is the first forward difference, f_i is the function evaluated at discrete time t_i, and f_{i+1} is the function evaluated at time $t_{i+1} = t_i + \delta$, with δ being the time step size.

For the cubic polynomial lookup phase function

$$\phi_{\text{lookup},s}(t) = \phi_i = A_s t^3 + B_s t^2 + C_s t + D_s$$

the difference form becomes

$$\phi_{\text{lookup},s}(t + \delta) = \phi_{i+1} = A_s(t + \delta)^3 + B_s(t + \delta)^2 + C_s(t + \delta) + D_s \tag{9}$$

or

$$\phi_{i+1} = \phi_i + \Delta\phi_i \tag{10}$$

where

$$\Delta\phi_i = 3A_s\delta t_i^2 + (3A_s\delta^2 + 2B_s\delta)t_i + A_s\delta^3 + B_s\delta^2 + C_s\delta \; . \tag{11}$$

Noting that $\Delta\phi_i$ is a function of the time index t_i, the process is extended to a second forward difference, $\Delta^2\phi_i$, in order to calculate the next value of the first forward difference:

$$\Delta\phi_{i+1} = \Delta\phi_i + \Delta^2\phi_i = \Delta\phi_i + 6A_s\delta^2 t_i + 6A_s\delta^3 + 2B_s\delta^2 \; . \tag{12}$$

Finally, the third forward difference is a constant value given by

$$\Delta^3\phi_i = 6A_s\delta^3 \; . \tag{13}$$

The cubic function and each forward difference are set to initial values at the beginning of each segment. For a lookup phase segment as defined in Sec. 3 the initial values begin at $t = 0$, yielding

$$\begin{aligned} \phi_0 &= D_s = 0 \\ \Delta\phi_0 &= A_s\delta^3 + B_s\delta^2 + C_s\delta \\ \Delta^2\phi_0 &= 6A_s\delta^3 + 2B_s\delta^2 \\ \Delta^3\phi_0 &= 6A_s\delta^3 \end{aligned} \tag{14}$$

where A_s, B_s, C_s, and D_s are the coefficients calculated in Eqs. (3) and (5).

The recursive calculation of the lookup phase function at time δ becomes

$$\begin{aligned} \phi_1 &= \phi_0 + \Delta\phi_0 \\ \Delta\phi_1 &= \Delta\phi_0 + \Delta^2\phi_0 \\ \Delta^2\phi_1 &= \Delta^2\phi_0 + \Delta^3\phi_0 \end{aligned} \tag{15}$$

or, in general, at time $i \cdot \delta$ during the segment,

$$\begin{aligned} \phi_{i+1} &= \phi_i + \Delta\phi_i \\ \Delta\phi_{i+1} &= \Delta\phi_i + \Delta^2\phi_i \\ \Delta^2\phi_{i+1} &= \Delta^2\phi_i + \Delta^3\phi_0 \; . \end{aligned} \tag{16}$$

Note that the forward difference calculations can be streamlined by normalizing with respect to the sample rate, thereby fixing the explicit time step size δ in Eqs. (14) to be unity.

The forward-difference implementation requires that the increment values be stored with sufficient mathematical precision to maintain the desired resolution even after many additions during a long envelope segment. The implementation used for this investigation uses 32-bit floating-point numbers.

5 EXAMPLES AND DISCUSSION

The particular topology of a synthesis algorithm using several envelope lookup routines may require the nonlinear lookup function to prevent problems on some envelope generators, but not others. In general, the computation and parameter storage overhead associated with the cubic lookup function method is not significant in non-real-time systems using complicated synthesis algorithms, but the overhead may be a consideration when dealing with extreme time constraints or simple synthesis algorithms.

Two elementary example synthesis algorithms using amplitude envelopes are presented in this section. The examples using the spline lookup function method were generated under the constraint that the lookup function during the attack portion of the envelope must be linear in order to reduce timbral alteration (that is, $m_{1,0} = m_{1,1} = 1/\mu_1$). This constraint is accomplished easily in the initialization procedure by setting the first forward difference of the attack segment equal to the desired envelope phase slope and setting the higher order differences to zero.

5.1 Example 1: Simple AM

The first example is an elementary synthesis algorithm using two unit generators, as shown in Fig. 12. The algorithm consists of a fixed-waveform digital oscillator whose amplitude is controlled by a single envelope function (simple amplitude modulation). The waveform from the digital oscillator and the template envelope function together produce a vaguely trumpetlike sound.

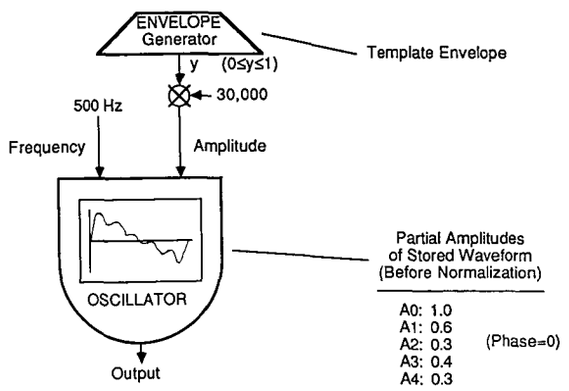


Fig. 12. Example synthesis algorithm comprising fixed-frequency, fixed-waveform oscillator with amplitude modulation applied by envelope generator.

The manual division of the template envelope (1 s in total duration) into three segments is shown in Fig. 13.

The duration of the template envelope expanded from its nominal 1-s duration to 2 s is shown in Fig. 14 for the simple linear-segment lookup method and for the

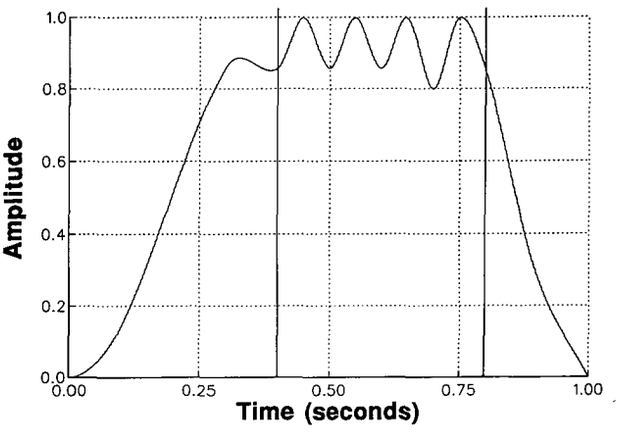
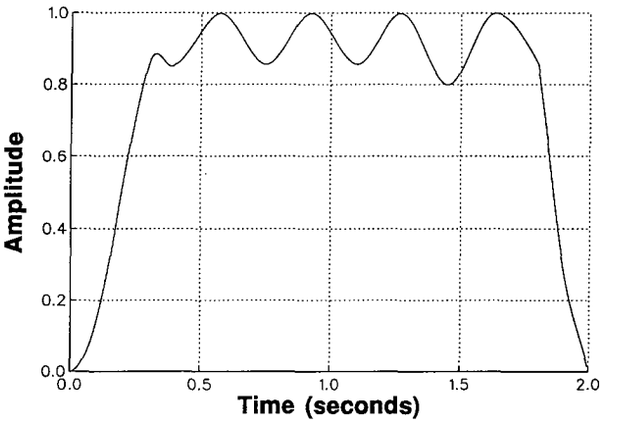
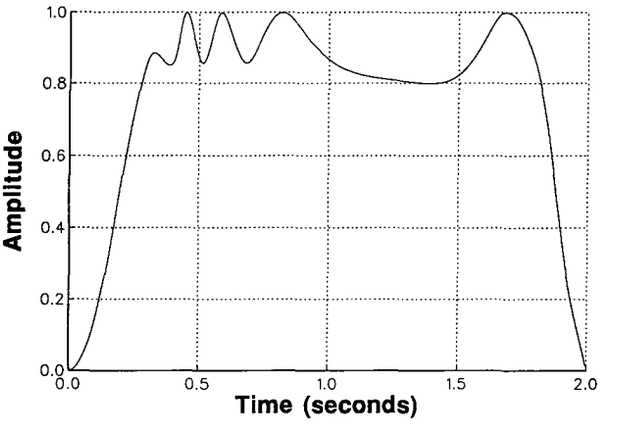


Fig. 13. Example template envelope (used in algorithm of Fig. 11) divided into three segments of durations (0.4, 0.4, 0.2) s.



(a)



(b)

Fig. 14. Envelope of Fig. 12 expanded in overall duration to 2 s by time scaling sustain (middle) portion. (a) Expansion using linear-segment lookup function. (b) Expansion using cubic-spline lookup function.

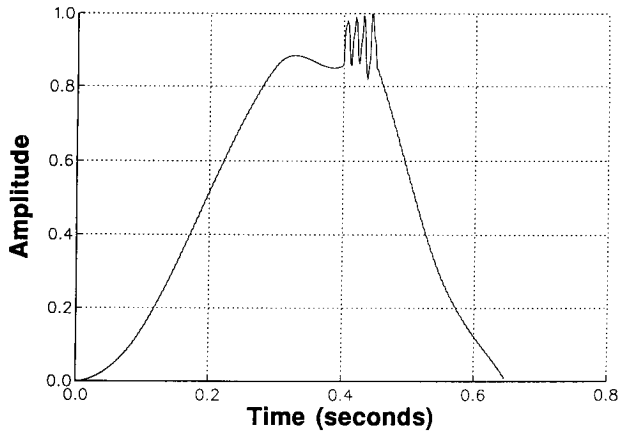
cubic lookup function method of Sec. 3. The curves in Fig. 14 were obtained using a normalized peak detector on the output waveform from the synthesis algorithm. For this example the doubled duration is accomplished by increasing only the duration of the middle (sustain) envelope segment: $(\tau_1, \tau_2, \tau_3) = (0.4, 0.4, 0.2)$, $(\tilde{\tau}_1, \tilde{\tau}_2, \tilde{\tau}_3) = (0.4, 1.4, 0.2)$, giving $(\mu_1, \mu_2, \mu_3) = (1.0, 3.5, 1.0)$. The linear segment method results in an unnatural broadening of the amplitude variations (tremelo), while the cubic lookup provides a more gradual transition from attack to sustain and from sustain to release. The informal perceptual result is that the envelope of Fig. 14(a) has an unnaturally broadened tremelo and sounds like a time-stretched envelope, while Fig. 14(b) retains a more muscial character.

Compressing the envelope from its nominal 1-s duration to 0.65 s is shown in Fig. 15, again for the linear and cubic lookup functions. The reduced duration is accomplished by decreasing the duration of the middle segment: $(\tilde{\tau}_1, \tilde{\tau}_2, \tilde{\tau}_3) = (0.4, 0.05, 0.2)$, giving $(\mu_1, \mu_2, \mu_3) = (1.0, 0.125, 1.0)$. The extreme compression warping of the sustain results in rapid amplitude fluctuations, which are perceived as a short click or noise burst in either the linear segment or the spline lookup cases. Either case is typically unacceptable. A somewhat more useful example of compression is shown in Fig. 16. In this case the compression to 0.65 s is spread over all three segments: $(\tilde{\tau}_1, \tilde{\tau}_2, \tilde{\tau}_3) = (0.2, 0.35, 0.1)$, yielding $(\mu_1, \mu_2, \mu_3) = (0.5, 0.875, 0.5)$. The sustain portion of the envelope is improved at the expense of a time-compressed attack segment.

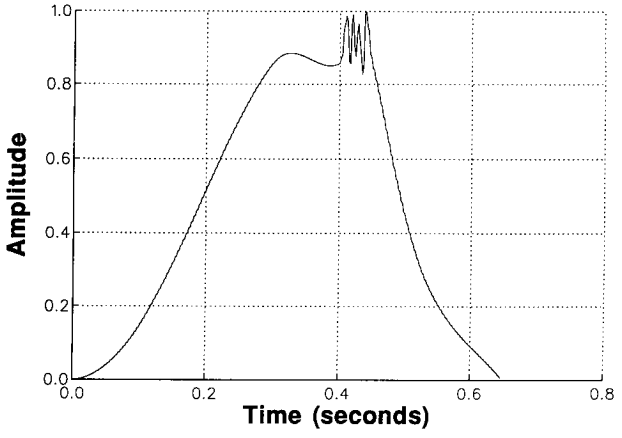
5.2 Example 2: FM

The second example is a more complicated synthesis algorithm utilizing frequency modulation (FM), high-Q low-pass filtering, and an overall amplitude envelope. In this example, depicted in Fig. 17, the FM carrier and modulator are both in the audio-frequency range, resulting in spectral sidebands [10]. For simplicity the same envelope shape is used to control both the modulation index and the output amplitude. The manual division of the amplitude and modulation index envelope into three segments is shown in Fig. 18.

Expansion of only the sustain segment to give a total duration of 2 s is shown in Fig. 19 for the linear segment and the cubic-spline lookup functions: $(\tilde{\tau}_1, \tilde{\tau}_2, \tilde{\tau}_3) = (0.15, 1.75, 0.1)$, yielding $(\mu_1, \mu_2, \mu_3) = (1.0, 2.333, 1.0)$. Like the previous synthesis examples, the curves

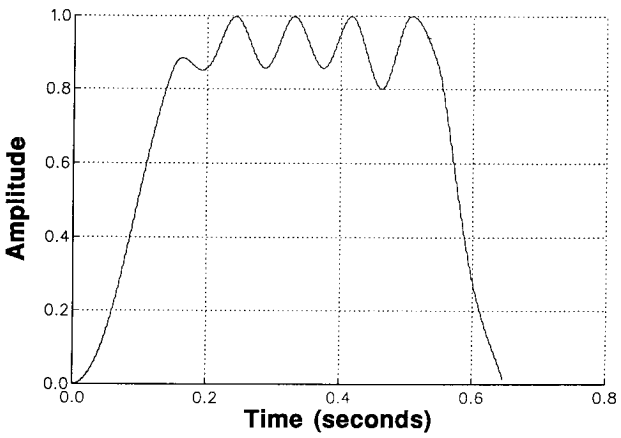


(a)

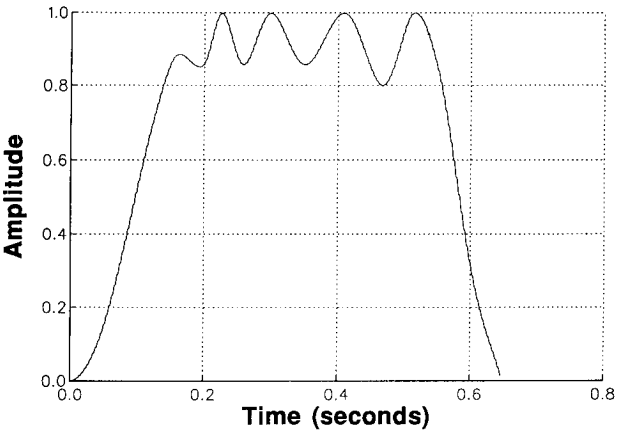


(b)

Fig. 15. Envelope of Fig. 12 compressed in overall duration to 0.65 s by time scaling sustain (middle) portion. (a) Compression using linear-segment lookup function. (b) Compression using cubic-spline lookup function.



(a)



(b)

Fig. 16. Envelope of Fig. 12 compressed in overall duration to 0.65 s by time scaling all three segments by factors (0.5, 0.875, 0.5). (a) Compression using linear-segment lookup function. (b) Compression using cubic-spline lookup function.

of Fig. 19 were obtained using a normalized peak detector on the output waveform from the algorithm of Fig. 17. The abrupt slope discontinuities at envelope segment boundaries present in the linear-segment case [Fig. 19(a)] are avoided using the cubic-spline lookup method [Fig. 19(b)]. Although the effects of the slope discontinuities are not significant in this example, problems similar to those shown previously in Fig. 3 can occur under certain conditions. In this elementary example the problem can be avoided to some extent by placing the envelope generator *after* the filter, but such a change of topology may not always be possible.

Another form of envelope warping is shown in Fig. 20. For this example the attack and release segment durations are cut in half while the overall duration is held at 1 s by increasing the duration of the sustain: $(\tilde{\tau}_1, \tilde{\tau}_2, \tilde{\tau}_3) = (0.075, 0.875, 0.05)$, giving $(\mu_1, \mu_2, \mu_3) = (0.5, 1.1667, 0.5)$. Note in particular the smooth transitions between envelope segments for the cubic-spline case [Fig. 20(b)].

6 CONCLUSION

We have described the use of nonlinear lookup functions (specifically cubic polynomials) for time warping segments of stored envelope functions in digital synthesis systems. The nonlinear time warping lessens the likelihood of audible timbral discontinuities between envelope segments with differing time warps by avoiding abrupt changes in the slope of the table lookup function, while preserving the details of the stored envelope function itself. Although the conventional linear time warping technique is completely satisfactory in many situations, the new nonlinear approach can be applied usefully in complex digital synthesis algorithms. The method requires precalculation of the coefficients for each cubic lookup segment followed by "on the fly" calculation of the cubic functions themselves using the well-known forward differences formulation. Computer simulations have been used to demonstrate the method.

7 ACKNOWLEDGMENT

Initial experiments involving the envelope lookup method described in this paper were performed by the author using the facilities of the University of Illinois Computer Music Project, James Beauchamp, Director. Comments and suggestions by two anonymous reviewers have been of great assistance in improving the focus

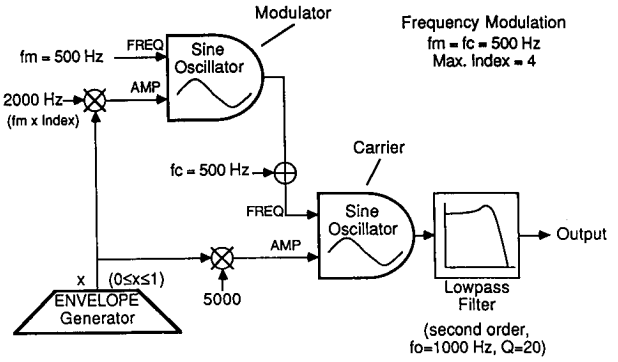


Fig. 17. Frequency modulation (FM) synthesis algorithm using modulation index, amplitude envelopes, and low-pass filter.

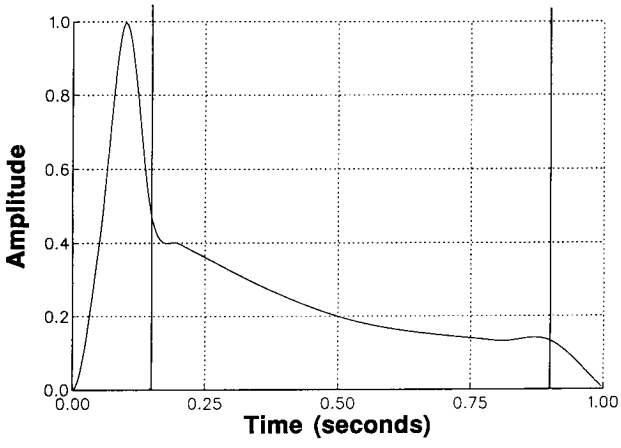
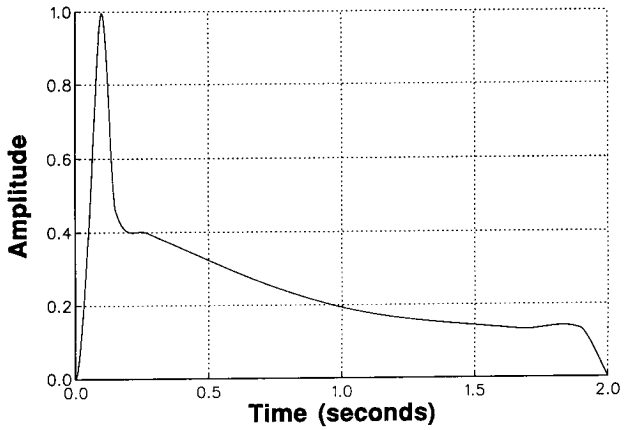
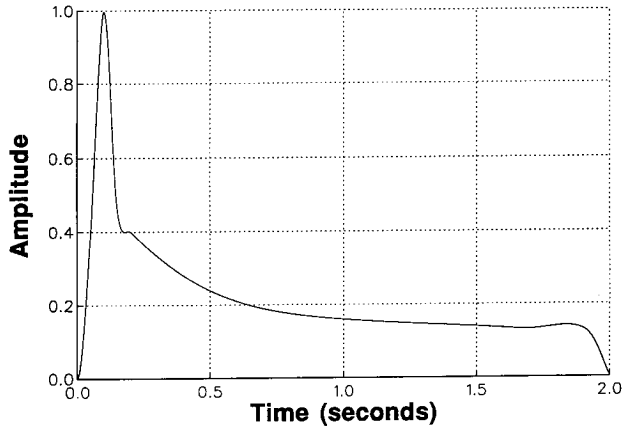


Fig. 18. Example template envelope (used in algorithm of Fig. 16) divided into three segments of durations (0.15, 0.75, 0.1) s.



(a)



(b)

Fig. 19. Envelope of Fig. 17 expanded in overall duration to 2 s by time scaling sustain (middle) portion. (a) Expansion using linear-segment lookup function. (b) Expansion using cubic-spline lookup function.

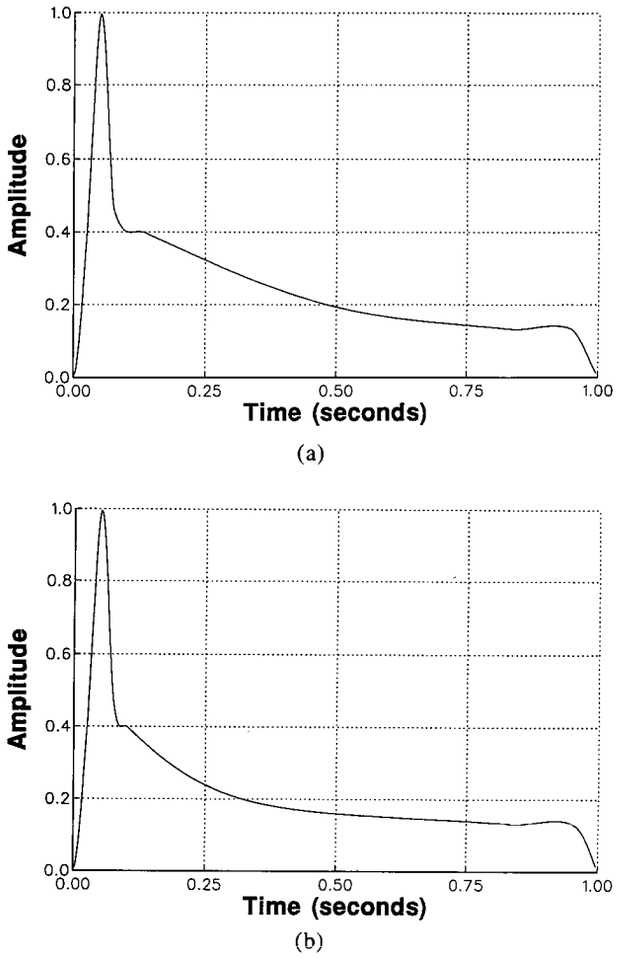


Fig. 20. Envelope of Fig. 17 with time scaling of all three segments by factors (0.5, 1.1667, 0.5), keeping overall duration 1 s. (a) Using linear-segment lookup function. (b) Using cubic-spline lookup function.

and clarity of this paper. The author also wishes to acknowledge Tony Agnello of Ariel Corporation for his assistance to the University of Nebraska.

8 REFERENCES

[1] J. Grey, "An Exploration of Musical Timbre," Ph.D. dissertation, Dept. of Music, Rep. STAN-M-2, Stanford University, Stanford, CA, 1975.

[2] M. V. Mathews, *The Technology of Computer Music*. (M.I.T. Press, Cambridge, MA, 1969).

[3] F. R. Moore, *Elements of Computer Music*, (Prentice-Hall, Englewood Cliffs, NJ, 1990).

[4] C. Dodge and T. A. Jerse, *Computer Music: Synthesis, Composition, and Performance*. (Schirmer, New York, 1985).

[5] C. Scaletti, "The Kyma/Platypus Computer Music Workstation," *Computer Music J.*, vol. 13, no. 2, pp. 23–38, 1989.

[6] D. Jaffe and L. Boynton, "An Overview of the Sound and Music Kits for the NeXT Computer," *Computer Music J.*, vol. 13, no. 2, pp. 48–55, 1989.

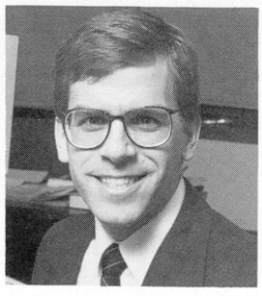
[7] E. Kreyszig, *Advanced Engineering Mathematics*. (Wiley, New York, 1979).

[8] Y. Mitsuhashi, "Musical Sound Synthesis by Forward Differences," *J. Audio Eng. Soc.*, vol. 30, pp. 2–9 (1982 Jan./Feb.).

[9] D. Hearn and M. P. Baker, *Computer Graphics*. (Prentice-Hall, Englewood Cliffs, NJ, 1986).

[10] J. M. Chowning, "The Synthesis of Complex Audio Spectra by Means of Frequency Modulation," *J. Audio Eng. Soc.*, vol. 21, pp. 526–534 (1973 Sept). Reprinted in *Foundations of Computer Music*, C. Roads and J. Strawn, Eds. (M.I.T. Press, Cambridge, MA, 1985).

THE AUTHOR



Robert C. (Rob) Maher was born in 1962 in Cambridge, U.K., of American parents. He holds a B.S. degree from Washington University in St. Louis (1984), an M.S. degree from the University of Wisconsin-Madison (1985), and a Ph.D. from the University of Illinois-Urbana (1989), all in electrical engineering. He has received several prestigious academic awards, including a four-year, full-tuition Langsdorf fellowship, a National Science Foundation Graduate Fellowship, a University of Illinois Graduate Fellowship, and an Audio Engineering Society Educational Grant. While a student, he also worked as a graduate research assistant for the University of Illinois Computer Music Project.

Dr. Maher is a member of the Tau Beta Pi, Eta Kappa Nu, and Phi Kappa Phi honor societies, and several professional organizations including the Audio Engineering Society, IEEE, ASA, ASEE, ICMA, and IMA. He is currently an assistant professor of electrical engineering at the University of Nebraska-Lincoln, with teaching and research interests in the application of advanced digital signal processing methods in audio engineering, electroacoustics, and computer music.