Synthesis of Trumpet Tones Using a Wavetable and a Dynamic Filter*

ANDREW HORNER, *AES Member*

Department of Computer Science, Hong Kong University of Science and Technology, Kowloon, Hong Kong

AND

JAMES BEAUCHAMP, *AES Fellow*

School of Music and Department of Electrical and Computer Engineering, University of Illinois, Urbana, IL 61820 USA

The spectral response of the trumpet varies dramatically depending on its performed pitch and dynamic. The shape of a spectrum can be measured in terms of its spectral centroid, which is closely related to subjective brightness. We observe that trumpet spectra with the same centroid have similar spectral shapes regardless of their pitches. Based on a training set of 15 trumpet tones, a family of spectral envelopes has been created, each formed by averaging spectra with similar centroid values. For synthesis, a source spectrum is first sampled from the highest centroid spectral envelope. The remaining spectral envelopes are approximated by means of a variable low-pass filter. For each note produced, the filter is swept according to a time-varying centroid. Amplitude and frequency control functions are also matched. The result is a method that efficiently synthesizes convincing trumpet tones across the instrument's pitch and dynamic ranges using a wavetable and a centroid-controlled filter.

0 INTRODUCTION

The trumpet's spectral response varies as a function of each tone's pitch and amplitude. Composers and synthesizer designers want instrument models that efficiently produce these richly varied tones.

To synthesize realistic trumpet tones it is not sufficient to synthesize trumpetlike spectra; it is also necessary to recreate realistic time-variant behavior of the spectra. Several researchers have used time-varying additive synthesis successfully [1]–[3], and this has become a standard technique, even though the number of control functions required is quite excessive. In 1982 Beauchamp [4] reported that musical instrument tones could be emulated with nonlinear-filter synthesis controlled by time-varying amplitude, fundamental frequency, and spectral centroid ("brightness") functions analyzed from prototype musical instrument sounds. However, the accuracy of this model was limited by the fact that only one characteristic spectrum was used to compute the model for a particular instrument.

The spectral centroid is a measure of timbral brightness and provides a good indication of wind instrument spectra [5]. Intuitively the spectral centroid is the frequency that splits a spectrum into two halves that balance each other. Thus a large centroid value indicates a bright spectrum, whereas a small centroid indicates a spectrum less bright, or dull (mellow or dark). Fig. 1 illustrates the difference between these two types of spectra.

We define the time-varying spectral centroid throughout this paper as

$$f_{\text{cg}}(t) = f_{\text{a}} \left[\frac{\sum_{k=1}^{N_{\text{har}}} k a_k(t)}{\sum_{k=1}^{N_{\text{har}}} a_k(t)} - 1 \right] \tag{1}$$

where N_{har} is the number of harmonics, f_{a} is the average fundamental frequency, and $a_k(t)$ is the amplitude envelope of the kth harmonic. The -1 term in Eq. (1) effectively shifts the spectrum to the left by the fundamental frequency, ensuring that the minimum centroid value is zero, regardless of the fundamental frequency. We find

* Manuscript received 1995 February 27.

this centroid definition useful for comparing centroids of tones with different fundamental frequencies and will use it throughout the remainder of this paper, referring to it simply as "centroid." Another definition we use is the time-varying rms amplitude,

$$\mathrm{rms}(t) = \sqrt{\textstyle\sum_{k=1}^{N_{\mathrm{har}}} a_k^2(t)}\,. \tag{2}$$

Trumpet spectra with equal centroids usually have very similar spectral shapes. For example, Fig. 2 shows three pairs of equal-centroid spectra taken from the attack and decay portions of a 588-Hz trumpet tone. The centroid of the first pair is 500 Hz, of the second 1000 Hz, and of the third 1500 Hz. Although there are some differences between the two spectra of each equal-centroid pair, they have very similar shapes. Since each pair contains spectral snapshots from completely different portions of the tones, it is evident that centroid is a good predictor of spectral similarity.

Moreover, spectra with the same centroid taken from different trumpet tones often have similar spectral shapes, even when their fundamental frequencies are different. For example, the two spectral envelopes shown in Fig. 3 come from equal-centroid portions of low- and midrange trumpet tones and have quite similar spectral shapes, including a strong resonance near 6000 Hz. Note that spectral envelopes provide a good means of describing persistent spectral features, such as the gradual downward slope, the valley at 5000 Hz, and the strong resonance near 6000 Hz exhibited by both curves in Fig. 3. A single spectral envelope could represent both curves with reasonable accuracy.

In some pioneering work Luce and Clark [6] derived steady-state spectral envelopes for the dynamics *pp*, *mf*, and *ff* for various brass instruments, including the trumpet. They took their data from a large number of pitches, although they were limited to 11 harmonics per note. Since their model does not include a specification of the time-variant behavior of the spectra, it has limited usefulness for synthesis. However, Slaymaker [7], [8] has used impulse responses derived from single spectral envelopes [9] to synthesize a number of musical instruments over wide pitch ranges.

While, as is well known, the trumpet's spectral centroid increases with increasing amplitude, its peak value, as shown in Fig. 4, remains relatively constant (at about 2000 Hz) over the instrument's pitch range. This contrasts with the instrument's peak rms amplitude, which increases with frequency (see Fig. 4). The relative independence of centroid and pitch suggests that they should form a useful complementary pair in modeling spectral variations. Therefore we pursue a centroid-based, rather than amplitude-based, model in this paper.

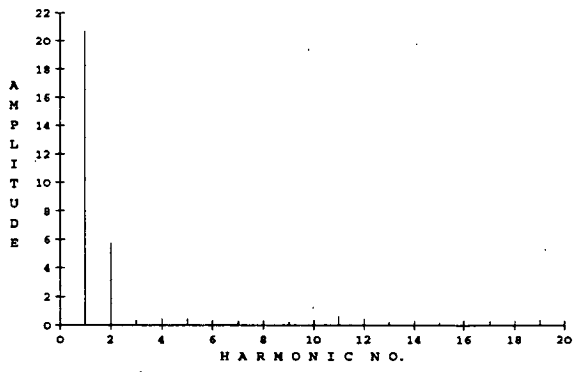
Section 1 outlines the spectral envelope analysis and synthesis model, which takes advantage of the strong correspondence between centroid and spectral shape. This model extends Luce and Clark's spectral envelope model and Beauchamp's nonlinear synthesis model using multiple spectral envelopes to capture the response of the trumpet at different centroid levels, thus providing full control over the dynamics of the trumpet. Section 2 shows how we derive and control an efficient wavetable and filter synthesis instrument that approximates the basic spectral envelope model. Section 3 presents and evaluates results generated by the wavetable and filter implementation.

1 DYNAMIC SPECTRAL ENVELOPE MODEL

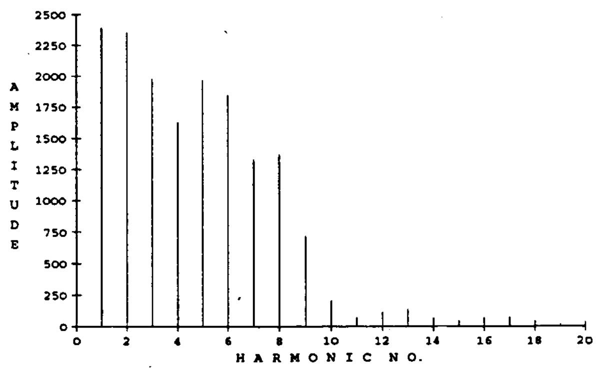
The spectral model proposed in this section uses a family of spectral envelopes to capture the response of a trumpet across its full pitch and dynamic ranges. It also proposes to represent the instrument's time-varying response, including attack and decay. We compute the spectral envelopes by averaging 10 groups of similar-centroid spectra from a training set of trumpet tones performed at various dynamic levels. Time-variant spectra are then synthesized from these spectral envelopes by sampling them at harmonic frequencies of the desired fundamental and by matching the output spectra to achieve the time-varying centroid, rms amplitude, and fundamental frequency of a prototype sound. This section describes the derivation of the spectral envelope model and its verification through additive synthesis.

1.1 Spectral Envelope Modeling

We construct several spectral envelopes by computing average spectral responses of the instrument correspond



(a)



(b)

Fig. 1. (a) Low-centroid spectrum. (b) High-centroid spectrum.

ing to different ranges of centroid values. For example, one spectral envelope corresponds to the response of the instrument at low centroid values, which occur during its attack and final decay, while another corresponds to the trumpet's peak steady-state response, when the centroid value is at its maximum. This model therefore assumes that the spectral response of the instrument changes strictly as a function of its centroid. Instruments

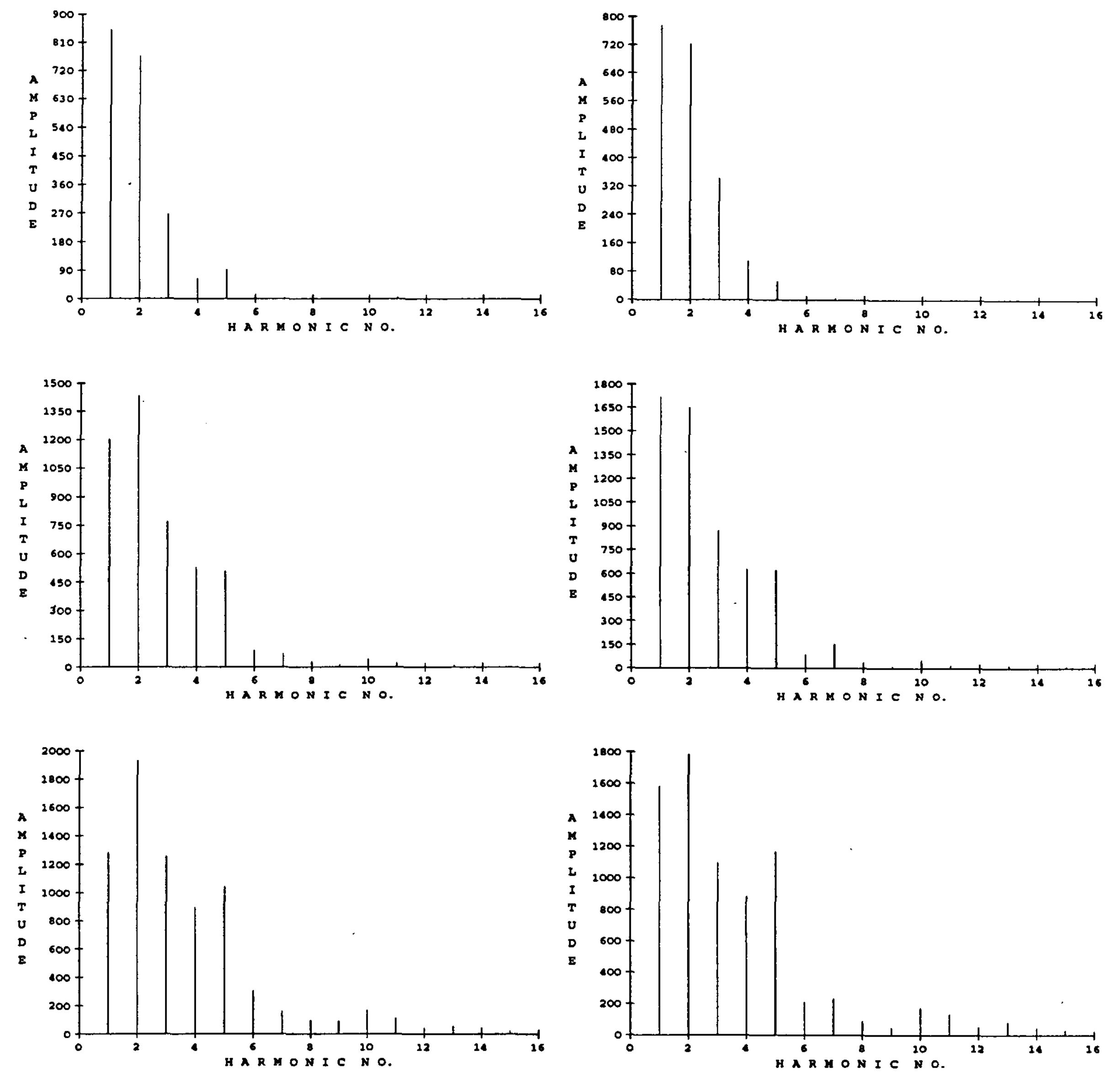


Fig. 2. Pairs of spectra having centroids 500 Hz (top pair from times 0.68 and 4.12), 1000 Hz (middle pair from times 1.29 and 3.24), and 1500 Hz (bottom pair from times 1.65 and 3.04) taken from 588-Hz trumpet tone.

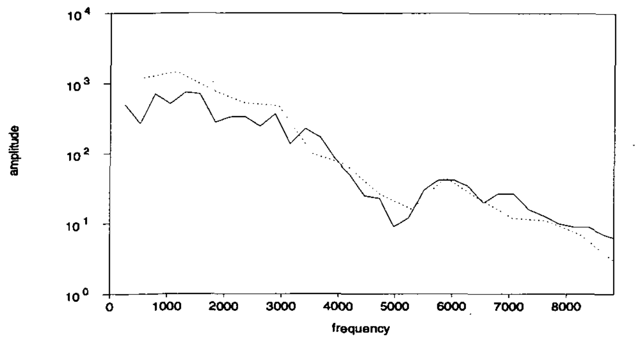


Fig. 3. Equal-centroid spectral envelopes from two different trumpet tones played at 262 and 588 Hz.

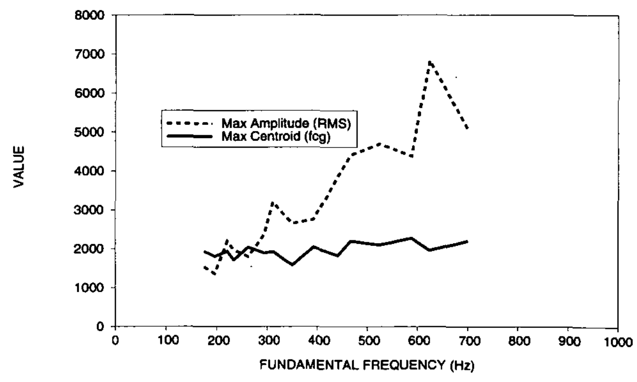


Fig. 4. Peak centroid and rms amplitudes for *ff* trumpet tones at various pitches.

with well-defined formant characteristics like the trumpet often exhibit this property, and we can use this to build an efficient trumpet model.

Fig. 5 shows an overview of the spectral envelope derivation procedure. Some standard preliminary steps precede actual construction of the spectral envelopes. After selecting a group of trumpet tones as the spectral envelope "training set," we use a pitch-synchronous short-time Fourier transform programs to analyze these tones [10]. The training set is comprised of 15 trumpet tones ranging from 175 to 700 Hz, each swelling from *pp* to *ff* and back down again. These tones represent the full pitch and dynamic range of the instrument. The resulting time-varying data contain spectra with centroids ranging from approximately 100 to 2000 Hz.

Construction of the spectral envelopes requires a subdivision of the centroid and frequency ranges of the instrument into discrete subranges. We implement this subdivision by means of a two-dimensional array of centroid versus critical-band frequency "bins."

For the frequency dimension we divide the range of 100 to 11 000 Hz into 23 critical bands as recursively defined by Zwicker and Terhardt [11],

$$f_{n_{\min}} = f_{n-1_{\min}} + 25 + 75(1 + 1.4f^2_{n-1_{\min}})^{0.69} \qquad (3)$$

where $f_{n_{\min}}$ is the minimum frequency of the nth band. Each spectral component of a sound is assigned a critical band according to its frequency. For example, the third harmonic of a 400-Hz fundamental (1200 Hz) would be assigned to critical band 9 because this band ranges from 1059 to 1228 Hz. By using critical bands, we maintain adequate perceptual frequency resolution while significantly reducing the 100 or more frequency bands that would be needed for a sufficient linear resolution.

For the centroid dimension we divide the range of 0 to 2000 Hz into 10 uniformly spaced subranges, each corresponding to one of the spectral envelopes. Each spectral component is assigned to a centroid subrange according to the centroid of the spectrum to which it belongs. For example, a 1200-Hz component of a spectrum whose centroid is 900 Hz would be assigned to the no. 9 critical band and the no. 5 frequency bin.

To calculate the average spectral envelope for each centroid subrange, we first normalize the individual spectra by their peak component amplitudes, so that they all contribute equally to the averages. Then we accumulate the individual spectral component amplitudes into their proper centroid and frequency bins, while incrementing a counter for each bin. The component's frequency and the centroid of the spectrum to which it belongs determine the bin that receives the component's amplitude. Finally, after accumulating all of the normalized component amplitudes (from 15 tones) in the various bins, we compute the average amplitude for each bin by dividing its accumulated value by the value of its bin counter. In our analysis the number of components per bin ranged from 0 to over 50 000 points. The rare occurrences of zero were eliminated by interpolation or extrapolation.

Fig. 6 shows the 10 spectral envelopes we found by this procedure based on our training set of trumpet tones. The spectral envelopes have a relatively small spread up to a peak at about 700 Hz, but beyond that point they exhibit a variable cutoff frequency, with the highest centroid envelope having the highest cutoff. Also, the highest centroid envelopes exhibit a pronounced formant near 6000 Hz. Note that the lowest centroid spectral envelope is based on spectra whose centroids are less than 10% of the peak centroid value. Since dramatic centroid changes occur during the attacks and decays of sounds, the model naturally represents these important transients quite well.

1.2 Verification of Spectral Envelope Model through Additive Synthesis

We can use additive synthesis to verify the spectral envelope model and its assumption that centroid and spectral shape are strong correlates. Fig. 7 shows the main steps of our additive synthesis procedure. After finding the spectral envelopes, we synthesize a particular trumpet tone (not necessarily one from the training set) by using its time-varying spectral centroid, rms amplitude, and fundamental frequency functions (Fig. 8) to control the model. These acoustic-based control functions contribute much of the natural nuance of the original tone to the synthetic sound. Alternatively, and if desired, ad hoc control functions can be substituted to control the model.

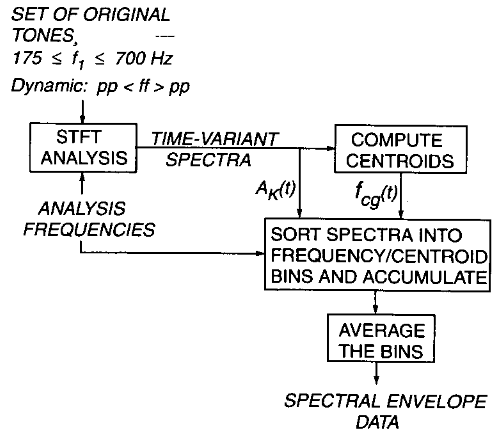


Fig. 5. Overview of spectral envelope computation.

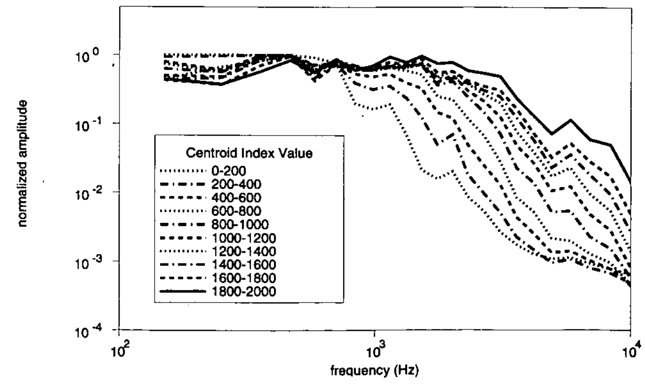


Fig. 6. Dynamic spectral envelopes corresponding to different centroid subranges computed for trumpet.

The time-varying spectral centroid function $f_{cg}(t)$ controls the selection of the appropriate spectral envelope pair at each time point. The selection can be made either by precalibrating the spectral envelopes for the pitch being generated or by just using the "expected centroid value" of each spectral envelope. Precalibration involves sampling all of the spectral envelopes at the synthetic tone's harmonic frequencies and computing the centroids according to Eq. (1). Then the two spectral envelopes whose centroids straddle the centroid function's value are selected. However, the expected centroid values for the spectral envelopes of Fig. 6 are 100, 300, 500, . . . , 1900. Under this assumption, the spectral envelope pair can be chosen using a simple linear formula, a faster method. Next, harmonic samples from the two spectral envelopes chosen are interpolated. Once this synthetic harmonic spectrum has been determined, we measure its rms amplitude and scale the individual components to match the amplitude rms(t) of the control function. Time-varying additive synthesis then proceeds in the usual fashion with the frequency control function $f_1(t)$ used to determine the tone's frequency variation.

To speed up the computation, we can convert the sampled spectra corresponding to the spectral envelopes into fixed wavetables using static additive synthesis. Time-varying additive synthesis is replaced by interpolation between wavetables. As long as the spectrum components have harmonic frequencies, the harmonic phases are consistent, and the same frequency control function is used for each table, multiple wavetable synthesis is perfectly equivalent to additive synthesis [12].

To test the quality of the sounds produced by this method, we have conducted formal listening tests with musically trained listeners. Our results showed that listeners were able to detect differences between original and synthetic tones presented in *AB* fashion (with *AA* as a control) only about half of the time. While there is room for improvement, the sounds were demonstrated to be of high quality.

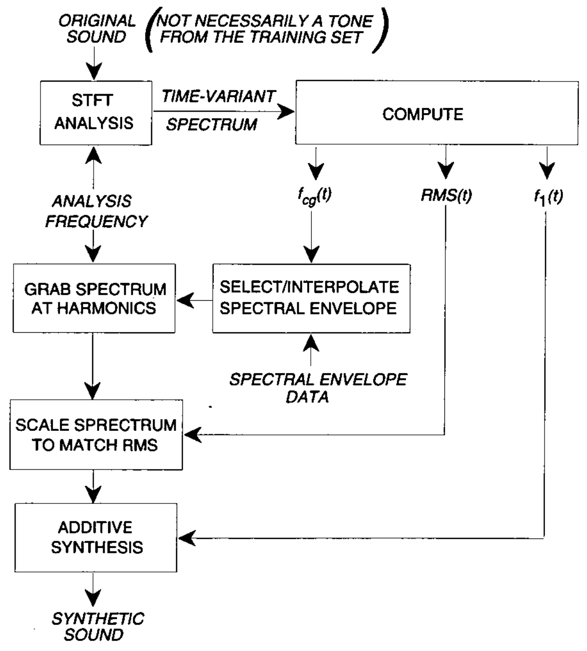


Fig. 7. Dynamic spectral envelope additive synthesis procedure.

2 WAVETABLE AND FILTER MODELING

The additive synthesis spectral envelope model suffers from taking as long to compute as standard additive synthesis. The multiple wavetable equivalent method speeds up the computation but does not work if the fundamental frequency changes by a large amount during a sound, as all of the wavetables should change as the pitch changes. The wavetable and filter model we describe here has neither of these inherent limitations, and it results in greater data reduction.

2.1 Wavetable and Filter Approximation of Spectral Envelopes

After finding the spectral envelopes, we can approximate them with a source and filter model. For a second-order analog low-pass filter, 10 sets of parameters are computed so that the corresponding filter responses approximate the 10 spectral envelopes (shown in Fig. 6) normalized by the highest centroid spectral envelope, as illustrated in Fig. 9. The highest centroid filter response is therefore ideally flat, whereas the other responses should approximate the low-pass shapes of the corresponding modified spectral envelopes. Hence the shapes of the larger centroid normalized envelopes be-

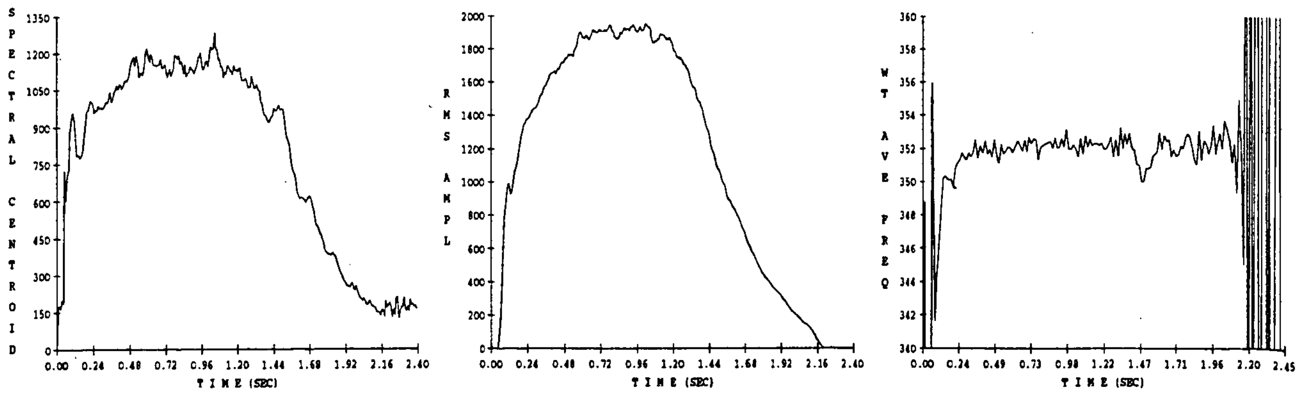


Fig. 8. Centroid, rms amplitude, and fundamental frequency time-varying control functions derived from 350-Hz trumpet tone.

come much simpler than the corresponding nonnormalized envelopes, allowing us to approximate them with a simple second-order filter structure. (We would need a much higher order structure if we tried to approximate the nonnormalized spectral envelopes, where the source would be a pulse waveform with a flat spectrum, to achieve the same quality of approximation.) The result is a centroid-programmable low-pass filter, which needs to be computed only once. So for the synthesis model we only need to store the filter parameters corresponding to the 10 centroid values.

Computation of the parameters of an analog (that is, sample-rate independent) filter begins with a basic second-order analog filter response function,

$$\Re(s) = \frac{G}{(1 + 2\zeta s/f_r + (s/f_r)^2} \tag{4a}$$

where G is the low-frequency gain, f_r is the undamped resonance frequency, and ζ is the damping factor. ζ must be greater than zero for the filter to be realizable and greater than $1\sqrt{2} \cong 0.7071$ if one wishes to avoid a response maximum.

The actual filter (magnitude) frequency response is calculated by substituting $s \leftarrow jf$ and taking the complex magnitude,

$$R(f) = \frac{G}{\cdot\sqrt{1 + 2(2\zeta^2 - 1)(f/f_r)^2 + (f/f_r)^4}} \tag{4b}$$

which can be rewritten as

$$R(f) = \frac{1}{\sqrt{b_0 + b_1 f^2 + b_2 f^4}} . \tag{4c}$$

In order to cover a broad frequency range we attempted to match the filter at three points of its response, namely, its b_0 value, which is inversely related to the low-frequency gain, its cutoff frequency f_c, where $R(f_c) = 1/\sqrt{2} \cong 0.7071$, and its threshold frequency f_t, where $R(f_c) = 0.1$. We found these by a quick enumerative search. The center frequencies of all of the 23 critical bands were tested as possible values of f_c and f_t, with the restriction that the threshold frequency be higher than the cutoff frequency. In addition, for each f_c and f_t combination, we tested all values of b_0 between 0 and 1 in steps of .01. Therefore we tested $100 \times 23 \times 23/2 = 26\,450$ combinations of b_0, f_c, and f_t and kept the set that gave the best match to a particular spectral envelope. For each combination, b_1 and b_2 were calculated using

$$b_2 = \frac{(100 - b_0)f_c^2 - (2 - b_0)f_t^2}{f_t^2 f_c^2 (f_t^2 - f_c^2)} \tag{5a}$$

and

$$b_1 = \frac{2 - b_0 - b_2 f_c^4}{f_c^2} \tag{5b}$$

and then $R(f)$ was evaluated at a number of frequencies to determine the best match to the original data.

The best match was determined by finding b_0, f_c, and f_t, which minimized the fitness function

$$\text{fitness}_m = \sum_{n=1}^{23} w(A_m(f_n)) \frac{|A_M(f_n)\,R(f_n) - A_m(f_n)|}{A_m(f_n)} \tag{6a}$$

where $A_m(f_n)$ is the amplitude of the mth spectral envelope, $A_M(f_n)$ is the amplitude of the highest centroid spectral envelope, and $R(f_n)$ is the filter response, each sampled at the center of the nth critical band given by $f_n = 0.5(f_{n_{\min}} + f_{n+1_{\min}})$, $f_{n_{\min}}$ being the band minimum frequency, given recursively by Eq. (3), and w an empirically derived weight function, given by

$$w(A_m(f_n)) = \frac{18}{17 + [1 - 2\log_2(A_m(f_n))]^2} . \tag{6b}$$

Basically, we designed Eq. (6a) to minimize the sum of the absolute relative errors in the output spectrum at each of the critical-band frequencies. In addition, we designed Eq. (6b) in an effort to achieve more psychoacoustically optimum fits between the original spectral envelopes and those generated by the wavetable–filter combination. Note that $0 < A_m(f_n) \leqslant 1.0$ so that $w(A_m(f_n))$ takes on small values when $A_m(f_n)$ is small; however, when $A_m(f_n) = 1$, $wA_m(f_n)) = 1$.

Further, to satisfy the realizability criterion and avoid a response maximum as mentioned, we needed b_0, b_1, and b_2 to be positive.

Once the filter coefficients b_0, b_1, and b_2 were computed, we verified the filter responses for each centroid subrange by plotting Eq. (4c). Fig. 10 shows the 10 filter responses that were matched to the modified spectral envelopes of Fig. 9 using this method. The accuracy of matching these spectral envelopes is limited because of the inherent response of a second-order low-pass filter, which decreases monotonically above its cutoff frequency. For example, we could not expect to match the increasing regions near 10 000 Hz on the low-centroid curves. However, the slopes of the filter curves match

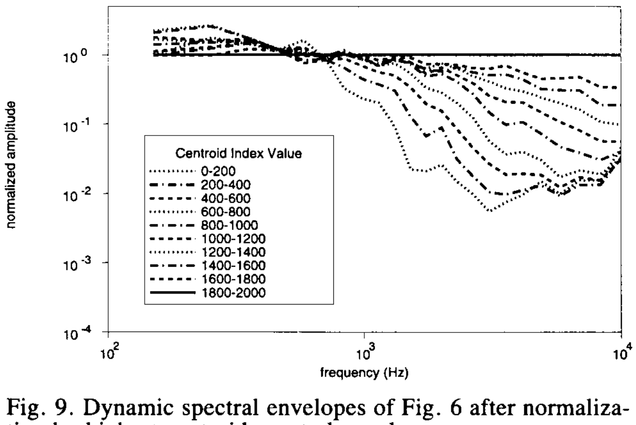


Fig. 9. Dynamic spectral envelopes of Fig. 6 after normalization by highest centroid spectral envelope.

respectably for frequencies in the vicinity of 1000 Hz.

While the 10 low-pass filter responses are fixed for synthesis, the source waveform varies with pitch. For a particular synthesis frequency, the source signal is provided by a fixed wavetable whose waveform is generated by additive synthesis of harmonics sampled from the highest centroid spectral envelope. Fig. 11(a) shows the spectral envelope from which source spectra are sampled. Theoretically a different spectrum and equivalent waveform would be needed for each pitch. Fig. 11(b) shows a typical source spectrum extracted from this envelope. (Another possibility, which we have not tried yet, would be to convert the highest centroid spectral envelope into an impulse response by taking its inverse Fourier transform and then to use the overlapped impulse-response synthesis method [8]. Although this method would increase the synthesis computation, its advantage is that it would automatically handle large changes of pitch without requiring changes of the source waveform.) Multiplication of the source spectrum (for a particular pitch) with the filter response (for a particular centroid control function), together with rms amplitude matching, should closely match the spectra obtained by the spectral envelope additive synthesis method described in Section 1.2.

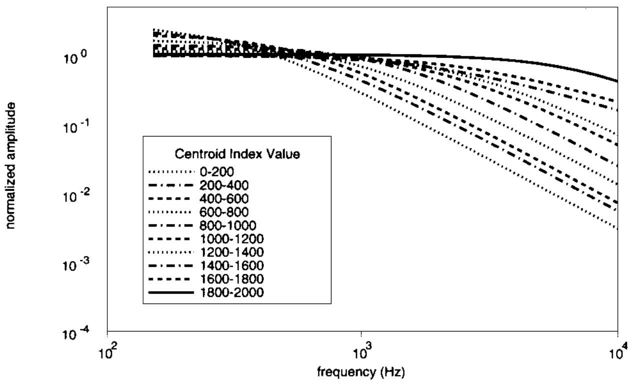
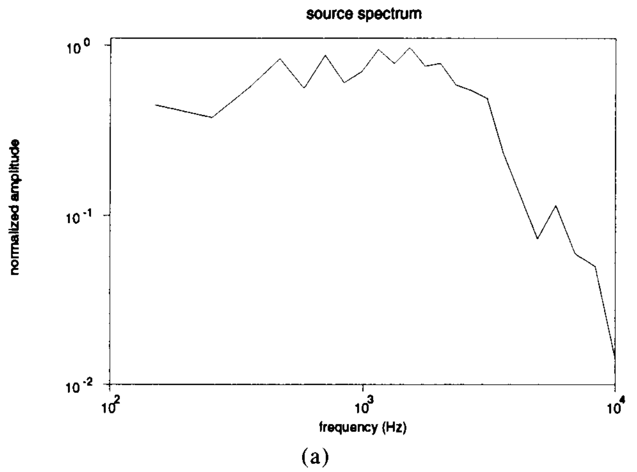
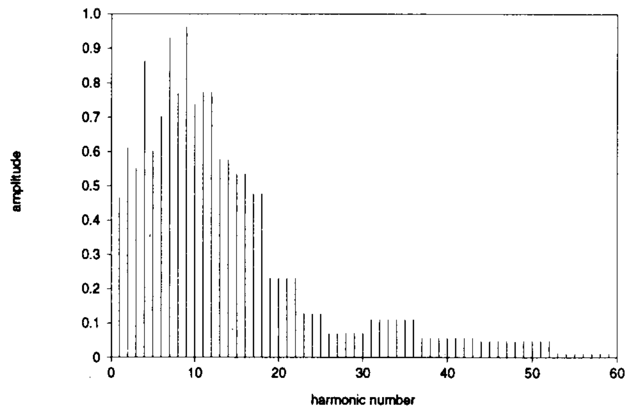


Fig. 10. Low-pass filter responses designed to match modified spectral envelopes of Fig. 9.



(a)



(b)

Fig. 11. (a) Spectral envelope used to derive source spectra for wavetable waveform. (b) Typical source spectrum extracted from source spectral envelope.

Note that the source spectral envelope includes the trumpet's formant at 6000 Hz as well as other interesting spectral features. Therefore the variable filter only needs to provide smooth approximations to the rolloffs of the spectral envelopes, while the source spectrum picks up the individual spectral nuances common to most of the spectral envelopes. The lowest centroid responses are the most poorly matched, but since the errors occur in very low-amplitude spectra, they are the least likely to have any perceptual impact.

Fig. 12 shows the final spectral envelope family which effectively results from wavetable and filter synthesis of the trumpet. It is obtained by multiplying the source spectral envelope of Fig. 11(a) by the low-pass family of Fig. 10. Fig. 13 summarizes the sequence of steps used for wavetable and filter approximation of the dynamic spectral envelopes.

2.2 Wavetable and Filter Synthesis

Fig. 14 outlines the wavetable and filter synthesis procedure. Synthesis uses the same centroid, rms, and fundamental frequency control functions as is used with the additive synthesis model. Here the frequency control function gives the wavetable's sampling increment, whereas the centroid control function translates into an appropriate set of second-order digital filter coefficients (which are interpolated for accuracy), and the rms function again controls the amplitude scaling of the output signal. We use the centroid, rms, and frequency functions of the tone we wish to match (which may or may

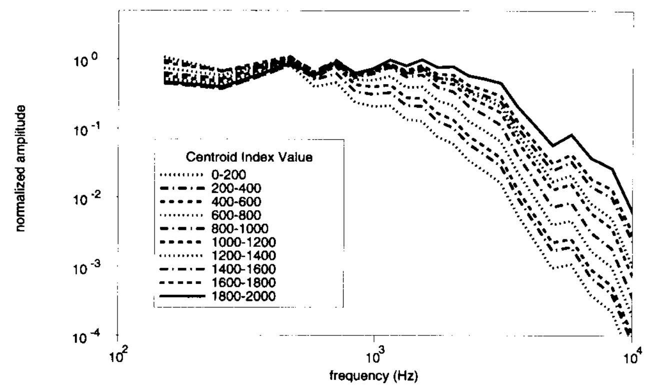


Fig. 12. Final spectral envelope family, an approximation of family of Fig. 6.

not be one from the original training set) and normalize the output of the wavetable and filter so that it matches the rms function exactly.

Central to the synthesis algorithm is the digital low-pass filter, which we have implemented as a recursion formula for a second-order IFR filter,

$$y_n = c_0x_n + c_1x_{n-1} + c_2x_{n-2} - d_1y_{n-1} - d_2y_{n-1} \,. \tag{7}$$

The coefficients c_0, c_1, c_2, d_1, and d_2 must be chosen so that the frequency response of this filter closely matches the analog filter characteristic $R(f)$ given by Eq. (4c). The coefficients depend on the analog filter parameters b_0, b_1, and b_2 as well as on the sample frequency f_s. Fortunately it is easy to convert from an analog filter definition to a digital form using the standard bilinear transform [13]. To accomplish this, we convert $R(f)$ back to $\Re(s)$ to get

$$\Re(s) = \frac{1}{\sqrt{b_0} + \sqrt{b_1 + 2\sqrt{b_0b_2}\,s} + \sqrt{b_2}\,s^2} \,. \tag{8}$$

Then we make the substitution

$$s \leftarrow f_x \cot\left(\frac{\pi f_x}{f_s}\right) \frac{1 - z^{-1}}{1 + z^{-1}} \tag{9}$$

where f_x is some fixed frequency corresponding to the middle of the trumpet's range (we use Bb$_4$ = 466 Hz), f_s is the sample frequency, and z^{-1} is the z transform unit delay operator. This transformation has the virtue of preserving the response for $f = 0$ and $f = f_x$, whereas the response at $f = \infty$ is mapped into $f = f_s/2$ (the half sample frequency). This leads to the IIR digital filter recursion formula

$$y_n = \frac{1}{d_0}(x_n + 2x_{x_{n-1}} + x_{n-2}) - d_1y_{n-1} - d_2y_{n-1} \tag{10a}$$

with coefficients defined as

$$p = f_x \cot\left(\frac{\pi f_x}{f_s}\right) \tag{10b}$$

$$v_0 = \sqrt{(b_0)}, \quad v_1 = \sqrt{b_1 + 2\sqrt{b_0b_2}\,p}, \quad v_2 = \sqrt{b_2}\,p^2 \tag{10c}$$

$$d_0 = v_0 + v_1 + v_2 \tag{10d}$$

$$d_1 = \frac{2(v_0 - v_2)}{d_0} \tag{10e}$$

$$d_2 = \frac{v_0 - v_1 + v_2}{d_0} \,. \tag{10f}$$

Since the output is scaled to achieve the proper rms amplitude anyway, there really is no need for the $1/d_0$ multiplier in Eq. (10a). Actually, we store 10 sets of the digital filter coefficients as defined by Eq. (10b–f), and the exact coefficient values used for each synthesis frame are determined by interpolation between the stored values according to the centroid control function.

To match the rms amplitude and centroid control functions on each frame we need to be able to predict these values as they would be produced by the low-pass filter at its various settings. This requires an evaluation of the frequency response for the pitch of the particular tone being synthesized. Given the digital filter coefficients,

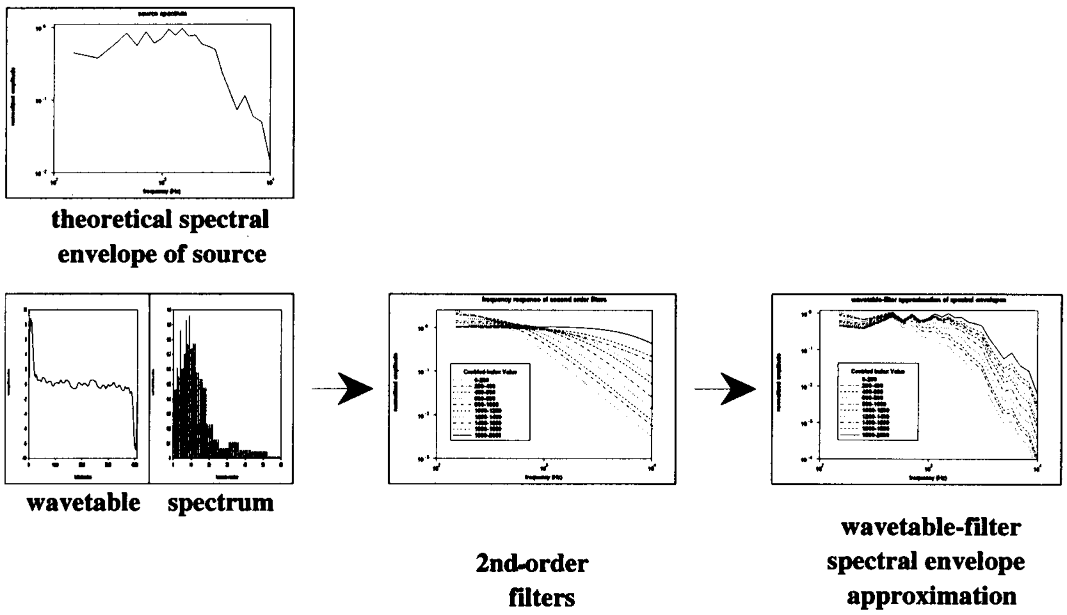


Fig. 13. Wavetable and filter approximation of spectral envelopes. Example source wavetable uses 175-Hz fundamental frequency.

we compute this response using

$$R'(f) = \frac{1}{d_0} \frac{\sqrt{[1 + 2\cos(\omega) + \cos(2\omega)]^2 + [2\sin(\omega) + \sin(2\omega)]^2}}{\sqrt{[1 + d_1\cos(\omega) + d_2\cos(2\omega)]^2 + [d_1\sin(\omega) + d_2\sin(2\omega)]^2}} \tag{11}$$

where $\omega = 2\pi f/f_s$. This response is then sampled at the harmonics of the fundamental frequency used for synthesis to give the synthetic harmonic amplitudes

$$a'_k = A_M(kf_a)\,R'(kf_a)\,. \tag{12}$$

The synthetic centroid and rms amplitude outputs are then computed using Eqs. (1) and (2). Note again that the response $R'(f)$ varies with the centroid value because the digital filter coefficients vary according to the centroid. Thus the synthetic f'_{cg} and rms′ are computed for each spectral envelope. Then to match a desired centroid value, it is a matter of finding the two spectral envelopes whose centroids straddle this value and then interpolating the filter coefficients accordingly. Alternatively we can use the "expected values" of the filter's centroids, as discussed in Section 1.2. To match the rms amplitude, we simply compute an amplitude multiplier for the output of the filter,

$$\text{ampscale}(t) = \frac{\text{rms}(t)}{\text{rms}'(t)}\,. \tag{13}$$

However, the time-varying filter coefficients and ampscale only need to be computed once per frame (control rate), which occurs at a considerably slower rate than the sample rate. For accuracy, these parameters can be interpolated linearly for each sample output.

The total computation amounts to a small number of memory accesses and arithmetic operations per sample, giving about a factor-of-10 improvement in computational efficiency over additive synthesis. Fig. 15 summarizes the complete modeling, approximation, and synthesis procedure.

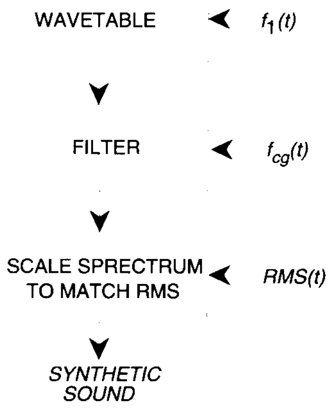


Fig. 14. Wavetable and filter synthesis procedure.

2.3 Additional Considerations for Practical Implementation

One especially useful feature of the wavetable and filter approximation is that it can use the same set of filters for all fundamental frequencies. What about the wavetable? Do we need a different wavetable for each fundamental frequency since we sample the spectral envelopes at harmonics of this frequency to construct the source wavetable? Theoretically, the answer is "yes." However, do sampling synthesizers use a different sample for each fundamental frequency of their sampled instrument sounds? No, they use two to four samples for each octave and interpolate the notes in between. The same applies here: we can use five wavetables at fundamental frequencies Bb$_3$, F$_4$, Bb$_4$, F$_5$, and Bb$_5$ to represent the full range of the trumpet and interpolate the notes in between. The wavetable and filter Csound

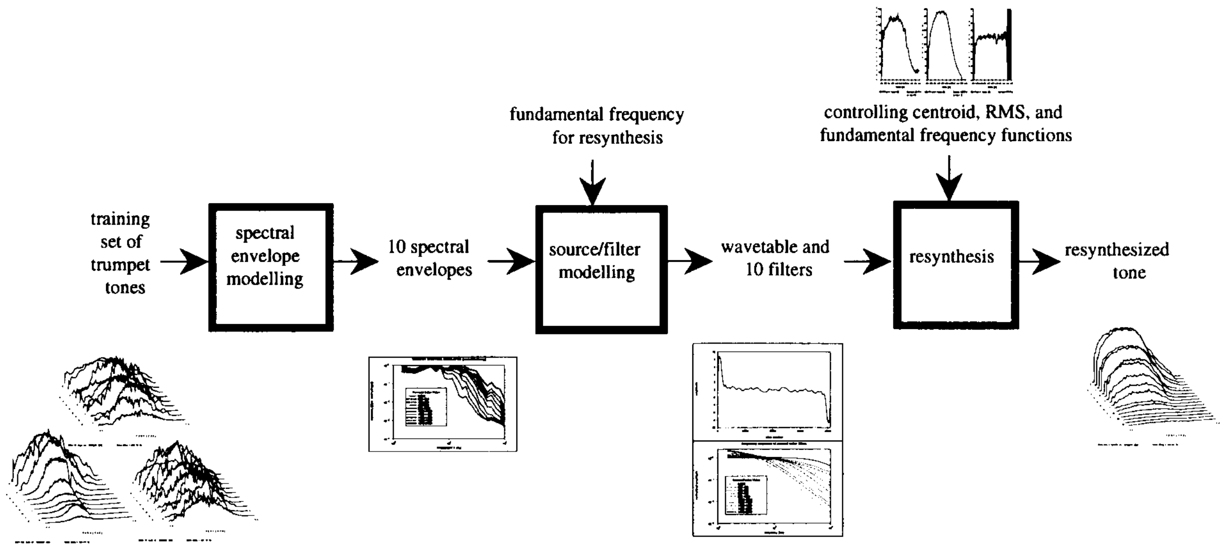

fundamental frequency for resynthesis
controlling centroid, RMS, and fundamental frequency functions
training set of trumpet tones
spectral envelope modelling
10 spectral envelopes
source/filter modelling
wavetable and 10 filters
resynthesis
resynthesized tone

Fig. 15. Overview of wavetable and filter spectral envelope modeling and synthesis.

instrument given in the Appendix uses this technique.

Centroid, rms amplitude, and frequency time-varying functions measured from acoustic sounds are quite complex. The microvariations and the variety of shapes of these parameters contribute much to the liveliness of the resulting synthetic sounds. However, for a practical synthesis instrument, we needed to construct prototype control function shapes based on piecewise linear approximations. Fig. 16 shows graphs of the three control functions used for the Csound instrument, comparable to those of Fig. 8. Microvariations are restored by the addition of low-frequency noise with a bandwidth of 7 Hz.

3 RESULTS

Figs. 17 and 18 show the time-varying spectra for two midrange trumpet tones, along with the corresponding spectra matched with wavetable and filter synthesis. The wavetable and filter synthetic tones definitely sound like trumpets, though they lack some of the detailed characteristics of the original tones. The low-register synthetic tones (such as in Fig. 17) often sound "stronger" than the corresponding original tones because they do not have the spectral irregularities of the originals. On the other hand, the time-variation nuances of the controlling frequency, rms, and spectral centroid functions do introduce cues that the synthetic sounds are "live," even though some of the spectral details have been lost.

It is not necessary that the three control functions be taken from tones used in the analysis training set. The nontraining-set match shown in Fig. 19 also looks and sounds good, proving the effectiveness of the wavetable and filter model at capturing the essence of the trumpet's spectral behavior.

In order to compare the accuracy of the more general dynamic spectral envelope synthesis method with that of the wavetable and filter method, we can measure each of their "relative errors" with respect to the original time-varying spectra. The relative error can be measured as a function of time and is given by

$$\varepsilon(t) = \sqrt{\left\{\frac{\sum_{k=1}^{N_{\text{har}}} [a_k(t) - a'_k(t)]^2}{\sum_{k=1}^{N_{\text{har}}} a_k^2(t)}\right\}} \tag{14}$$

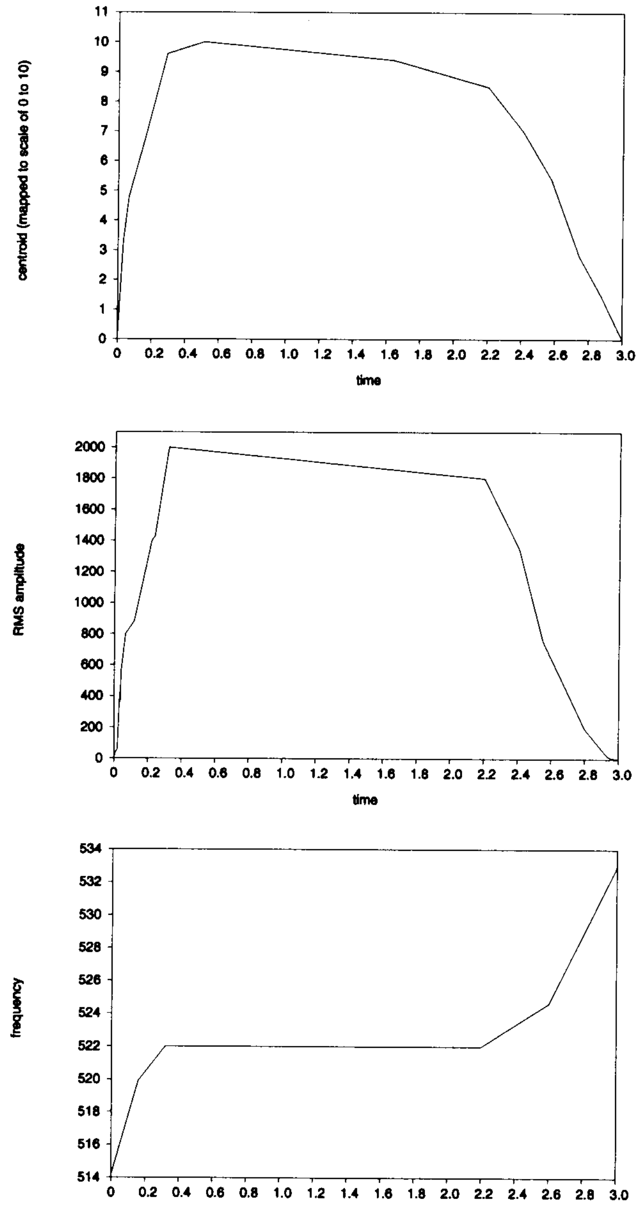


Fig. 16. Centroid (mapped linearly to a scale of 0:10), rms amplitude, and fundamental frequency time-varying control functions for practical synthetic instrument.

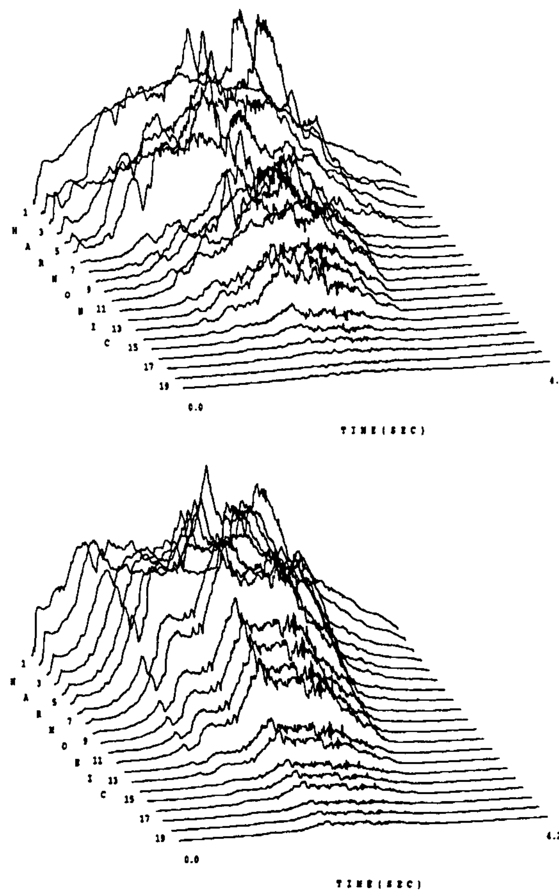


Fig. 17. Time-varying spectra for original 262-Hz trumpet tone and its wavetable and filter synthesis match. Original tone was in spectral envelope training set.

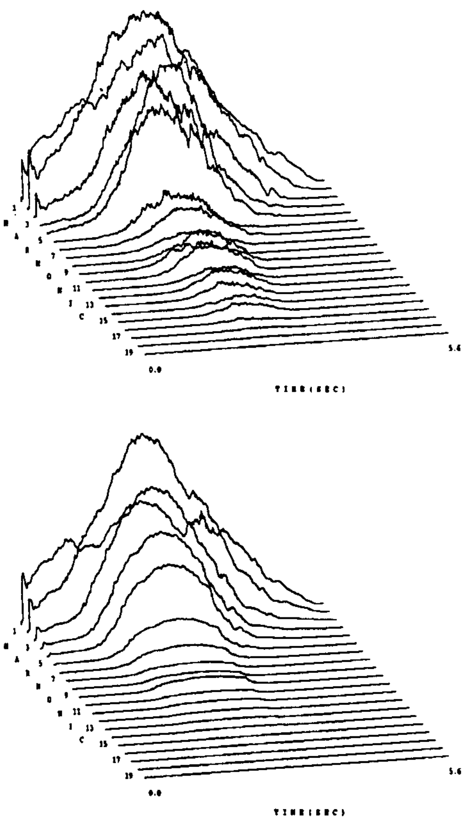


Fig. 18. Time-varying spectra for original 588-Hz trumpet tone and its wavetable and filter synthesis match. Original tone was in spectral envelope training set.

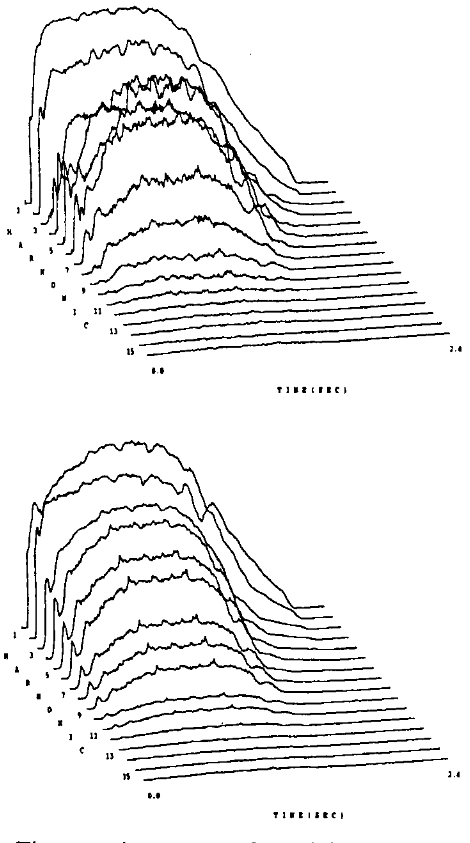


Fig. 19. Time-varying spectra for original 350-Hz trumpet tone and its wavetable and filter synthesis match. Original tone was not in spectral envelope training set.

where $a_k(t)$ is the original kth harmonic at time t and $a_k(t)$ is the corresponding data-reduced harmonic amplitude. Fig. 20 shows plots of relative error versus time for the 262-, 588-, and 350-Hz trumpet tones. Each plot compares the error for the spectral envelope additive synthesis method, using the spectral envelopes of Fig. 6, with that of the wavetable and filter method, which is based on the spectral envelopes of Fig. 12. It is interesting to note that on average the errors for the wavetable and filter synthesis case are only slightly higher than those for the dynamic spectral envelope synthesis case, indicating that we have almost as good approximations to the spectral envelopes. Sometimes the wavetable and filter method has an even smaller error than does the dynamic spectral envelope method (on which it is based). Therefore we do not lose much accuracy with the wavetable and filter approximation while gaining considerably in computational efficiency.

4 CONCLUSIONS

Spectral envelopes provide a good representation of the spectral and dynamic behavior of the trumpet, and

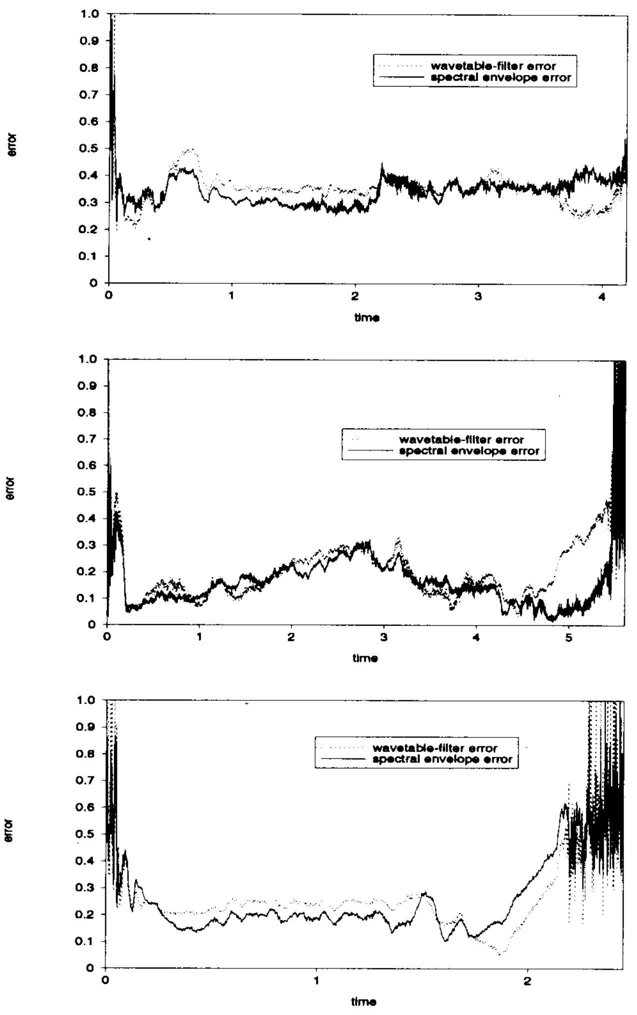


Fig. 20. Relative error versus time for two tones in training set [262 Hz (top), and 588 Hz (middle)], and one tone not in training set [350 Hz (bottom)].

the wavetable and filter approximation is an efficient means of using this representation. The wavetable and filter model produces high-quality synthesized trumpet tones with only three time-varying control functions —spectral centroid, rms amplitude, and fundamental frequency. The Appendix gives a Csound implementation of the wavetable and filter instrument design. The wavetable and filter model should also work for other acoustic instruments that have well-correlated centroids and spectral shapes, especially the other brass and wind musical instruments.

APPENDIX
Csound IMPLEMENTATION OF WAVETABLE AND FILTER SYNTHESIS

A1 Csound Orchestra for Trumpet Synthesis

```
sr = 22050
kr = 2205
ksmps = 10
nchnls = 1

giseed = 5.5
;*********************************************
instr    1

          ipchfreq = p4
          iampl = p5
          iamp2 = p6
          imaxbr = p7

          ifreq = cpspch(ipchfreq)
          iattack = 0.24*(1.1667 - .1667*ifreq/261)
; don't allow attack to be more than 30% of tone
          if iattack< 0.30*p3 igoto initl
              iattack = 0.30*p3
initl:      idecay = 0.8
; don't allow decay to be more than 65% of tone
          if idecay< 0.65*p3 igoto init2
              idecay = 0.65*p3
init2:      isustain = p3 – iattack – decay
          if isustain ! = 0 igoto init3
              isustain = .1*idecay
              idecay = .9*idecay
init3:
          kpct line 0.0, p3, 1.0
          ktime = kpct*p3
          kpt = kpct*100

; prototype dfr curve (from ctpt03.pv.an)
          kfreq linseg – .015, iattack/2, – .004,
          iattack/2, .0, isustain, .0, idecay/2, .005,
idecay/2, .021
          kfreqrl randi. 0004, 10, giseed
; 0.4% nuances
          giseed = frac(giseed + 3.141592654)
          kfreqr2 rand .002, giseed
; 0.2% noise
          giseed = frac(giseed + 3.141592654)
          kfreq = (kfreq + kfreqrl + kfreqr2)*ifreq
          display kfreq, p3

; prototype rms curve (from ctpt03.pv.an)
          krms linseg0, .031*iattack, .021*iampl,
.031*iattack, .028*iampl, .041*iattack, .21*iampl,
.01*iattack, .18*iampl, .017*iattack, .28*iampl,
.08*iattack, .4*iampl, .16*iattack, .44*iampl,
.32*iattack, .7*iampl, .06*iattack, .714*iampl,
.25*iattack, iampl, isustain, iamp2, .26*idecay,
.075*iamp2, .18*idecay, .42*iamp2, .31*idecay,
.109*iamp2, .17*idecay, .0109*iamp2, .015*idecay,
.007*iamp2, .01*idecay, .0047*iamp2, .02*idecay,
.0028*iamp2, .03*idecay, .0019*iamp2, .005*idecay,
0
          krmsr randi .05*krms, 20, giseed
; 5% nuances
          giseed = frac(giseed + 3.141592654)
          krms = abs(rksm + krmsr)
          display krms, p3

; prototype pseudo-brightness curve
          kbr linseg 0, .01*iattack, .33*imaxbr,
.1*iattack, .48*imaxbr, .3*iattack,
.68*imaxbr, .4*iattack, .96*imaxbr, .1*attack +
.1*isustain, 1.0*imaxbr, .6*isustain, .94*imaxbr,
.3*isustain, .85*imaxbr, .26*idecay, .70*imaxbr,
.21*idecay, .54*imaxbr, .21*idecay, .28*imaxbr,
.16*idecay, .15*imaxbr, .16*idecay, 0

          kbrrl randi 0.07*kbr, 20, giseed
; 7% nuances
          giseed = frac(giseed + 3.141592654)
;         kbrr2 rand 0.02*kbr, giseed
; 2% noise
kbrr2 = 0
          giseed = frac(giseed + 3.141592654)
          kbr = as(kbr + kbrrl + kbrr2)
          display kbr, p3
w1:     if ipchfreq>8.01 goto w2
            iwt = 1
            goto wend
w2:     if pichfreq> 8.0 goto w3
            iwt = 2
            goto wend
w3:     if ipchfreq > 9.01 goto w4
            iwt = 3
            goto wend
w4:     if ipchfreq > 10.00 goto w5
            iwt = 4
            goto wend
w5:         iwt = 5
wend:   awt oscili krms, ifreq + kfreq, iwt, giseed
        giseed = frac(giseed + 3.141592654)

; set filter parameters
        icut0 = 20
        icutl = 300
        icut2 = 500
        icut3 = 825
        icut4 = 975
        icut5 = 1225
        icut6 = 1525
        icut7 = 1925
        icut8 = 2450
        icut9 = 3000
        icut10 = 4000

        kw2 = frac(kbr)
        kw1 = 1.0-kw2
f1:     if kbr >= 1 kgoto f2
            kcut = kw1*icut0 + kw2*icut1
            kgoto fend
f2:     if kbr >= 2 kgoto f3
            kcut = kw1*icut1 + kw2*icut2
            kgoto fend
f3:     if kbr >= 3 kgoto f4
            kcut = kw1*icut2 + kw2*icut3
            kgoto fend
f4:     if kbr >= 4 kgoto f5
            kcut = kw1*icut3 + kw2*icut4
            kgoto fend
f5:     if kbr >= 5 kgoto f6
            kcut = kw1*icut4 + kw2*icut5
            kgoto fend
f6:     if kbr >= 6 kgoto f7
            kcut = kw1*icut5 + kw2*icut6
            kgoto fend
f7:     if kbr >= 7 kgoto f8
            kcut = kw1*icut6 + kw2*icut7
            kgoto fend
f8:     if kbr >= 8 kgoto f9
            kcut = kw1*icut7 + kw2*icut8
            kgoto fend
f9:     if kbr >= 9 kgoto f10
            kcut = kw1*icut8 + kw2*icut9
            kgoto fend
f10:     if kbr > = 10 kgoto fmax
            kcut = kw1*icut9 + kw2*icut10
            kgoto fend
fmax:    kcut = icut10
fend:
         display kcut, p3
         afilt reson awt, 0, sqrt(2.)*kcut, 0
         asig balance afilt, awt
         kgoto end
end:
; .1% noise in signal
         anoise rand 0.001*asig, giseed
         giseed = frac(giseed+3.141592654)
         asig = asig+anoise
         out asig
         endin
```

A2 Csound Score for Trumpet

```
f1 0 8193 10 0.376492 -0.836273 0.875826
-0.702022 -0.944316 0.782612 0.755986 0.755986
0.785038 0.588679 -0.544976 0.544976 0.489619
0.489619 -0.233817 -0.233817 0.132038
-0.132038 0.132038 -0.072570 0.072570 0.072570
0.114808 0.114808 -0.114808 -0.114808 0.059510
0.059510 0.059510 0.059510 -0.059510 0.059510
0.049870 0.049870 0.049870 -0.049870 0.049870
-0.049870 0.049870 0.013012 0.013012 0.013012

f2 0 8193 10 0.574816 -0.875826 -0.702022
0.782612 -0.755986 -0.785038 -0.588679
-0.544976 0.489619 -0.233817 0.233817
-0.132038 -0.072570 0.072570 -0.072570
-0.114808 -0.114808 -0.114808 0.059510
-0.059510 -0.059510 0.049870 0.049870 0.049870
-0.049870 -0.049870 0.013012 0.013012 0.013012

f3 0 8193 10 0.836273 0.702022 0.782612 -0.785038
0.588679 0.544976 -0.489619 0.233817 0.132038
0.072570 -0.072570 -0.114808 -0.114808
0.059510 -0.059510 0.049870 0.049870 -0.049870
-0.049870 -0.013012 -0.013012

f4 0 8193 10 0.875826 0.782612 -0.785038 0.544976
-0.233817 -0.132038 -0.072570 -0.114808
-0.114808 0.059510

f5 0 8193 10 0.875826 0.782612 -0.544976 0.233817
-0.132038 -0.072570 -0.022570

i1 0 3.0 8.01 2000 1800 9
i1 2 3.0 7.08 2000 1800 9
i1 2 3.0 9.01 2000 1800 9
i1 4 3.0 7.05 2000 1800 9
i1 4 3.0 8.08 2000 1800 9
i1 4 3.0 9.00 2000 1800 9
e
```

5 ACKNOWLEDGMENT

This work was supported in part by the Research Board of the University of Illinois at Urbana-Champaign and by the Hong Kong Research Grant Council's projects DAG93/94.EG02 and HKUST597/94E. Special thanks are due to Lydia Ayers for her comments and suggestions, especially on the Csound instrument in the Appendix.

6 REFERENCES

[1] J. Beauchamp, "A Computer System for Time-Variant Harmonic Analysis and Synthesis of Musical Tones," in H. von Foerster and J. Beauchamp (Eds.), *Music by Computers*. (Wiley, New York, 1969).

[2] J. Risset and M. Mathews, "Analysis of Musical Instrument Tones," *Physics Today*, vol. 22, no. 2, pp. 23–30 (1969).

[3] J. Grey and J. Moorer, "Perceptual Evaluation of Synthesized Musical Instrument Tones," *J. Acoust. Soc. Am.*, vol. 62, pp. 454–462 (1977).

[4] J. W. Beauchamp, "Synthesis by Spectral Amplitude and 'Brightness' Matching of Analyzed Musical Instrument Tones," *J. Audio Eng. Soc.*, vol. 30, pp. 396–406 (1982 June).

[5] J. Grey and J. Gordon, "Perceptual Effects of Spectral Modifications on Musical Timbres," *J. Acoust. Soc. Am.*, vol. 63, pp. 1493–1500 (1978).

[6] D. Luce and M. Clark, "Physical Correlates of Brass-Instrument Tones," *J. Acoust. Soc. Am.*, vol. 42,

pp. 1232–1243 (1967).

[7] F. Slaymaker, "Synthesizing Musical Tones from an Impulse Response Stored in a Digital Memory," *J. Acoust. Soc. Am.*, vol. 67, pp. 713–715 (1980).

[8] F. Slaymaker, "Synthesizing Musical Tones by Summing Successive Read-outs of a Stored Impulse Response," *J. Acoust. Soc. Am.*, vol. 95, pt. 2, p. 2889 (1994).

[9] W. Strong and M. Clark, "Synthesis of Wind-Instrument Tones," *J. Acoust. Soc. Am.*, vol. 41, pp. 39–52 (1966).

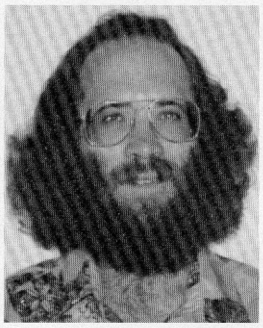
[10] J. W. Beauchamp, "Unix Workstation Software for Analysis, Graphics, Modification, and Synthesis of Musical Sounds," presented at the 94th Convention of the Audio Engineering Society, *J. Audio Eng. Soc. (Abstracts)*, vol. 41, p. 387 (1993 May), preprint 3479.

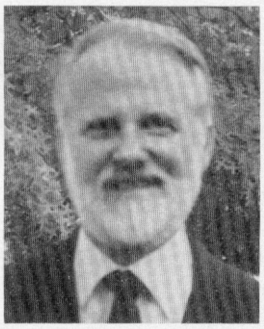
[11] E. Zwicker and E. Terhardt, "Analytical Expressions for Critical-Band Rate and Critical Bandwidth as a Function of Frequency," *J. Acoust. Soc. Am.*, vol. 68, pp. 1523–1525 (1980).

[12] A. Horner and J. Beauchamp, "Piecewise Linear Approximation of Additive Synthesis Envelopes: A Comparison of Various Methods," to be published in *Computer Music Journal* (1995).

[13] J. Kaiser and R. Golden, "Design of Wideband Sampled-Data Filters," *Bell Sys. Tech. J.*, vol. 43, pp. 1533–1546 (1964).

THE AUTHORS


A. Horner


J. Beauchamp

Andrew Horner is an assistant professor in the Computer Science Department at the Hong Kong University of Science and Technology. He is also a warden of the Graphics and Music Experimentation (GAME) Room, and cochair of the upcoming 1996 International Computer Music Conference. He received his Ph.D. in computer science from the University of Illinois at Urbana-Champaign, where he worked in the Computer Music Project, the CERL Sound Group, the Center for Complex Systems, and the Illinois Genetic Algorithm Laboratory. He received his Master's degree in computer science from the University of Tennessee, Knoxville, and his Bachelor's degree in Music from Boston University. Dr. Horner's primary research interests are in applying artificial evolutionary techniques to sound computation and computer-assisted composition.

●

James Beauchamp was born in Detroit in 1937. He received B.S. and M.S. degrees in electrical engineering at University of Michigan during 1960–61 and a Ph.D. in electrical engineering at the University of Illinois at Urbana-Champaign (UIUC) in 1965, studying with Lejaren Hiller Jr. In 1965 he joined the Department of Electrical Engineering at UIUC and in 1969 took a joint appointment with that department and the School of Music. He currently holds the position of professor in these two departments.

In the 1960s and 1970s he developed special analog synthesizer equipment for use by composers at UIUC. Meanwhile, he pioneered work on sound analysis/synthesis using digital computers. This has culminated in his development of two public domain software packages, SNDAN, for spectral analysis, display, modification, and synthesis of musical sounds, and Music 4C, for score-based music synthesis. During 1968–69 he was a research associate at Stanford University's Artificial Intelligence Project working on problems in speech recognition. In 1988 he was a Visiting Scholar at Stanford's Center for Computer Research in Music and Acoustics (CCRMA). During 1994–95 he was a visiting researcher at Institut de Recherche et Coordination Acoustique/Musique (IRCAM) in Paris, France, where he worked primarily on the perceptual effects of data reduction on the timbre of orchestral instrument sounds with Stephen McAdams.

Dr. Beauchamp teaches courses at UIUC in musical acoustics, electronic music technology, audio, and computer music in both the School of Music and the Department of Electrical and Computer Engineering. He has been director of the UIUC Experimental Music Studios (1969–73) and the Computer Music Project (1984–93). He is a member of the Acoustical Society of America (ASA), a fellow of the Audio Engineering Society, and member of the International Computer Music Association (ICMA) and the Society for Electro-Acoustic Music in the U.S. (SEAMUS). He was president of ICMA during 1981–83 and served on its Board of Directors during 1980–83 and 1987–93. He is also currently a member of the Technical Committee for Musical Acoustics of the ASA.