Structured Audio: Creation, Transmission, and Rendering of Parametric Sound Representations

BARRY L. VERCOE, WILLIAM G. GARDNER, AND ERIC D. SCHEIRER, STUDENT MEMBER, IEEE

Invited Paper

Structured audio representations are semantic and symbolic descriptions that are useful for ultralow-bit-rate transmission, flexible synthesis, and perceptually based manipulation and retrieval of sound. We present an overview of techniques for transmitting and synthesizing sound represented in structured format, and for creating structured representations from audio waveforms. We discuss applications for structured audio in virtual environments, music synthesis, gaming, content-based retrieval, interactive broadcast, and other multimedia contexts.

Keywords*—Audio coding, audio systems, digital audio broadcasting, multimedia computing, multimedia data bases, music.*

I. Introduction

In this paper, we introduce the concept of *structured audio* representations. These are description formats that are made up of semantic information about the sounds they represent and that make use of high-level or algorithmic models. Certain representations that have been previously examined elsewhere contain structured information, such as musical-instrument digital interface (MIDI) musical-event lists, music synthesis languages, and the linear-prediction model of speech. We will interrelate this work and subsume it into a larger discussion about general-purpose structured-sound representations.

We will discuss various examples of technology and applications that exemplify our ideas about structured media description and explain why this way of organizing things is a valuable one. We are not attempting to provide an exhaustive review of research into audio synthesis, analysis, or coding but to highlight concepts and previous work that relate to our narrative; many of the subdomains we will discuss are large enough to support extensive review articles in their own right.

Manuscript received July 1, 1997; revised December 4, 1997. The Guest Editor coordinating the review of this paper and approving it for publication was T. Chen.

B. L. Vercoe and E. D. Scheirer are with the Machine Listening Group, MIT Media Laboratory, Cambridge, MA 02139 USA.

W. G. Gardner is with Wave Arts, Inc., Arlington, MA 02174 USA.

Publisher Item Identifier S 0018-9219(98)03380-5.

We specifically focus on two primary applications of structured sound. First, structured descriptions allow ultralow-bit-rate transmission of audio content, exploiting formerly unexamined forms of redundancy in signals; second, they also allow perceptually informed access, providing more naturalistic interfaces for the search and manipulation of sonic data. We will also describe a structured-sound system we have developed for the Motion Pictures Experts Group (MPEG)-4 standard, which unifies many ideas about algorithmic and structured compression in a context of audio coding, and highlight future directions toward annotation methods applicable to sound-understanding systems.

A. Rationale

There have been many explorations into methods of analyzing and synthesizing sound; it is now apparent that much of this work may be profitably connected and related in order to build transmission and analysis systems that are more powerful than their individual parts. Our own projects on "NetSound" [14] and the MPEG-4 structured audio decoding standard [97] have shown us the value of integrating research at the forefront of sound analysis, musical sound synthesis, sound-effects processing, and sound coding. By exploring such connections, we hope to stimulate discussion about structure in sound and the ways that it may be exploited in many sorts of audio applications.

We especially hope to demonstrate how much potential, both for researchers and for application developers, there is for broader perspective in thinking about the semantics of sound. In the past, "structured audio" has sometimes been envisioned only as "three-dimensional (3-D) audio for virtual reality." While this is one important component, we argue that there are many other useful kinds of structured audio as well. Representations that contain explicit information about the perceptual qualities of a sound provide expanded sound quality and functionality for sound systems of all kinds.

As integrated multimedia applications and signal-processing systems become more common, it is important for audio and acoustics researchers who focus on one area to be aware of the rest of the world of sound research, to understand how results from one area may be leveraged to another, and to assist in the construction of multidimensional tools that draw from many different disciplines.

B. Definitions

While any attempt to draw boundaries between different representations, coding schemes, or processing algorithms inevitably produces gray areas, we will attempt to define our terms carefully in order to provide context for later sections.

A *structured media representation* has a number of features by which it can be identified. One characteristic is that structured representations encode a signal, in our case a sound, in relation to a *model*—that is, they make assumptions about the nature of the sound being represented and use these assumptions in defining a parameter space. Sounds that fall outside the assumptions for a particular model will often be impossible to represent accurately in its parameter space. We might further say that the lower the dimensionality of the feature space, the "more structured" the representation is—in a very-low-dimensional representation, each coordinate must have relatively "more meaning." In a structured representation, the parameters used have a "semantic meaning" in a larger context. That is, the parameters are individually interpretable as representing certain high-level features about the sound, and their manipulation gives simple control over perceptual aspects of the sound.

Traditionally, within a certain signal-processing and pattern-recognition literature, the term "parametric," especially as distinguished from "nonparametric," is used to describe representations in which the dimensions of variation of a sound can be described using a simple equation, generally continuously varying in the parameters. Our use of this term subsumes both meanings and, more generally, any model for sound in which different sounds can be distinguished using a set of parameters, regardless of the nature of the particular model.

It is important that we discuss structured sound models in terms of their *perceptual* aspects—a fundamental concern in the development of structured audio representations is an understanding of how listeners perceive sound. We are less interested in properties often desirable from an engineering standpoint, such as perfect reconstruction of a target sound, than we are in the relationship of a parametric model to human audition. Thus, in many cases, the development of error criteria for comparison of different representation methods is problematic—a simple "minimum squared error" criterion does not suffice because humans do not measure "noise" or "reduction in quality" of sound this way.

To contrast two simple examples, a digitally sampled waveform encoded in pulse code modulation (PCM) format is not a structured representation—it is a high-dimensional description of the sound, there is no contextual model embodied in the parameter space, and the individual sample values have little meaning taken one at a time. It is only by examining many, many samples in relation to one another (and likely transforming them into some other domain) that we might begin to understand the perceptual qualities of the sound. Although the description may be made more compact by using subband coding and perceptual approximation [40], [46], the resulting bit stream still contains little structure.

On the other hand, a music track stored in MIDI format is a highly structured representation. MIDI data are very compact, requiring only hundreds of bits per second rather than hundreds of thousands for PCM data, and the individual parameters represent perceptual aspects of the signal such as "pitch" and "onset time." There is a very strong music-generation model used to allow this concise description. The MIDI representation has drawbacks, though—it is not applicable to description of subtly changing sounds, the description does not completely specify a sound (because timbre is left out of the MIDI description), and it is very difficult to generate a MIDI representation automatically from a given waveform. However, as the MIDI standard is a very important, widely used sound-encoding scheme, it is clear that these are not insurmountable barriers to the utility of a representation.

In fact, for many domains, there is great applicability for sound representations that only encode partial information about a sound or that cannot be easily encoded from a sound waveform. The task of *coding*, in which we are interested in a complete loop—from encoding through transmission to decoding—is only one area within the study of structured-sound representations. For other domains—such as interactive music systems, content-based retrieval, and virtual environments—we may consider other representations with other features.

The structured formats we will discuss have many properties that may be used to contrast them and examine their suitability for various applications. We will consider the degree to which formats are *encodable*—that is, how easy it is to extract the representation directly from an audio waveform—and whether they are *synthesizable*—whether the representation can be turned back into sound without more information. For synthesizable descriptions, we wish to know how *accurate* the resynthesis may be with respect to a target sound. We are interested in the *generality* of a representation—to know whether it applies to every sound or only a few specific kinds of sound. Sound formats that are highly *semantic*, that is, for which the individual parameters have a clear high-level meaning, are easier to manipulate. Last, in many applications, we wish to consider the *efficiency* or *compactness* of a sound format; the more compact a representation is, the less bandwidth it takes to transmit sounds in that format.

These properties govern the situations in which a certain representation is useful. There is no reason to believe that one sound model will be appropriate for all tasks; part of the goal of this paper is to explore the range of possibilities for representations applicable to many different domains.

It is important to realize that there is an essential tradeoff in structured-signal representations between efficiency and generality. That is, representations that are maximally compact for a particular type of sound must make use of strong assumptions about the process by which that sound is generated. Because of these assumptions, the representation is a poor one for sounds that do not lie in the domain being modeled. For example, the MIDI representation is useful for representing certain kinds of sounds, such as piano music, because the pitch/onset/duration model is appropriate for that domain. Other sounds, such as speech, are not modeled well within this framework, and so other representations must be discovered that are more appropriate for those domains. By embodying specific assumptions about the nature of the sound, the MIDI format allows for very concise description of sounds that have this nature but precludes its use for sounds that do not. In contrast, the PCM format is very general; it allows the (inefficient) representation of any band-limited sound whatsoever.

This tradeoff is not a drawback when designing representations for the transmission of sound, as it is always possible to conjoin two representations into one by adding a few extra bits to indicate which representation to use. The "coding space" of the joint representation is then the union of the coding spaces of the two underlying representations. This approach is represented by general-purpose synthesis languages; see Section II-D.

Structured-audio methods are useful for sound compression and transmission because they exploit signal redundancies that other coding methods cannot. Existing sound-compression methods work in one of two ways. Lossless coders remove *entropic* or *information-theoretic* redundancy [103] from a digitally sampled signal. This redundancy arises from the fact that successive samples are not statistically independent and that some sample values occur more often than others. Sounds coded in this manner may be exactly reconstructed. Perceptual coders remove *perceptual* redundancy from a signal, that is, redundancy created by overspecifying the sound format with regard to the human perceptual system and including details that cannot be perceived. When the perceptually unneeded information is removed, the result is a lossy coding scheme; that is, the original waveform cannot be reconstructed exactly from the coding. However, the reconstructed waveform will sound like the original to a human listener.

Most sound signals also contain what we term *structural redundancy,* which arises in several ways. First, many notes or sound events in a soundtrack sound the same or nearly the same; in waveform coding, these events will simply be represented multiple times. Using timbral models, we can concisely describe the similarities and differences between different notes with the "same" timbre. Many soundtracks contain sections of repeated material; for example, footsteps, drumbeats in popular music, and exact recapitulations in classical music. If we can represent these repetitions symbolically, we can reclaim a great deal of redundant information. Last, many sounds are more simply represented as processes than as waveforms; for example, consider speech sounds processed to add artificial reverberation. We may use a highly efficient speech coder to transmit the flat, unreverberated speech, and parameters to (or an algorithmic description of) a reverberation algorithm, and do the effects postprocessing after the speech is decoded. This is a more efficient transmission method than using a general-purpose coding scheme to transmit the reverberated waveform.

C. Overview

The remainder of this paper will present an in-depth discussion of various methods for generating and processing sound. Table 1 contains a summary of the various analysis and synthesis methods discussed, showing where in the paper they are presented and their properties according to the list in the preceding section.

Section II describes representations appropriate for creating sound from structured descriptions, including music synthesis, effects processing, and our recent work on the MPEG-4 structured-audio standard. Section III describes the state of the art in extracting structured descriptions from waveform audio, including parameter estimation and so-called "computational auditory scene analysis." Section IV discusses applications for structured-audio representations, such as very-low-bit-rate transmission, content-based retrieval, and sound-manipulation systems. Last, Section V concludes with brief thoughts on the implementation methodologies useful for structured-audio systems and describes future research prospects in this area.

II. Sound from Structure

In this section, we discuss methods of generating sound from a structured description. If we wish to communicate a sound description over a channel, we require both an encoder and a decoder. The encoder and decoder have *a priori* knowledge of the underlying sound model; the description is then a set of model parameters, expressed in symbolic or numerical form. Thus, the description is a parameterized sound representation. Principally, we are interested in descriptions that are reasonably compact and describe the sound in some meaningful or semantic way.

The encoder may obtain the parameters by analyzing existing sound, in which case the encoder/decoder process is often termed an *analysis/synthesis* procedure. The encoding might also be strictly generative, however, such as in the reproduction of a musical score or other scripted sequence of sound events. In this case, the model parameters are not obtained automatically but by a human designer authoring them to create the desired content or from a generative algorithm implementing a world model.

A. MIDI and Other Event Formats

MIDI is a combined hardware and software specification for musical content [57], [69]. It allows for the real-time transmission of musical-instrument performance data,

Table 1 Properties of Various Structured Representations of Sound. See Referenced Sections for Details. "Encodable" Representations Are Those That Can Be Automatically Extracted from Sound with Current Technology. "Synthesizable" Representations Are Those That Can Be Turned Back into Sound; For These, "Accurate" Models Are Those for Which the Reconstructed Sound Must Be Very Similar to the Original or Target Sound. "General" Representations Are Those That Can Encode All Sorts of Sound. "Semantic" Representations Assign High-Level Meaning to the Individual Parameters. "Efficient" Representations Are Those in Which Different Sounds May Be Distinguished with Relatively Few Bits

	Section	General	Encodable	Synthesizable	Semantic	Accurate	Efficient
PCM	I. B.	Yes	Yes	Yes	No	Yes	No
Subband coding	I. B.	Yes	Yes	Yes	No	Yes	Somewhat
MIDI/Event lists	II. A.	No	No	Yes	Yes	No	Yes
Sampling	II. B. 1.	Yes	Somewhat	Yes	Somewhat	Yes	Somewhat
Additive Synthesis	II. B. 2. III. A. 3.	No	Yes	Yes	No	Somewhat	Somewhat
Subtractive Synthesis	II. B. 3.	No	No	Yes	No	Yes	Yes
LPC speech model	II. B. 3. III. A. 1	Speech only	Yes	Yes	Somewhat	Yes	Somewhat
Granular Synthesis	II. B. 4.	No	Somewhat	Yes	Somewhat	Somewhat	Yes
FM Synthesis	II. B. 5.	Music only	Somewhat	Yes	No	Somewhat	Yes
Physical Models	II. B. 6. III. A. 3.	No	Difficult	Yes	Yes	Yes	Yes
Perceptual Models	II. B. 7.	No	Difficult	Somewhat	Yes	Somewhat	Yes
Model-based effects	II. C.	As needed	No	Yes	Yes	Somewhat	Maybe
Synthesis languages	II. D.	No	No	Yes	Somewhat	Yes	Yes
MPEG-4 scenes	II. D.	Yes	Difficult	Yes	Yes	Yes	Yes
Phonemes	III. A. 2	Speech only	Somewhat	Somewhat	Yes	No	Yes
Pitch tracks	III. B. 2	No	Yes	No	Yes		Yes
Rhythm/Tempo tracks	III. B. 2	Music	Somewhat	No	Yes		Yes
Timbre Models	III. B. 2	Music	Difficult	Difficult	Maybe	Maybe	Maybe
Auditory Scenes	III. C.	Yes	Difficult	Difficult	Yes	Difficult	Maybe

including note start and stop events, control parameter updates, and patch, or timbre, changes. Although MIDI has a number of deficiencies, such as poor time resolution and low bandwidth [70], it is an overwhelmingly successful and useful standard.

The MIDI file format is a standard method of representing time-sequenced events and can efficiently represent a musical performance as a set of note onset and offset times with associated amplitudes, patch numbers, and control parameters. The file does not encode the exact sound to be played, only a patch number; the sound realization depends on the device used to play back the sound. Another standard, called General MIDI, defines a set of standard musical-instrument sounds and assigns patch numbers for them; any musical instrument capable of rendering the General MIDI sound set can play back General MIDI files. This standard does not guarantee a high degree of sonic consistency across platforms because different instruments use different methods to synthesize the sounds, but it does provide an extremely compressed representation for musical data.

MIDI is one example of an *event-list representation.* An event list is a sequence of control parameters that, taken alone, do not define the quality of a sound but instead specify the ordering and characteristics of parts of a sound with regard to some external model. The external model is often encapsulated in the form of a hardware synthesizer using one of the methods in Section II-B or as a set of descriptions in an algorithmic synthesis language—see Section II-C. Many event-list formats only contain information about timbre at the highest semantic level, in the form of specifying which sound to use from a list of those available; others contain rudimentary timbral data. For example, a new addition to the MIDI specification allows the inclusion of downloadable sound samples (see Section II-B1), and the score file language for the algorithmic synthesis language "Csound" (see Section II-D) can contain data for the synthesizer to process.

Event lists are particularly appropriate to soundtracks, which are characterizable as a series of discrete events; piano and other percussive instruments are the prime example. Musical instruments with more continuous control capability, such as the violin, are harder to describe via parameters in an event list. Speech and singing are very difficult due to the slow and subtle transitions from sound to sound.

Very powerful and user-friendly software packages have been developed for authoring MIDI event lists. Called "sequencers," because they allow the specification and modification of event sequences, they have become an important tool of the modern composition studio. It is likely that new versions of these tools will be created to handle new structured-sound formats.

Related to event lists, and omnipresent in the real-world musical community, is *standard musical notation.* This familiar "dots-on-lines" notation, having evolved over many centuries of Western music, is a powerful and flexible medium for the description of certain kinds of music. It is highly graphical in nature, however—cues such as the shape of a particular object on the page, or whether it is filled in or left open, indicate the durations of notes in the composition. As a result, it is difficult for today's computers to use this notation effectively. Many recent research projects attempt to allow the interconnection of computer-based methods and standard musical notation [101].

B. Sound Representations and Synthesis Methodologies

This section will discuss various algorithmic representations of sound, most developed as music- or speech-synthesis techniques. It is important to note that although the text is describing synthesis algorithms, in each case there is an implicit domain (or "representation space") of parameters used to control the synthesis method and a range of sounds easily generated with the method. The suitability of each of the methods for a particular application depends on the sound that must be generated and the sophistication of parameterization that is available. For each method, the parameters in the domain map to a specific sound from the range in a complex model-dependent way; thus, each synthesis method is also a parametric sound model.

Table 2 summarizes the typical domain and range of each method discussed here.

1) Sampling: Many modern music synthesizers are based on *sampling synthesis* [64]. Individual instrument sounds (notes) are digitally recorded and stored in memory in the instrument. When the instrument is played, the note recordings are reproduced and mixed (added together) to produce the output sound. This can be a very effective and realistic synthesis method, although it requires a lot of memory.

Several techniques can greatly reduce storage requirements. First, by transposing the pitch of a sample during playback, a single recording can reproduce a range of notes. Second, quasi-periodic sounds can be "looped" after the attack transients have died out by replaying one or more periods of the recording. Thus, the internal recording stored in memory only consists of the attack section and a looping section. The synthesis technique based on looping a single waveform is known as *wavetable synthesis;* sampling synthesizers are sometimes called wavetable synthesizers.

A looping section composed of a single period of the fundamental frequency will only reproduce perfectly harmonic timbres, which tend to sound lifeless. To synthesize nonharmonic partials, the looping section must contain more than one period of the fundamental. Long loops, on the order of seconds, may be necessary to reproduce complex sounds like a bowed string section realistically.

When a sound that should normally decay is looped, the amplitude envelope of the sound must be artificially restored; this is easily accomplished by multiplying the output of the sample playback engine with an amplitude envelope. Another useful technique is to process the output with a frequency-dependent filter whose response is changed over time. This can restore a natural timbral evolution to an otherwise lifeless and static sound. A recent development is the creation of new representations in which sound samples are associated with simple processing methods that describe envelopes and filter parameters [89].

Sampling is a technique that can be applied to any type of sound. Compact descriptions are obtained when the original sound can be decomposed into recurring elements, like notes in a musical piece, or if the elements themselves are quasiperiodic and appropriate for a looped representation. Sampling is poor at encoding transitions between elements as part of a larger context. For example, speech can be reconstructed by concatenating individual phoneme recordings, but the resulting synthesis may sound awkward because the transitions between phonemes are incorrect and contextual parameters such as stress and intonation are missing. When longer term information about these continuous parameters is added to the representation, high-quality speech may be produced.

Sampling can be used effectively to reproduce environmental sounds, as witnessed by the large number of

Table 2 The Domain (Parameterization) and Range (Sounds Typically Generated) for Each of the Synthesis Methods Discussed in this Section. See Text for Details on Methods. While the Theoretical Ranges Are Broader Than Shown Here, We Focus on Those Sounds for Which the Method Is Best Suited in a Practical Sense

Method	Domain	Range
Sampling	PCM samples; control parameters for algorithmic manipulation	Discrete notes and other isolated sound events
Additive synthesis	Time-varying amplitudes and frequencies of sinusoids	Continuous periodic sounds
Subtractive synthesis	Excitation sound; filter parameters	Speech and other source-filter sounds
Synchronous granular synthesis (FOF)	Grain shape, pitch, format structure	Speech and singing
Asynchronous granular synthesis	Grain shape, control parameters, time-frequency distribution	Noisy, textural sounds
FM synthesis	Carrier and modulation waveshapes	Harmonic & inharmonic notes with discrete spectra
Physical modeling	Model parameters (lengths, stiffnesses, etc.)	Plucked, struck, bowed, and blown instruments
Perceptual modeling	Model parameters (depend on model)	Sound effects, sound textures

sample libraries devoted to sound effects for film and video postproduction. The process of creating sound effects for film, called "Foley" after its originator, is rather similar to a sampling synthesis paradigm. An expert Foley artist uses special (or sometimes common) equipment to produce sounds like thunder, footsteps, gunshots, etc.; these sounds are recorded to an effects track and later mixed with the rest of the film sound. The samples often require additional processing to fit the needed context, for example, trimming, time-scale modification, reverberation, flanging, filtering, etc. The subject of model-based postproduction is discussed in a later section.

2) Additive Synthesis: Additive synthesis is a method of synthesizing sound from the superposition of sinusoidal components, each with time-varying amplitudes and frequencies. Quasi-periodic musical sounds, consisting of a set of partials, are particularly well suited to this representation, and consequently, additive synthesis has been used extensively to study musical-instrument timbre [35], [43], [82], [83], [110] (see also Section III-A). Amplitude and frequency data are created algorithmically or by analyzing acoustic data; tones are resynthesized by driving a bank of sinusoidal oscillators with this information.

Impulsive sounds cannot be realistically synthesized using a small number of sinusoidal tracks. This means that additive techniques are ineffective at synthesizing, for example, a flute tone that contains a great deal of breath noise. Serra [102] proposed an extension to the additive model that includes a time-varying noise component. Sinusoidal analysis is performed by computing the short-time Fourier transform (STFT), finding spectral peaks, and matching peaks in adjacent analysis frames to form sinusoidal tracks [66]. The difference between the original sound and the synthesized sinusoids is computed, and the spectrum of this residual is smoothed. The synthetic noise component is defined by the time-varying set of spectral envelopes and resynthesized using the inverse STFT. Using this model, it is possible to resynthesize the breath component of a flute tone and vary this independently of the pitched component.

3) Subtractive Synthesis: Subtractive synthesis is characterized by a harmonically rich source sound that is subsequently filtered, a model that applies to many physical sound-producing systems. Many of the early analog music synthesizers were based on the subtractive-synthesis paradigm. Typically, the output of a periodic oscillator (triangle, sawtooth, square, and pulse waves have been commonly used) is passed through a voltage-controlled filter (VCF) and then to a voltage-controlled amplifier (VCA). Both the VCF and VCA are controlled by simple envelope generators that specify the time evolution of the sound's spectrum and amplitude. Simple networks of oscillators, filters, and amplifiers can synthesize many different sounds from a small number of control parameters, but the resulting tones have a distinctive "analog synthesizer" character that, while sometimes desirable, is difficult to avoid.

Speech synthesizers are often based on a subtractive synthesis model. Voiced speech (i.e., a vowel) is created by a periodic glottal excitation that is filtered by the resonances of the vocal tract (see [80] for a complete discussion of speech production). The resonances, or *formants*, give a vowel-like quality to the sound that is independent of pitch; the perceived vowel depends on the pattern of formant frequencies. Pitched glottal sources can be modeled using periodic pulses, fricative sources can be modeled using noise, and the vocal-tract resonances can be modeled with simple pole-zero filter networks [79]. The formants can be estimated from recorded speech using linear prediction [5], [61] (see also Section III-A1).

Rabiner [78] has described a rule-based speech synthesizer that can convert sequences of phonemes into control signals for a digital speech synthesizer. Modern speech synthesizers, though perfectly intelligible, still suffer from a mechanical quality and are easily distinguished from natural speech. Subtle differences between the subtractive model and the physical speech process, and the lack of pitch and timing cues, create this lack of expression that is important to the human perception of natural speech. It is very difficult to arrive at algorithmic representations of these sorts of cues [16].

4) Formant and Granular Synthesis: Driving a formant filter with a periodic source of impulses results in an output that is a sum of delayed formant time responses; this suggests a time-domain method of generating voiced speech where copies of formant filter impulse responses are overlaid. The synthesis method of Rodet [88] defines a time-domain formant wave function (FOF, for *fonctions d'onde formantique*), parametrized in terms of center frequency, amplitude, and bandwidth. The FOF's are synchronously overlaid in time to create a pitched output; summing multiple FOF's allows the creation of complicated spectral shapes to simulate speech or instrument sounds [23].

The general synthesis method of summing parametrized time functions is called *granular synthesis* [24], [85]. Typically, the sound components, or *grains*, are band-limited time functions on the order of 10–20 ms duration obtained by windowing an existing signal or by applying an envelope to an oscillator. These window functions control both the time-domain (length) and frequency-domain (bandwidth) characteristics of the grain. Each band-limited grain tiles a portion of time-frequency space; thus, by summing many grains in a specific pattern, it is possible to control the distribution of energy in time-frequency space. This synthesis method is often used in a way that is highly abstract and best suited to the generation of noisy or textural sounds like water, wind, applause, and fire [86].

5) Frequency Modulation (FM) and Wave-Shaping Synthesis: One of the most widely used synthesis techniques is FM synthesis [17]. FM results when the instantaneous frequency of a waveform, called the *carrier*, is modulated according to the amplitude of a modulating wave. This creates symmetrical sidebands above and below the carrier frequency, with each sideband offset from the carrier by an integer multiple of the modulating frequency. Increasing the depth of the modulation (the modulation scaling factor is called the modulation index) creates sidebands of greater intensity. By suitable choice of the carrier and modulation frequencies, it is easy to obtain harmonic spectra or bell-like inharmonic sounds that have a preponderance of low-frequency energy and a spectral bandwidth controlled by the modulation index. Furthermore, the dependence of the amplitudes of spectral components on the modulation index is complex, so that slowly changing the modulation index imparts a rich timbral evolution of the sound.

Using a simple FM circuit, where both the modulation index and the signal amplitude are controlled by envelope generators, it is possible to synthesize brass-like and woodwind-like tones. A slightly more complicated circuit is required to generate percussive sounds containing inharmonically related partials. Bowed string and piano tones can be synthesized using complicated modulating waves [52], [100]. Although FM techniques provide a large variety of musically useful timbres, the sounds tend to have an "FM quality" that is readily identified. Also, there are no straightforward methods to determine a synthesis algorithm from an analysis of a desired sound; therefore, the algorithm designs are largely empirical.

FM synthesis is an efficient method to create harmonic sounds whose spectral content evolves over time. It is one of a general family of techniques called wave-shaping synthesis [53], [84]. Wave shaping is accomplished by applying a nonlinear function to the output of a sinusoidal oscillator, causing the creation of harmonic distortion products. These overtones create a rich harmonic spectrum. The harmonic content can be made to evolve over time by changing the amplitude of the oscillator or by adjusting the nonlinear distorting function. The technique is well suited to reproducing brass tones [7] because these instruments have the property that the high-frequency energy depends on the amplitude of the fundamental due to nonlinearities in the mouthpiece. An analysis method is possible; Beauchamp [7] derived the nonlinear distorting function from acoustic measurements of brass tones, and the resulting synthesis algorithm produced realistic brass-sound synthesis.

6) Physical Modeling: The synthesis of musical sounds by physical modeling of the instrument is an approach that has recently come to the forefront of synthesis technology [32], [107], [108]. Physical modeling offers faithful sonic reproduction under both static and dynamic playing conditions. The control parameters of the synthetic instrument are identical to the real controls, providing a natural expressive interface. The simulation of instruments is sufficiently accurate to include idiosyncratic behavior, such as a saxophone that squawks when misplayed.

An acoustical instrument can be simulated by solving the differential equations (e.g., the wave equation) that govern its behavior. The problem can be discretized by using numerical techniques to solve for the values of acoustic variables at a grid of points distributed throughout the instrument. The direct solution of these differential equations is of course prohibitively expensive for real-time performance; however, wind and string instruments are based on acoustic tubes and vibrating strings, which may be accurately modeled as one-dimensional waveguides.

One-dimensional waveguides are efficiently modeled in signal processors using bidirectional delay lines; the signals in the delay lines represent the two modal waves that propagate in opposite directions in the waveguide. For a vibrating string, rigid termination at the end of the waveguide is implemented by connecting the output of one delay line to the input of the other through an inverting gain (multiplication by -1). Any frequency-dependent losses or dispersion in the acoustic medium can be lumped into discrete elements placed in series with the delay lines. Consequently, it is possible to model the physics of an acoustic tube or a vibrating string using a pair of bidirectional delay lines in series with a frequency-dependent filter. This is easily implemented for real-time performance on typical digital signal processors.

In practice, waveguide synthesis models require additional components to simulate a given instrument. For example, in a clarinet model, one end of the waveguide is terminated by a single reed mechanism and the other end is terminated by a bell. The reed mechanism acts to inject energy into the waveguide; efficient physical models of the single reed have been developed [107]. The bell radiates high frequencies and reflects low frequencies, which are easily implemented using a crossover filter at the terminating junction [108]. The clarinet model may also include models for the tone holes; these lead to yet more complicated filtering in the waveguide [113].

Other physical models that have been developed include brass instruments [19], human singers [20], bowed string instruments [107], and flutes [48]. In addition, two-dimensional waveguide meshes have been developed [112]; these allow the modeling of drums, soundboards, cymbals, gongs, etc. Only simple physical systems have yet been modeled with these methods, and it is still an open problem to determine the best way to realize the damping filters and nonlinearities for a particular model. These features are particularly important, as most instruments sound very similar when simplified to a linear waveguide model; it is the addition of nonlinear elements that gives specific instruments their characteristic expressivity.

7) Perceptual Models: A desirable attribute of a sound representation is that it allow the parameters to control independently perceivable qualities of the sound. Many of the synthesis algorithms we have discussed provide perceptual controls such as pitch and amplitude as a natural result of their design; we seek methods that can systematically provide perceptually relevant control over other aspects of sound.

Historically, this problem has been approached by considering a set of sounds and experimentally establishing the perceptually relevant axes in this space of sounds using multidimensional scaling [42], [105]. This is accomplished by first obtaining similarity judgments for all pairs of the sounds; a mathematical technique is then used to assign the sounds to points in a Cartesian space such that the distances between the points match the similarity judgments for the corresponding sounds. If the similarity judgments are based on only a few perceptual cues, then a low-dimensional space should account for the similarity judgments. Descriptive axes into the space can be determined by correlating subjective or objective parameters with the data. Once the perceptual axes are determined, a synthesis method can be devised with corresponding perceptual parameters [122].

An alternative approach is to perform a principal component analysis of the set of sounds [15], [51], [92]. The result of this procedure is a set of orthogonal basis vectors, linear combinations of which can reconstruct any of the analyzed sounds. Often, a small subset of these vectors accounts for most of the variation in the data, and combinations of these vectors alone can reconstruct any of the original sounds with high accuracy. Consequently, the original sounds can be parametrized as a small set of basis vector weights. It is then possible to interpolate between sounds by interpolating between these weights. There is no guarantee that this interpolation will be perceptually meaningful because the basis vectors are obtained by statistical rather than perceptual means; nevertheless, the basis vectors often represent salient features in the sound.

C. Model-Based Postproduction

The postproduction process turns multitrack recordings, in which each component of the sound scene such as music, speech, and sound effects is represented separately, into a distribution format, such as a monophonic, stereophonic, or other multichannel mix. This mixing process involves the dynamic filtering of soundtrack components, the spatial positioning of sound, and the addition of other spatial effects, such as artificial reverberation [36]. In a strictly musical production, there may be complete artistic freedom in applying these effects; in a film or video production, the spatial cues are carefully coordinated with the visual scene to enhance realism or artistic impact.

Currently, postproduction work is a laborious task done by skilled sound engineers. They use knowledge about the presentation format, artistic context, and sound material to carefully design filtering and spatial effects for a particular piece of content. With structured techniques, it may be possible to perform some of this manipulation algorithmically. A strictly computational implementation requires a *production model* to constrain the processing and determine exactly what effects must be applied to which sounds.

One approach is to use a virtual-reality world model, such as the virtual-reality modeling language [118], where the world is physically defined and the listener's relationship to the world is defined by the camera viewpoint. This approach would not be adequate for many video and film productions; for example, we do not expect the positions of sounds to jump when the camera changes viewpoint within the same scene or to Doppler shift during rapid camera motion. A better model needs to be developed that anticipates cinematographic requirements.

The MPEG-4 standard [97] will contain a scene-description language for the high-level description of audio scenes made of both waveform-encoded and structured-audio sound sources. These sources can be related to each other spatially and temporally and processed with transforms and filters custom designed for the postproduction of each sound scene. As it allows scenes to contain sound sources coded in many different ways, the scene description acts as a metarepresentation for the different coding methods in the standard. Each track of audio may be coded in the way most appropriate to the scene and track requirements. These tracks are then filtered and, if desired, presented spatially using a method that is appropriate to the listener's speaker configuration.

The *Spatialisateur* system [47] is an example of a general system for the spatial processing of sounds, intended for multimedia or virtual-reality applications. It allows control over the position of multiple sounds and provides separate control for reverberation, which is implemented as a set of perceptual controls. The system also produces a variety of output formats, including binaural formats for headphones or loudspeakers, or multichannel surround sound formats. Gardner [37] has recently described a more advanced system that takes the listener's position into account to make the spatial synthesis more robust to listener motion and provide higher quality perceptual cues to spatial location of sounds.

D. Synthesis Languages

Many computer languages for specifying sound-synthesis algorithms have been developed [38], [87], [93]; for example, the popular Csound language [116], which is descended from the MUSIC IV and MUSIC V languages developed by Mathews in the 1960's [65]. Most of these languages are based on *unit generators*, which are simple functional blocks like oscillators, filters, and envelopes that may be connected into networks to describe a signal flow path. The connections can be specified using a functional syntax, such as

$$\text{output} = \text{oscillator}(\text{freq}) * \text{envelope}(\text{time})$$

or made by graphically connecting objects on a screen, such as with the MAX program [77].

Synthesis languages play an important role in structured audio because they can be used to define hybrid representations of sound. That is, many soundtracks can be more effectively represented when components are generated using different synthesis methods. Where a hardware synthesizer likely uses only one or two synthesis methods for all the different sounds it generates, these synthesis languages allow each instrument to be generated with an appropriate method—FM for brasses, physical modeling for strings, sampling for drums. In addition, signal-flow networks may be created that do not correspond exactly to any "standard" method of sound synthesis. Hybrid synthesis methods can be created as needed.

General-purpose sound-synthesis languages that contain both wavetable and algorithmic synthesis methods are theoretically complete in the kinds of sounds they can generate; there are no sounds that cannot be created by using arbitrary algorithms to process wavetable data. However, it requires a great deal of skill to write algorithms that generate pleasing sound algorithmically. As there are not yet robust methods for automatically generating the appropriate algorithm to synthesize a given soundtrack, it is not likely that such languages will replace acoustic musical performance.

We have developed a new language based on MUSIC V called "structured audio orchestra language" (SAOL) for the transmission of synthetic sound algorithms in the "structured audio and effects" component of MPEG-4 [95], [97]. SAOL brings many new features to this lineage, including a more powerful functional abstraction, a new effects-processing metaphor, better score-based controllability, and the possibility of describing dynamic sequencing algorithms and virtual performers. The language retains features from Csound that allow the development of efficient compilers for dedicated signal-processing hardware.

Between this language and the scene-description capability described above, the MPEG-4 standard provides a very powerful, efficient method for encoding and transmitting all sorts of sounds. It can be viewed as the unification of all of the structured-sound methods we have described to this point. "Natural" sounds are represented as PCM samples, subband coded waveforms, or with a sophisticated linear predictive coding (LPC) model. "Synthetic" sounds are represented as algorithms in SAOL, controlled by a MIDI file or a score written in a new event-list format. Any of the methods described above, or hybrids or new methods, may be represented as SAOL code. These pieces are composited together under the control of software algorithms, also described in SAOL, for manipulating and filtering the sounds.

III. Structure from Sound

In this section, we survey methods for processing sound, especially for the purpose of extracting structured representations directly from acoustic waveforms. In some cases, such as LPC (which can be viewed as a specific example of a *parameter-estimation* technique), we are interested in synthesizable representations that at some future point will be turned back into sound. In others, we are interested in *characterizing* or describing a sound and using the resulting structured information to solve some problem other than synthesis. We call methods for analyzing sound and producing synthesizable representations *invertible* methods. Certain approaches lie in the gray area between these interests; in some of these cases, such as "automatic transcription" systems, the representations extracted may be similar to the event-list formats described in Section II-A.

Multimedia applications such as search and retrieval, signal manipulation, and signal classification are particularly well suited to nonsynthesizable structured-audio representations. Highly structured signal descriptions are necessary for these applications because in order for humans to interact with signals, the interface must provide information about the semantic sound content to the user. This perceptual representation is most readily available from a structured description. In some cases (for example, MIDI and other event-based representations), the high-level information is created by the author directly; in others, we are interested in extracting this information from a low-level representation (typically a waveform) and creating a hybrid or *annotated*, description. The high-level structured information acts as metadata on the underlying low-level content.

Although there are no representations yet that allow this, it is interesting to imagine a description format in which data suitable for resynthesis, such as LPC soundtracks, synthesis languages, or MIDI files, could be annotated with higher level semantic information, such as information on speaker identity, the phonemes spoken, or the emotive content of the speech. The future MPEG-7 work on standards for search and retrieval of multimedia information is currently envisioned to contain these sorts of representations; thus, the envisioned MPEG-7 standard can be seen as unifying the various methods of *describing* sound in the same way as MPEG-4 unifies methods for *generating* sound.

A. Invertible Parameter Estimation

Parameter-estimation approaches attempt to find the best fit of a sound-generation model to a target sound by analyzing the sound and finding model parameters that allow the model to best approximate it. In invertible cases, the model and parameter set are a complete enough characterization of the signal to allow resynthesis (although a particular type of lossy synthesis); in others, which we term *descriptive* analysis, the goal is not resynthesis but some type of signal understanding.

1) Linear Prediction: The most common and well-studied parametric representation for sound is undoubtedly the *linear prediction* model of speech, which was also discussed as a form of subtractive synthesis in Section II-B3. This model has shown great utility in ultralow-bit-rate coding of speech for transmission, as well as serving as a front end for certain attempts at speech recognition (although it has been largely supplanted by cepstral frames for this purpose in modern systems).

The LPC model is useful for speech transmission because it serves as a crude approximation to the actual physical process of speech generation. There is a large literature on LPC analysis of sounds, both "classical" analyses for error criteria such as mean-squared error [61] and more recent work that attempts to use perceptual error criteria to guide the parameter estimation [6]. This process generally takes the form of spectral whitening with an inverse filter, followed by subtraction and pitch estimation to model the residual. The spectral whitening may be accomplished in many ways and in any of several domains, including the time domain, frequency domain, cepstral domain, and autocorrelation lag space [61]. In many cases, the correspondence is so direct between the LPC model and the speech model that we may estimate parameters of the vocal tract itself based on LPC analysis of the sounds produced [18].

LPC-based methods form an important component of the modern toolkit for low-bit-rate coding. Using codebook-excited linear prediction techniques, transmission that is perceptually lossy (that is, degraded in quality) but still adequate for understanding content and identifying speakers may be achieved at bit rates as low as 2 kb/s. Perceptually "transparent" coding (that is, coding in which the resynthesized signal is perceptually indistinguishable from the original source) may be achieved for speech at about 16 kb/s. International standards for sound transmission at low bit rates such as MPEG-4 and those of the International Telecommunications Union—Telecommunications Sector (ITU-T) include LPC variants.

2) Phonemic Encoding: A second invertible method for modeling speech, using even lower bit rates than LPC, is phonemic encoding. Using a speech recognizer, we model speech as a sequence of phonemes, perhaps in conjunction with pitch track or vocal-quality information. Using a speech synthesizer, we can recreate speech containing the same verbal content as the original sound but with a different speaker's voice. As phonemic speech rates are extremely low (perhaps 4–8 phonemes/s) and the phoneme alphabet rather small (human languages have anywhere from around ten to around 60 phonemes), the resulting bit stream is extremely compact. However, there are not yet widespread models for synthetic speech that allow the presentation of cues to speaker identity [25] or emotional quality [16] in a robust, yet compact, fashion. Speech analysis/synthesis is nonetheless an appropriate transmission technique for applications in which these cues are not of paramount importance.

3) Model Parameter Estimation: In theory, any of the synthesis methods described in Section II may be the target model for a parameter-estimation approach, but in practice, some are more appropriate than others. Synthesis methods that are heavily hybridized, such as those expressed as "instruments" in high-level music-synthesis languages, correspond to a very complex parameter space that is not easily searched. Methods that can be parametrized by a vector of known, fixed-length, scalar values, such as additive, FM, and physical modeling synthesis, are more amenable to such an approach.

The earliest mechanized attempts at model-based sound analysis/resynthesis involved the *channel vocoder,* in which the subband envelopes of a sound are calculated and then applied to modulation of some other sound. Dudley's classic work with the channel vocoder used only analog electronics; his early work, with the goal of speech transmission, preceded the advent of digital computers altogether [27].

The analysis of sound by estimating parameters for additive synthesis has been a historically fruitful one. In the early days of computer music, this process was called "analysis by synthesis" [83] and sparked a great deal of important research on musical timbre, such as the discovery that the relative temporal evolution curves of partials in a complex tone are an important characteristic of its timbre and that the attack transient is a more important perceptual cue to instrument type than the steady-state portion of a sound.

Early work on additive-synthesis modeling used painstaking hand analysis of spectra to trace out the relative strengths of harmonics of a complex sound [110]. This approach was later replaced by automation with the *phase vocoder* [26], [72], which performs a similar task automatically and often generates a lower dimensional representation. Further refinements include *sinusoidal analysis* [66], which models a sound as the sum of a set of sinusoids whose amplitudes and frequencies vary in time. All of these methods generate representations appropriate for additive resynthesis and also for signal analysis and manipulation to various degrees.

Parameters to other synthesis techniques have been estimated in several ways. Smith [106] used sophisticated digital-filter theory to characterize the formant structure of a violin sound and used that knowledge to create a physical model of the violin body. More recently, Casey [13] and Horner [45] have used learning methods (neural networks in Casey's case, genetic algorithms in Horner's) to perform a search through the synthesis parameter space for a given model. That is, using a fixed, but parametric, synthesis model such as FM synthesis or a physical model of a musical instrument, with which we wish to model a target acoustic sound, we use the following steps:

1) synthesize a sound using test parameters;
2) compare the resulting synthetic sound with the target sound;
3) adjust the test parameters in a way that will make the synthetic sound more closely match the target;
4) return to step 1).

Obviously, the proper methods for 2) and 3) are not at all obvious for most models—What is the proper domain of comparison? What is the best adjustment step?—but good results have been empirically demonstrated for certain instrument and synthesis models.

The differences between neural networks and genetic algorithms for this sort of procedure are subtle, but a short discussion is useful. In neural-network systems, a set of desired outputs is matched through a set of recursive update rules to a set of training inputs. In Casey's case, the inputs were a dimensionally reduced sound model, and the outputs the parameters to a physical model of a musical instrument. The result of this matching process is a set of *training weights* that efficiently represent the covariance of output with input but may be difficult to interpret semantically. In contrast, in the genetic-algorithms framework, the algorithms themselves are the space of learning; the algorithms are gradually (or, in some cases, suddenly) adjusted until they give the desired result. In many cases, the result may be interpreted by a human reader; however, genetic processing may also expand the effective parameter space unnecessarily.

The estimation of parameters for physical models has a particular advantage over equivalent estimation for additive, FM, or other abstract synthesis models in that the resulting parameter set has a clearly understandable interpretation, which aids in further signal manipulation. For example, Casey's estimation of clarinet parameters yields such data as "breath speed," "reed stiffness," "mouth pressure," "tube length," and so forth. In contrast to this, sinusoidal modeling and FM synthesis parameter estimation do not correspond to data spaces so easily interpretable by musicians.

While models with broad parameter spaces such as FM synthesis and sinusoidal modeling may be equally useful for *engineering* applications, such as compression and transmission, models with parameters that are more semantic are clearly superior for authorship and manipulation of structured sound. On the other hand, models of the former type are more general, and thus less information about model selection and characterization needs to be presented. A similar distinction applies to phonemic encoding versus LPC coding of speech. While LPC-coded speech is more general, and able to capture more "natural" variation in a speech signal, the phonemic encoding is easier to manipulate at a high semantic level; for example, replacing one word with another.

B. Descriptive Analysis of Sound

The previous sections described methods for analyzing sounds and estimating parameters to sound models for future resynthesis; however, there has also been a great deal of research into structured-audio representations that cannot be converted back into sound. While these models are not applicable to the transmission of sound at low bit rates, they often fall instead into the *signal understanding* application domain. That is, they enable applications such as multimedia retrieval or signal manipulation tools that allow an author to control the alteration of digital sound using perceptually based guidelines.

1) Speech Processing: The most common signal-understanding methods are certainly speech-processing algorithms, such as speech-recognition, speaker-identification, speech-understanding, and language-identification systems. Many volumes have been written on these techniques; we will not discuss them further here.

2) Musical Features: Aside from speech applications, probably the most common signal-understanding method is the *pitch tracker.* Many musical and vocal sounds have a pitch, which is an important cue to high-level understanding—in speech, to the prosody and emotive content of the words, and in music, to the melodic structure of the line. Pitch extraction is similar to, but more difficult than, the problem of *fundamental frequency* extraction; the term *pitch* refers to a psychological construct that may be present even in signals missing a fundamental frequency. Pitch tracks are clearly a nonsynthesizable representation for sound. If all that is preserved of a signal is a stream of numbers corresponding to samples of the pitch, we cannot use the pitch track to reconstruct the original signal without a great deal more information.

Pitch trackers typically operate in the time domain, in time-frequency space, or in various lag spaces, such as autocorrelation or average-magnitude difference function space. Various implementations span a great deal of engineering territory, from very fast, low-latency, relatively inaccurate systems appropriate for transducing an acoustic musical performance into a synthetic one in real time [50] to highly accurate models of the human perceptual system, which take more time to calculate [67].

There have been recent attempts to build systems capable of extracting rhythmic information from musical signals. Vercoe [115] described the use of the *narrowed autocorrelation* [12] for beat extraction. The rhythmogram [111] is a multiscale representation that has been used to analyze monophonic music and speech and give a visual representation of the rhythmic content. Goto [41] built a system for analysis of popular music; his system extracts the timings of the drumbeats and looks them up in a data base of patterns; his system not only can determine the beat and tempo but also can find the strong and weak beats of a musical pattern. Scheirer [94] has described a highly parallel algorithm, with a vocoder-like frequency decomposition and banks of parallel comb-filter resonators, which is claimed to be very robust for determining the tempo and beat in a broad range of musical situations.

There have been some attempts to build systems that can automatically recognize and classify musical timbre, at least in a monophonic (one-note-at-a-time) setting. The study of musical timbre is still a fairly ill-defined area; there is not even an agreed-upon extensional definition of the term "timbre." Some researchers have attempted to use dynamic spectrum analyses, i.e., the STFT [1], [2], while others try to reduce the dimensionality of the problem using principal components analysis [109] or homomorphic deconvolution [11]. Further, some of the parameter-estimation methods described above are applicable to instrument recognition if they are used with a goodness-of-fit measure. In any case, a system that could identify instruments robustly, particularly in the presence of other interfering instruments, would be a useful addition to a suite of musical tools. This recognition capability does not yet exist.

3) Signal Classification: There have been a few attempts to use statistical methods to classify whole signals by type. Speech-music discrimination has been approached several times, using a variety of features and classification methods. The system claiming the best results [98] used 13 spectral and temporal features and examined four multidimensional classifiers, resulting in up to 98% discrimination performance in real time over a large test set of data. Foote [33] used cepstral frames (homomorphic signal analysis and data reduction) and a branching classifier to distinguish musical genres from one another for a restricted set of test cases.

C. Computational Auditory Scene Analysis (CASA)

An important recent development in the study of sound-analysis methods is the new field of CASA. Named as the computational analogue to Bregman's seminal psychoacoustic work on auditory scenes and auditory streaming [10], CASA organizes research in a number of historically separate areas into a common framework. The various topics addressed include *automatic transcription*, which is the attempt to reconstruct a "score" from an audio waveform, and the so-called "cocktail party" problem, which is the attempt to recognize speech in the presence of competing sounds.

There is an important relationship between CASA research and the structured-audio approach to sound representation. In many cases, CASA systems can be viewed as the analysis component corresponding to structured sound synthesis. A multisource structured soundtrack that combines synthetic music, speech voice-over, and sound effects, then applies artificial reverberation and spatial processing, results in a multichannel set of mixed audio waveforms. CASA methods attempt to separate and/or understand the component sounds of the waveforms by grouping together pieces of the sound that are perceived as belonging together. The use of CASA systems to act as a fully automatic "structured-audio encoder" is still far in the future, however, as today's state-of-the-art CASA models are not yet capable of doing such sophisticated processing.

As with parameter estimation, the bulk of recent CASA research has centered in the speech community. Many systems have been built to attempt to solve the *concurrent vowel* problem, where vowel sounds from two different speakers are isolated and recognized. A wide variety of approaches have been taken, including using pitch trackers [121], grouping sinusoidal "tracks" from a phase-vocoder representation [28], and using coupled-oscillator models [119]. Recently, approaches have been taken that incorporate more high-level linguistic information to guide the models [74] or incorporate binaural information [58], [73], attempting to utilize spatial cues. The best systems today can provide somewhat accurate performance in highly constrained test circumstances but not yet anything close to robust separation in general environments.

Researchers involved in this sort of CASA work typically pay close attention to results from the psychoacoustic and perceptual study of human audition. When new discoveries are made in these areas, they are used to guide the construction of new models [22]. The construction of real-time models is not yet a concern, as achieving best performance is paramount; many such separation models take hundreds of times longer than real time on a fast modern computer to process sounds.

In contrast, research into *blind source separation* attempts to use statistical measurements to separate signals encoded into multiple channels of sound [3], [8], [104]. That is, if multiple microphones are placed in a room, the cross correlations in the signal can be used to separate the various input sounds. These techniques may run very quickly and may be more accurate than human perception; on the other hand, they are not appropriate for circumstances in which there are more signals to be separated than channels of audio available. They have a great potential for automatic structured-scene encoding.

An important area of research within CASA is the musical scene analysis, or *automatic-transcription*, problem. Automatic-transcription systems attempt to reverse the musical-performance process to recover an event-list description of a piece of music from an acoustic performance. Early attempts at transcription placed a large number of constraints on the music that could be analyzed [71], especially the number, timbre, and harmonic relationship of the voices. Later work relaxed the harmonic relationships that could be included [60] and included a more perceptually motivated analysis [62], [68]. Most recently, transcription systems have attempted to use high-level constraints, domain information, or rule-based systems to guide the acoustic analysis [49], [63], [96].

Even with this relatively long historical evolution, automatic-transcription systems are still very rudimentary compared to the problem at hand. It is unlikely that robust systems capable of translating expressive acoustic performances in genres like rock and roll, jazz, or classical music into high-quality MIDI representations will be built within the next decade. The state-of-the-art systems cannot yet transcribe music with more than three or four simple voices and cannot be applied to music with unknown timbres.

As well as representations applicable to speech sounds and musical sounds, there has been a small amount of work on the classification of noisy, quasi-periodic, and impulsive sounds. Saint-Arnaud and Popat used the Wold decomposition to attempt the separation of sounds into periodic and stochastic components [91]; this approach is similar to the "deterministic + stochastic" model of Serra [102], which used direct spectral analysis. The Saint-Arnaud and Popat system was used to classify (and resynthesize, although classification was the goal) machine sounds and other "sound textures." Ellis [29] built a large system using an intermediate model of the human auditory and perceptual grouping system, which could, to some extent, isolate sound effects and noises in complex auditory scenes. His system is based on a novel sound representation called the *weft*, an extension of the autocorrelation model. It also includes representations of noise textures and impulsive sounds.

A number of audio-perception researchers have reported that human listeners seem to be able accurately to distinguish characteristics of the excitation method from characteristics of the affected object in the perception of transient sounds such as impacts, fractures, and "bouncing and breaking" sounds [119]. There have been recent attempts [15] to apply this sort of decompositional approach to automatic perceptual analysis of transient sounds.

IV. Applications

In this section, we will describe some applications for structured-audio techniques. We will juxtapose the use of structured-sound descriptions with use of traditional audio-coding schemes, contrasting the relative advantages of each. References are provided where we are aware of implementations of systems similar to what we describe; in some cases, the description is still a speculative one.

These applications fall into two broad groups. In one, the advantage of using structured-audio techniques is due to their efficiency or conciseness; in the other, the advantage is due to flexibility or the availability of perceptual hooks for searching or modifying signals.

A. Low-Bandwidth Transmission

Many of the synthesis and analysis descriptions in Sections II and III have been described with regard to ultralow-bit-rate compression. This application is a clear area in which structured-audio techniques provide impressive gains over even the best perceptual coding schemes for waveforms. Structured coding for low-bit-rate transmission can be viewed as the elimination of structural redundancy, as described in Section I-B, by using algorithmic, scene-based, and event-list descriptions of sound.

When implementing low-bandwidth coding with structured audio, we are typically trading off transmission bandwidth for computation on the client side. That is, we transmit a structured description and dynamically render it into sound on the client side rather than rendering in a studio on the server side. As certain sound-processing algorithms (physical modeling synthesis, 3-D spatialization) are quite expensive computationally, only situations in which adequate horsepower is available at the client for rendering are appropriate for this form of transmission. Additionally, traditional coding forms have the advantage that we may estimate the amount of computation required for decoding and rendering by examining the bit stream. With structured methods, this may be difficult, and is uncomputable in situations where the representation is sufficiently powerful to represent arbitrary algorithms.

It is interesting to think about the relationship between structured-audio representations, especially algorithmic ones like synthesis languages and the MPEG-4 standard, and the Kolmogorov complexity of sound waveforms [54]. In theoretical computer science, the Kolmogorov complexity of a string is the length of the shortest program that, when executed, produces that string as output. For truly random strings, entropic coding is typically near the Kolmogorov limit, but for algorithmically generated strings, an algorithm-based approach must be more efficient in the long run if the underlying algorithm is discovered. Rissanen's minimum description length (MDL) principle [81] has similar implications—that description length is composed of both the parametric size of a representation and the size of the algorithm needed to decode or understand the representation. Thus, we may view algorithmic encoding methods as using MDL models in conjunction with perceptual limits.

B. Sound Generation from Process Models

Structured-audio representations typically provide better parameters for high-level control of sound than do traditional waveform representations. This means that in applications where we have a *process model* governing the computation, it is generally easier to couple the sound representation tightly to the process model if the sound representation is structured. By "process model," we mean that the sound is not created from an event list but rather is dynamically generated in response to evolving, nonsound-oriented computation.

Video games fall into this category; in fact, the video-game domain has naturally embraced structured-audio concepts for many years. The sound effects for a video game cannot be scored, as they must be scheduled in response to user interaction. Most recent games have used a wavetable approach to sound effects; that is, artists create sounds in a studio using a process not unlike Foley for filmmaking, and the sounds are triggered by sample playback as they are needed. Few games today can afford to allocate computational resources to sound synthesis that would be required to use more sophisticated techniques. Such techniques might include having source/excitation models for sound ("a steel bullet, traveling at 300 m/s, impacts a concrete wall") or otherwise creating an object-oriented model of sonic properties. If future hardware-based audio systems that include more advanced structured capability become widely available, some of these techniques might be explored.

Some video games include musical soundtracks, and various structured and nonstructured approaches have been taken to presenting them. Many games include MIDI file specifications for each section or level of play; as the players advance to a new level, they are presented with a new musical theme as background. The use of MIDI files serves as a form of structured compression, because only the event data must be stored; many PC-based sound cards provide the ability to synthesize them into audio in hardware. The musical and timbral quality of such backgrounds has often been quite low, however, and many video-game players often simply shut the music off.

More recently, the increase in storage provided by CD-ROM-based video games has allowed the sound designer to include custom wavetable instruments, or even full waveform audio soundtracks, with a game, improving the musical quality. Last, certain games have attempted to create dynamic soundtrack algorithms, where the background music actually changes in response to the moment-to-moment action in the game [56]; that is, when the game activity is faster, the music will become more agitated or otherwise change to reflect and influence the mood. Such algorithms are very difficult to create, bearing a close relationship with the body of research on computer composition [76].

A second form of process model, sometimes related to gaming, is in virtual-reality applications. To create the illusion of virtual space, sound systems must reverberate and spatialize the sounds present in the virtual environment, and these effects must be dynamic in response to listener interaction and movement [37]. The effects cannot be "mixed into" a soundtrack in the studio, because the particular parameters to be used are not known until it is time to present the sound to the listener. Thus, structured models for sound that can be controlled parametrically are an essential component in sound systems for virtual environments.

C. Flexible Music Synthesis

The currently prevalent MIDI standard leads to many problems for composers who use it for creating and disseminating their work. The use of fixed hardware/software synthesizers that differ in sound quality makes it difficult for composers to know what their pieces will sound like on any particular listener's equipment when the music is transmitted using MIDI. Current end-user synthesis systems—particularly the most common, General-MIDI-based wavetable systems—allow for very little expressivity in performance or sophisticated control of timbre. High-end physical modeling synthesizers that do have this expressivity are very difficult to control—it is difficult to design and build physical interfaces that are easy to use and yet give the desired degree of control.

Structured-audio encoding methods based on synthesis languages, like NetSound [14] and MPEG-4 Structured Audio [97], alleviate the unknown and unsophisticated nature of fixed synthesizers. Because the algorithms for synthesis are created and transmitted by the composer along with the event descriptions, any system that can understand and render the structured audio encoding will use the same algorithms to generate the same sound. When new, more sophisticated synthesis methods are invented, they can be immediately incorporated in a composition rather than being dependent on the hardware-development life cycle for distribution.

Structured-audio systems based on music-synthesis languages can also represent synthesis methods with very expressive nuance and yet allow the use of simple controls by composers. By using the synthesis language to represent not only the instrument but also a synthetic performer who "knows about" the timbral qualities of the instrument, the composer can specify only the notes or chords to be played and perhaps a small number of parameters that control the virtual performer. The synthetic performer, then, using the notes given by the composer, produces control parameters for the instrument to generate the sound [21], [39].

This method is similar to the method commonly used by pop-music composers who incorporate drum machines into their work. However, using a structured audio language, the algorithms embodied in the "drum machine" or other virtual performer may be very sophisticated. As with synthesis, when new methods are discovered for improving the musical "feel" of an algorithm [9], composers/programmers may immediately incorporate them into their compositions. The relationship between the composer and synthesizer becomes similar to the traditional composer–performer relationship. The composer delegates the final decisions about notes, timing, and control to the performer in order to focus more clearly on high-level issues.

D. Interactive Music Applications

As well as allowing for flexible, controllable music synthesis, structured sound methods are typically used in interactive music performance systems. By *interactive performance,* we refer to a process in which a computer system listens to a human performer or performers and adjusts its own performance in some way in response. Thus, these systems utilize both "structure from sound" techniques, when they listen to the human performers, and "sound from structure" techniques, when they produce new sounds in response to their current model of the musical situation.

Some of the earliest work of this sort was on *automatic accompaniment;* that is, the use of computer accompanists in a "music-minus-one" framework, where the computer played an accompaniment to the human part. Early systems used the computer only to play back recorded music, as a tape would, but soon gave way to truly interactive systems that could follow the human performer [114] and learn from experience in rehearsal [117]. More recent systems have included compositional sophistication in the computer algorithms [90] and have sometimes grown to enormous proportions [59].

Algorithmic structured audio techniques have a great deal of potential in the karaoke market, as the algorithms for sensing and following the performer may be specified in conjunction with the synthesis methods. Structured-audio representations that allow for the real-time construction and control of structural models (like tempo, phrase, and so on) might be used to allow karaoke-like systems to adjust themselves in myriad ways to the human performer.

E. Waveform Manipulation

The application areas described in the preceding paragraphs make use of structured-audio representations to control synthesis. However, there are also compelling applications for the analysis and extraction of structured-sound descriptions.

There are many tools available to assist sound designers, editors, and composers in working with waveform audio. However, most of these tools are deaf with regard to the content of the waveform being analyzed. If capabilities for extracting structured information are added to such tools, they could be much more powerful and user friendly. For example, using timbre recognition (technology yet to be developed), a soundtrack editor could instruct the program to "skip to where the violins come in." Advanced auditory scene-analysis software would allow a composer to turn an existing sound clip into symbolic data such as MIDI for manipulation, or perhaps even to remove or amplify individual instruments from an audio mix.

If a waveform is annotated with symbolic representations of its musical content, these symbols can be used for alignment and synchronization. For example, Scheirer [94] describes using a beat-tracking algorithm to locate points in a song that are aligned with a drum track using time-stretching techniques. Standards such as SMPTE time codes already provide interfaces for a wide range of tools to schedule and synchronize each other and are commonly used in the sound-editing industry; it is easy to imagine other ways that such markers could be associated with structured representations.

Dynamic waveform manipulation may itself become part of a real-time system if enough information is extracted from the signal to allow accurate control. We have used the music-synthesis language Csound [116] to dynamically harmonize singers in real time: first, an input signal is pitch tracked so that we know the starting point and extract the glottal pulse from the waveform as a model of the vowel and vocal timbre. Then we resynthesize new singers at harmonically related pitches specified through real-time keyboard control. The effect upon mixing these sounds together is that of a chorus of the same voice's singing in harmony. To accomplish this, we are dynamically building, and then using, a structured representation of the original singer.

F. Content-Based Retrieval

Content-based multimedia retrieval, also known as *query by image content* for images and *query by audio content* (QBAC) for sound, is the process of accessing a multimedia data base by automatically analyzing and categorizing the digital content of the entries. There have been many systems built for this purpose in the visual domain [30], [75], with varying degrees of success, but relatively few attempts in the audio domain—see [123] for one suggested framework.

There are many compelling applications for QBAC systems. There have been several companies started recently to provide Web-based "intelligent agent" systems, which try to suggest songs, musicians, or albums (as well as movies, foods, etc.) that their users might enjoy [31]. Using QBAC technology, such a system could make recommendations based on its own assessment of the musical content rather than only on statistical analysis of user communities, which is how such systems work today.

"Query by humming" systems [55] allow users to access a data base of songs by humming or whistling the melody to a computer interface. Typically, the data base is hand entered or created from symbolic MIDI data; with sufficiently powerful automatic-transcription technology, the data base could be created automatically from acoustic data. A data base of automatically annotated performances would additionally be an extremely useful tool for musicologists and other researches into human-performance practice. Some of the transcription and description analysis tools described in Section III have been used for this purpose [96].

Others have built systems [34] that use speech recognition to aid searching of video segments. Analogous systems could be built for searching video segments based on musical content or sound effects—this is particularly useful in the case where the action sought occurs off-screen.

V. Conclusion

There are no systems yet implemented that integrate the variety of techniques we have described in a flexible real-time package. Clearly, to make possible many of the scenarios we described earlier, a broadly distributed framework for structured audio is necessary. We have built two prototype systems: one that allows a certain amount of capability in a real-time implementation (NetSound) and one with broader capability that has not yet been implemented efficiently (the verification model of the MPEG-4 structured-audio standard). It is our hope that by including powerful structured capability in a respected international standard, more companies will choose to build systems containing this sort of functionality.

Existing standards for audio transmission such as MPEG, MPEG-2, and ITU-T became most practically useful once dedicated hardware was available to decode and play back downloaded audio streams. For the MPEG-4 structured-audio standard, the situation will likely be no different. The capabilities described by MPEG-4 are just at the current state of the art in many areas, so it may be several years after standardization before fully functional hardware implementations are available. However, the language and capabilities have been designed with efficient hardware implementation in mind.

There is a difficult issue in the development of new systems for representation and transmission of multimedia; namely, the chicken-and-egg problem of content and implementations. If content is to be created in a new (structured) format, there must be authoring tools and decoding/rendering tools available for the authoring and playback process; however, this implies that those tools must be provided in the absence of existing content. It is a difficult question whether to make an expenditure in time and resources on either side of this coin—authoring content in formats for which there are few playback tools or creating playback tools for formats in which little content is available.

The standardization process helps to solve this problem by allowing industrial interests to come together to decide on a technical solution and proceed together toward new tools, implementations, and content. However, the transmission of truly structured and scene-based audio description requires rather large changes at many points of the modern content-delivery path in authoring, production, transmission, broadcast, and playback.

In this paper, we have discussed a wide variety of techniques for algorithmic representation, annotation, transmission, and synthesis of sound in structured formats. Using these techniques, future sound hardware will allow simple, flexible access to composers and sound designers for the efficient control of sound, and future multimedia data bases will be searchable by parameters to sound content. We have tied together a number of typically separate domains of inquiry into a unified framework to demonstrate the shared viewpoint that underlies them all. Last, we have described standards at and beyond the state of the art that unify structured-audio methods.

Future work on structured audio will use these representations for sound in many applications. New algorithms for creating and processing sound will be developed and efficient hardware implementations of these techniques constructed.

References

[1] J. B. Allen, "Short term spectral analysis, synthesis, and modification by discrete Fourier transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-25, pp. 235–238, Mar. 1977.

[2] J. B. Allen and L. R. Rabiner, "A unified approach to short-time Fourier analysis and synthesis," *Proc. IEEE*, vol. 65, pp. 1558–1564, Nov. 1977.

[3] S.-I. Amari, A. Cichocki, and H. H. Yang, "A new learning algorithm for blind signal separation," in *Advances in Neural Information Processing Systems 8*. Cambridge, MA: MIT Press, 1996.

[4] B. S. Atal, "Automatic speaker recognition based on pitch contours," *J. Acoust. Soc. Amer.*, vol. 52, pp. 1687–1697, 1972.

[5] B. S. Atal and S. L. Hanauer, "Speech analysis and synthesis by linear prediction of the speech wave," *J. Acoust. Soc. Amer.*, vol. 50, pp. 637–655, 1971.

[6] B. S. Atal and M. R. Schroeder, "Predictive coding of speech signals and subjective error criteria," *IEEE Trans. Acoust., Speech, Signal Proc.*, vol. ASSP-27, pp. 247–254, 1979.

[7] J. Beauchamp, "Brass tone synthesis by spectrum evolution matching with nonlinear functions," *Comput. Music J.*, vol. 3, no. 2, pp. 35–43, Summer 1979.

[8] A. J. Bell and T. Sejnowski, "An information maximization approach to blind separation and blind deconvolution," *Neural Computation*, 1989, pp. 1129-1159.

[9] J. Bilmes, "Timing is of the essence: Perceptual and computational techniques for representing, learning, and reproducing expressive timing in percussive music," S.M. thesis, Massachusetts Institute of Technology Media Lab, Cambridge, 1993.

[10] A. Bregman, *Auditory Scene Analysis*. Cambridge, MA: MIT Press, 1989.

[11] J. Brown, "Computer identification of musical instruments using pattern recognition," presented at the Society for Music Perception and Cognition Meeting, Cambridge, MA, 1997.

[12] J. Brown and M. Puckette, "Calculation of a 'narrowed' autocorrelation function," *J. Acoust. Soc. Amer.*, vol. 85, pp. 1595–1601, 1989.

[13] M. A. Casey, "Understanding musical sound with forward models and physical models," *Connection Sci.*, vol. 6, pp. 355–371, 1994.

[14] M. A. Casey and P. Smaragdis, "Netsound," in *Proc. 1996 Int. Computer Music Conf.*, Hong Kong, P.R.C., 1996.

[15] M. A. Casey, Auditory group theory with applications to statistical basis methods for structured audio, Ph.D. dissertation, Massachusetts Institute of Technology Media Lab, Cambridge, 1997.

[16] J. Cahn, "Generating expression in synthesized speech," M.S. thesis, Massachusetts Institute of Technology Media Lab, Cambridge, 1990.

[17] J. Chowning, "The synthesis of complex audio spectra by means of frequency modulation," *J. Audio Eng. Soc.*, July 1977, pp. 526–534.

[18] P. R. Cook, "Identification of control parameters in an articulatory vocal tract model, with applications to the synthesis of singing," Ph.D. dissertation, CCRMA, Stanford Univ., Stanford, CA, 1991.

[19] ——, "TBone: an interactive waveguide brass instrument synthesis workbench for the next machine," in *Proc. Int. Computer Music Conf.*, Montreal, Canada, 1991, pp. 297–299.

[20] ——, "SPASM, a real-time vocal tract physical model controller; and Singer, the companion software synthesis system," *Comput. Music J.*, Spring 1993, pp. 30–44.

[21] P. R. Cook, "A hierarchical system for controlling synthesis by physical modeling," in *Proc. Int. Computer Music Conf.*, 1995, Banff, CA, pp. 108–109.

[22] M. Cooke, *Modeling Auditory Processing and Organization*. Cambridge, U.K.: Cambridge Univ. Press, 1993.

[23] G. De Poli and A. Piccialli, "Pitch-synchronous granular synthesis," in *Representations of Musical Signals*, G. De Poli, A. Piccialli, and C. Roads, Eds. Cambridge, MA: MIT Press, 1991, pp. 187–220.

[24] G. De Poli, A. Piccialli, and C. Roads, Eds., *Representations of Musical Signals*. Cambridge, MA: MIT Press, 1991.

[25] G. Doddington, "Speaker recognition—Identifying people from their voices," *Proc. IEEE*, vol. 83, pp. 1651–1664, Nov. 1985.

[26] M. B. Dolson, “The phase vocoder: A tutorial,” *Comput. Music J.*, Winter 1986, pp. 14–27.
[27] H. Dudley, “Remaking speech,” *J. Acoust. Soc. Amer.*, vol. 11, pp. 169–177, 1939.
[28] D. P. W. Ellis, “A computer implementation of psychacoustic grouping rules,” in *Proc. 12th Int. Conf. Pattern Recognition*, Jerusalem, Israel, Oct. 1994.
[29] ——, “Prediction-driven computational auditory scene analysis,” Ph.D. dissertation, Dept. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, 1996.
[30] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz, “Efficient and effective querying by image content.” *J. Intell. Inform. Syst.*, 1994, pp. 231–262.
[31] Firefly Network, Inc. (1997). WWW white paper. [Online]. Available: http://www.firefly.net/products/FireflyTools.html.
[32] J.-L. Florens and C. Cadoz, “The physical model: Modeling and simulating the instrumental universe,” in *Representations of Musical Signals*, G. De Poli, A. Picccialli, and C. Roads, Eds. Cambridge, MA: MIT Press, 1991, pp. 227–268.
[33] J. Foote, “A similarity measure for audio classification,” in *Proc. AAAI 1997 Spring Symp. Intelligent Integration and Use of Text, Image, Video, and Audio Corpora*, Stanford, CA, Mar. 1997.
[34] J. T. Foote, M. G. Brown, G. J. F. Jones, K. Spdrck Jones, and S. J. Young, “Video mail retrieval by voice: Toward intelligent retrieval and browsing of multimedia documents,” in *Proc. IMMI-1, 1st Int. Workshop Intelligence and Multimodality in Multimedia Interfaces*, Edinburgh, Scotland, July 1995.
[35] M. D. Freedman, “Analysis of musical instrument tones,” *J. Acoust. Soc. Amer.*, vol. 41, pp. 793–806, 1967.
[36] W. G. Gardner, “Reverberation algorithms,” in *Applications of Signal Processing to Audio and Acoustics*, M. Kahrs and K. Brandenberg, Eds. New York: Kluwer, 1998.
[37] ——, “3-D audio using loudspeakers,” Ph.D. dissertation, Massachusetts of Technology Media Lab, Cambridge, 1997.
[38] G. E. Garnett, “Music, signals, and representations: A survey,” in *Representations of Musical Signals*, G. De Poli, A. Picccialli, and C. Roads, Eds. Cambridge, MA: MIT Press, 1991, pp. 325–370.
[39] B. Garton, “Virtual performance modeling,” in *Proc. Int. Computer Music Conf.*, 1992, pp. 219–222.
[40] A. Gersho, “Advances in speech and audio compression,” *Proc. IEEE*, vol. 82, pp. 900–918, June 1994.
[41] M. Goto and Y. Muraoka, “Music understanding at the beat level: Real-time beat tracking for audio signals,” in *Readings in Computational Auditory Scene Analysis*, H. Okuno and D. Rosenthal, Eds. Mahweh, NJ: Erlbaum, 1998.
[42] J. M. Grey, “Multidimensional perceptual scaling of musical timbres,” *J. Acoust. Soc. Amer.*, vol. 61, pp. 1270–1277, 1977.
[43] J. M. Grey and J. A. Moorer, “Perceptual evaluations of synthesized musical instrument tones,” *J. Acoust. Soc. Amer.*, vol. 62, pp. 454–462, 1977.
[44] B. Grill, B. Edler, M. Hahn, K. Iijima, N. Iwakami, T. Moriya, J. Ostermann, S. Nakajima, M. Nishiguchi, T. Nomura, W. Oomen, H. Purnhagen, E. Scheirer, N. Tanaka, A. P. Tan, and R. Taori, Eds. “MPEG-4 audio Working Draft 4.0,” ISO/IEC JTC1/SC29/WG11 (MPEG) Bristol Document N1631, 1997.
[45] A. Horner, B. Beauchamp, and L. Haken, “FM matching synthesis with genetic algorithms,” *Comput. Music J.*, vol. 17, no. 4, pp. 17–29, Winter 1993.
[46] N. Jayant, J. Johnston, and R. Safranek, “Signal compression based on models of human perception,” *Proc. IEEE*, vol. 81, pp. 1385–1421, Oct. 1993.
[47] J. M. Jot, “Synthesizing three-dimensional sound scenes in audio or multimedia production and interactive human-computer interfaces,” presented at the Fifth International Conference on Interfaces to Real and Virtual Worlds, 1996.
[48] M. Karjalainen, U. K. Laine, T. Laakso, and V. Valimaki, “Transmission-line modeling and real-time synthesis of string and wind instruments,” in *Proc. Int. Computer Music Conf.*, San Francisco, CA, 1991, pp. 293–296.
[49] K. Kashino, K. Nakadai, T. Kinoshita, and H. Tanaka, “Application of Bayesian probability network to music scene analysis,” in *Readings in Computational Auditory Scene Analysis*, H. Okuno and D. Rosenthal, Eds. Mahweh, NJ: Erlbaum, 1998.
[50] W. B. Kuhn, “A real-time pitch recognition algorithm for music applications,” *Comput. Music J.*, pp. 60–71, Fall 1990.
[51] R. G. Laughlin, B. D. Truax, and B. V. Funt, “Synthesis of acoustic timbres using principal component analysis,” in *Proc. Int. Computer Music Conf.*, Glasgow, Scotland, 1990, pp. 95–99.
[52] M. LeBrun, “A derivation of the spectrum of FM with a complex modulating wave,” *Comput. Music J.*, pp. 51–52, Winter 1977.
[53] ——, “Digital waveshaping synthesis,” *J. Audio Eng. Soc.*, pp. 250–266, Apr. 1979.
[54] M. Li and P. M. B. Vitanyi, *An Introduction to Kolmogorov Complexity and Its Applications*. New York: Springer-Verlag, 1993.
[55] A. Lindsay, “Using contour as a mid-level representation of melody,” M.S. thesis, Massachusetts Institute of Technology Media Lab, Cambridge, 1996.
[56] ——, “Multimodal interaction in the perception of musical interpolation,” presented at the Society for Music Perception and Cognition Meeting, Cambridge, MA, 1997.
[57] D. G. Loy, “Musicians make a standard: the MIDI phenomenon,” *Comput. Music J.*, pp. 181–198, Winter 1989.
[58] R. F. Lyon, “Computational models of neural auditory processing,” in *Proc. IEEE ICASSP*, 1984.
[59] T. Machover. (1997). The brain opera archive. [Online]. Available: http://brainop.media.mit.edu/Archive/index.html.
[60] R. Maher, “An approach for the separation of voices in composite musical signals,” Ph.D. dissertation, Univ. of Illinois Urbana-Champaign, 1989.
[61] J. Makhoul, “Linear prediction: a tutorial review,” *Proc. IEEE*, vol. 63, pp. 561–580, Mar. 1975.
[62] K. Martin, “Automatic transcription of simple polyphonic music: Robust front end processing,” Massachusetts Institute of Technology Media Lab, Cambridge, Perceptual Computing Tech. Rep. 399, 1996.
[63] K. Martin and E. D. Scheirer, “Automatic transcription of simple polyphonic music: Incorporating high-level knowledge,” presented at the Society for Music Perception and Cognition Conference, 1997.
[64] D. C. Massie, “Wavetable Sampling Synthesis,” in *Applications of Signal Processing to Audio and Acoustics*, M. Kahrl and K. Brandenberg, Eds. New York: Kluwer, to be published.
[65] M. Mathews, *The Technology of Computer Music*. Cambridge, MA: MIT Press, 1969.
[66] R. J. McAulay and T. F. Quatieri, “Speech analysis/synthesis based on a sinusoidal representation,” *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, 1986, pp. 744–754.
[67] R. Meddis and M. Hewitt, “Virtual pitch and phase sensitivity of a computer model of the auditory periphery. I: Pitch identification,” *J. Acoust. Soc. Amer.*, vol. 89, pp. 2866–2882, 1991.
[68] D. Mellinger, “Event formation and separation in musical sound,” Ph.D. dissertation, CCRMA Stanford Univ., Stanford, CA, 1991.
[69] R. Moog, “MIDI: Musical Instrument Digital Interface,” *J. Audio Eng. Soc.*, May 1986, pp. 394–404.
[70] F. R. Moore, “The dysfunctions of MIDI,” *Comput. Music J.*, pp. 19–28, Spring 1988.
[71] J. A. Moorer, “On the segmentation and analysis of continuous musical sound by digital computer,” Ph.D. dissertation, CCRMA, Stanford Univ., Stanford, CA, 1975.
[72] ——, “The use of the phase vocoder in computer music applications,” *J. Audio Eng. Soc.*, pp. 42–45, Feb. 1978.
[73] T. Nakatani, M. Goto, T. Ito, and H. Okuno, “Multi-agent based binaural sound stream segregation,” in *Readings in Computational Auditory Scene Analysis*, H. Okuno and D. Rosenthal, Eds. Mahweh, NJ: Erlbaum, 1998..
[74] S. H. Nawab, C. Espy-Wilson, R. Mani, and N. Bitar, “Knowledge-based analysis of speech mixed with sporadic environmental sounds” in *Readings in Computational Auditory Scene Analysis*, H. Okuno and D. Rosenthal, Eds. Mahweh, NJ: Erlbaum, 1998.
[75] R. Picard, “A society of models for video and image libraries,” *IBM Syst. J.*, vol. 35, pp. 292–312, 1996.
[76] S. T. Pope, “Music composition and scoring by computer,” in *Music Processing*, G. Haus, Ed. Los Angeles, CA: A-R Editions, 1992.
[77] M. Puckette, “Combining event and signal processing in the

MAX graphical programming environment," *Comput. Music J.*, pp. 68–77, Fall 1991.

[78] L. R. Rabiner, "Speech synthesis by rule: an acoustic domain approach," *Bell Syst. Tech. J.*, pp. 17–37, 1968.

[79] ——, "Digital formant synthesizer for speech synthesis studies," *J. Acoust. Soc. Amer.*, vol. 43, pp. 822–828, 1968.

[80] L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals.* Englewood Cliffs, NJ: Prentice-Hall, 1978.

[81] J. Rissanen, "Modeling by the shortest data description," *Automatica*, pp. 465–471, 1978.

[82] J.-C. Risset and M. V. Mathews, "Analysis of musical-instrument tones," *Phys. Today*, pp. 23–30, 1969.

[83] J.-C. Risset and D. L. Wessel, "Exploration of Timbre by analysis and synthesis," in D. Deutsch, Ed., *Psychology of Music*. Orlando, FL: Academic, 1982.

[84] C. Roads, "A tutorial on nonlinear distortion or waveshaping synthesis," *Comput. Music J.*, pp. 29–34, Summer 1979.

[85] ——, "Granular Synthesis of Sound," in *Foundations of Computer Music*, C. Roads and J. Strawn, Eds. Cambridge, MA: MIT Press, 1985, pp. 145–159.

[86] ——, "Asynchronous Granular Synthesis," in *Representations of Musical Signals*, G. De Poli, A. Piccialli, and C. Roads, Eds. Cambridge, MA: MIT Press, 1991, pp. 143–186.

[87] ——, *The Computer Music Tutorial.* Cambridge, MA: MIT Press, 1996.

[88] X. Rodet, "Time-domain formant wave-function synthesis," *Comput. Music J.*, pp. 9–14, Fall 1984.

[89] D. Rossum, "The SoundFont 2.0 file format," Joint E-Mu/Creative Tech Center white paper, 1995.

[90] R. Rowe, *Interactive Music Systems*. Cambridge, MA: MIT Press, 1993.

[91] N. Saint-Arnaud and K. Popat, "Analysis and synthesis of sound textures," in *Readings in Computational Auditory Scene Analysis*, H. Okuno and D. Rosenthal, Eds. Mahweh, NJ: Erlbaum, 1998.

[92] G. J. Sandell and W. L. Martens, "Perceptual evaluation of principal-component-based synthesis of musical timbres," *J. Audio Eng. Soc.*, pp. 1013–1028, Dec. 1995.

[93] C. Scaletti and K. Hebel, "An object-based representation for digital audio signals," in *Representations of Musical Signals*, G. De Poli, A. Piccialli, and C. Roads, Eds. Cambridge, MA: MIT Press, 1991, pp. 371–390.

[94] E. D. Scheirer, "Tempo and beat analysis of acoustic musical signals," *J. Acoust. Soc. Amer.* vol. 103, pp. 588–601, Jan. 1998.

[95] ——, "The MPEG-4 structured audio standard," in *Proc. IEEE ICASSP*, 1998.

[96] ——, "Using musical knowledge to extract expressive performance information from audio recordings," in *Readings in Computational Auditory Scene Analysis*, H. Okuno and D. Rosenthal, Eds. Mahweh, NJ: Erlbaum, 1998.

[97] ——, "Structured audio and effects processing in the MPEG-4 multimedia standard," *ACM Multimedia Syst.*, to be published.

[98] E. D. Scheirer and M. Slaney, "Construction and evaluation of a robust multifeature speech-music discriminator," in *Proc. IEEE ICASSP*, 1997.

[99] K. R. Scherer, "Vocal affect expression: A review and a model for future research," *Psych. Bul.*, pp. 143–165, 1986.

[100] B. Schottstaedt, "The simulation of natural instrument tones using frequency modulation with a complex modulating wave," *Comput. Music J.*, pp. 46–50, Winter 1977.

[101] E. Selfridge-Field, Ed., *Beyond MIDI: The Handbook of Musical Codes.* Cambridge, MA: MIT Press, 1997.

[102] X. Serra, "A system for sound analysis/transformation/synthesis based on a deterministic plus stochastic decomposition," Ph.D. dissertation, CCRMA, Stanford Univ., Stanford, CA, 1989.

[103] C. Shannon, "A mathematical theory of communications," *Bell Syst. Tech. J.*, pp. 379–423, 623–656, 1948.

[104] P. Smaragdis, "Information theoretic approaches to source separation," S.M. thesis, Massachusetts Institute of Technology Media Lab, Cambridge, 1997.

[105] R. N. Shepard, "The analysis of proximities: Multidimensional scaling with an unknown distance function," *Psychometrika*, pp. 125–140, 1962.

[106] J. O. Smith, "Techniques for digital filter design and system identification with application to the violin," Ph.D. dissertation, CCRMA, Stanford Univ., Stanford, CA, 1983.

[107] ——, "Musical applications of digital waveguides," Music Department, Stanford Univ., Stanford, CA, Rep. STAN-M-39, 1987.

[108] ——, "Physical modeling using digital waveguides," *Comput. Music J.*, pp. 74–9, Winter 19921.

[109] J. Stapleton and S. C. Bass, "Synthesis of musical tones based on the Karhunen-Loève transform," *IEEE Trans. Acoust., Speech, Signal Processing*, Mar. 1988, pp. 305–319.

[110] W. Strong and M. Clark, "Synthesis of wind-instrument tones," *J. Acoust. Soc. Amer.*, vol. 41, pp. 39–52, 1967.

[111] N. P. Todd, "The auditory 'primal sketch': A multiscale model of rhythmic grouping," *J. New Music Res.*, pp. 25–70, 1994.

[112] S. Van Duyne and J. O. Smith, "Physical modeling with the 2-D digital waveguide mesh," in *Proc. 1993 Int. Computer Music Conf.*, pp. 40–47.

[113] V. Valimaki, M. Karjalainen, and T. I. Laakso, "Modeling of woodwind bores with finger holes," in *Proc. 1993 Int. Computer Music Conf.*, pp. 32–39.

[114] B. Vercoe, "The synthetic performer in the context of live performance," in *Proc. 1984 Int. Computer Music Conf.*, pp. 199–200.

[115] ——, "Perceptually-based music pattern recognition and response," in *Proc. 1994 Int. Conf. Music Perception and Cognition.*, pp. 59–60

[116] B. Vercoe, "Csound: A manual for the audio processing system," Massachusetts Institute of Technology Media Lab, Cambridge, 1996.

[117] B. Vercoe and M. Puckette, "Synthetic rehearsal: Training the synthetic performer," in *Proc. 1985 Int. Computer Music Conf.*, pp. 275–278.

[118] Virtual reality modeling language. (1997). [Online]. Available: http://www.vrml.org.

[119] D. Wang, "Stream segregation based on oscillatory correlation," in *Readings in Computational Auditory Scene Analysis*, H. Okuno and D. Rosenthal, Eds. Mahweh, NJ: Erlbaum, 1998.

[120] W. Warren and R. Verbrugge, "Auditory perception of breaking and bouncing objects," in *Natural Computation*, W. Richards, Ed. Cambridge, MA: MIT Press, 1988.

[121] D. Weintraub, "A theory and computational model of auditory monaural sound separation," Ph.D. dissertation, CCRMA, Stanford Univ., Stanford, CA, 1985

[122] D. L. Wessel, "Timbre space as a musical control structure," *Comput. Music J.*, pp. 45–52, Summer 1979.

[123] L. Wyse and S. Smoliar, "Toward content-based audio indexing and retrieval," in *Readings in Computational Auditory Scene Analysis*, H. Okuno and D. Rosenthal, Eds. Mahweh, NJ: Erlbaum, 1998.

Barry L. Vercoe was born in New Zealand. He received degrees in music and mathematics from the University of Auckland, New Zealand, and the Ph.D. degree in composition from the University of Michigan, Ann Arbor.

He conducted postdoctoral research in digital synthesis at Princeton University, Princeton, NJ. He is the developer of the Music360 (1969), Music11 (1973), and Csound (1985) software-synthesis languages, which have become standards in the industry. He has taught at Oberlin College, Oberlin, OH, and Yale University, New Haven, CT. Since 1971, he has been a Member of the Faculty of the Massachusetts Institute of Technology (MIT), Cambridge, where he was a Founding Member of the MIT Media Laboratory in 1985. At the Media Lab, he has been Head of groups on music and cognition, synthetic listeners and performers, and machine listening. Under a Guggenheim Award in 1982–1984, he did pioneering work at IRCAM, Paris, France, on live instrument tracking, score following, and automatic accompaniment, which remains a special interest.

William G. Gardner was born in Meriden, CT, in 1960. He received the B.S. degree in computer science and the Ph.D. degree from the Massachusetts Institute of Technology (MIT), Cambridge.

For seven years, he was a Software Engineer for Kurzweil Music Systems, where he helped develop software and signal-processing algorithms for Kurzweil synthesizers. He recently founded Wave Arts, Inc., a company that develops innovative audio-processing software. His research interests are spatial audio, reverberation, sound synthesis, real-time signal processing, and psychoacoustics.

Dr. Gardner is a member of the Audio Engineering Society and the Acoustical Society of America. He received a Motorola Fellowship at the MIT Media Lab and the 1997 Audio Engineering Society Publications Award.

Eric D. Scheirer (Student Member, IEEE) was born in Binghampton, NY. He received the bachelor's degree in computer science and linguistics from Cornell University, Ithaca, NY, and the M.S. degree from the Massachusetts Institute of Technology (MIT) Media Laboratory, Cambridge. He currently is pursuing the Ph.D. degree at the MIT Media Lab.

His research focuses on the construction of music-understanding computer systems and methods of structured audio coding and transmission. He has been an Intern with the Interval Research Corporation. He is an Editor of the Motion Pictures Experts Group (MPEG)-4 audio standard, the Principal Technical Contributor to the sound-synthesis components of MPEG-4, and the Inventor of the MPEG-4 standard synthesis language SAOL. He is an accomplished jazz trombonist.

Mr. Scheirer is a member of the Audio Engineering Society. He received an Interval Research Fellowship to the MIT Media Lab and was twice an Advanced Music Performance Scholar at MIT.