Helsinki University of Technology
Department of Electrical and Communications Engineering
Laboratory of Acoustics and Audio Signal Processing

Evaluation of Modern Sound Synthesis Methods

Tero Tolonen, Vesa Välimäki, and Matti Karjalainen

Report 48
March 1998

ISBN 951-22-4012-2
ISSN 1239-1867

Espoo 1998

Table of Contents

List of Figures

List of Tables

Abstract

In this report, several digital sound synthesis methods are described and evaluated. The methods are divided into four groups according to a taxonomy proposed by Smith. Representative examples of sound synthesis techniques in each group are chosen. The evaluation criteria are based on those proposed by Jaffe. The selected synthesis methods are rated with a discussion concerning each criterion.

Keywords: sound synthesis, digital signal processing, musical acoustics, computer music

Preface

The main part of this work has been carried out as part of phase I of the TEMA (Testbed for Music and Acoustics) project that has been funded within European Union's Open Long Term Research ESPRIT program. The duration of phase I was 9 months during the year 1997. This report discusses digital sound synthesis methods. As Deliverable 1.1a of the TEMA project phase I, it aims at giving guidelines for the second phase of the project for the development of a sound synthesis and processing environment.

The partners of the TEMA consortium in the phase I of the project were Helsinki University of Technology, Staatliches Institut für Musikforschung (Berlin, Germany), the University of York (United Kingdom), and SRF/PACT (United Kingdom). In Helsinki University of Technology (HUT), two laboratories were involved in the TEMA project: the Laboratory of Acoustics and Audio Signal Processing and the Telecommunications and Multimedia Laboratory.

The authors would like to thank Professor Tapio "Tassu" Takala for his support and guidance as the TEMA project leader at HUT. We are also grateful to Dr. Dr. Ioannis Zannos, who acted as the coordinator of the TEMA project, and representatives of other partners for smooth collaboration and fruitful discussions.

This report summarizes and extends the contribution in the TEMA project by the HUT Laboratory of Acoustics and Audio Signal Processing. We would like to acknowledge the insightful comments on our manuscript given by Professor Julius O. Smith (CCRMA, Stanford University, California, USA), Dr. Davide Rocchesso and Professor Giovanni de Poli (both at CSC-DEI, University of Padova, Padova, Italy).

Espoo, March 26, 1998

Tero Tolonen, Vesa Välimäki, and Matti Karjalainen

1. Introduction

Digital sound synthesis methods are numerical algorithms that aim at producing musically interesting and preferably realistic sounds in real time. In musical applications, the input for sound synthesis consists of control events only. Numerous different approaches are available.

The purpose of this document is not to try to reach the details of every method. Rather, we attempt to give an overview of several sound synthesis methods. The second aim is to establish the tasks or synthesis problems that are best suited for a given method. This is done by evaluation of the synthesis algorithms. We would like to emphasize that no attempt has been made to put the algorithms in any precise order as this, in our opinion, would be impossible.

The synthesis algorithms were chosen to be representative examples in each class. With each algorithm, an attempt was made to give an overview of the method and to refer the interested reader to the literature.

The approach followed for evaluation is based on a taxonomy by Smith (1991). Smith divides digital sound synthesis methods into four groups: abstract algorithms, processed recordings, spectral models, and physical models. This document follows Smith's taxonomy in a slightly modified form and discusses representative methods from each category.

Each method was categorized into one of the following groups: abstract algorithms, sampling and processed recordings, spectral models, and physical models. More emphasis is given to spectral and physical modeling since these seem to provide more attractive future prospects in high-quality sound synthesis. In these last categories there is more activity in research and, in general, their future potential looks especially promising.

This document is organized as follows. Selected synthesis methods are presented in Chapters 2 – 5. After that, evaluation criteria are developed based on those proposed by Jaffe (1995). An additional criterion is included concerning the suitability of a method for distributed and parallel processing. The evaluation results are collected in a table in which the rating of each method can be compared. The document is concluded with a discussion of the features desirable in an environment in which the methods discussed can be implemented.

2. Abstract Algorithms, Processed Recordings, and Sampling

The first experiments that can be interpreted as ancestors of computer music were done at 1920's by composers like Milhaud, Hindemith, and Toch, who experimented with variable speed phonographs in concert (Roads, 1995). In 1950 Pierre Schaeffer founded the Studio de Musique Concrète in Paris (Roads, 1995). In *musique concrète* the composer works with sound elements obtained from recordings or real sounds.

The methods presented in this chapter are based either on abstract algorithms or on recordings of real sounds. According to Smith (1991), these methods may become less common in commercial synthesizers as more powerful and expressive techniques arise. However, they still serve as a useful background for the more elaborate sound synthesis methods. Particularly, they may still prove to be superior in some specific sound synthesis problems, e.g., when simplicity is of highest importance, and we are likely to see them in use for decades.

The chapter starts with three methods based on abstract algorithms: FM synthesis, waveshaping synthesis, and the Karplus-Strong algorithm. Then, three methods utilizing recordings are discussed. These are sampling, multiple wavetable synthesis, and granular synthesis.

2.1 FM Synthesis

FM *(frequency modulation)* synthesis is a fundamental digital sound synthesis technique employing a nonlinear oscillating function. FM synthesis in a wide sense consists of a family of methods each of which utilizes the principle originally introduced by Chowning (1973).

The theory of FM was well established by the mid-twentieth century for radio frequencies. The use of FM in audio frequencies for the purposes of sound synthesis was not studied until late 60's. John Chowning at Stanford University was the first to study systematically FM synthesis. The time-variant structure of natural sounds is relatively hard to achieve using linear techniques, such as additive synthesis (see section 3.1). Chowning observed that complex audio spectra can be achieved with just two sinusoidal oscillators. Furthermore, the synthesized complex spectra can

be varied in time.

2.1.1 FM Synthesis Method

In the most basic form of FM, two sinusoidal oscillators, namely, the carrier and the modulator, are connected in such a way that the frequency of the carrier is modulated with the modulating waveform. A simple FM instrument is pictured in Figure 2.1 (a). The output signal $y(n)$ of the instrument can be expressed as

$$y(n) = A(n)\sin[2\pi f_{\mathrm{c}} n + I\sin(2\pi f_{\mathrm{m}} n)], \tag{2.1}$$

where $A(n)$ is the amplitude, f_{c} is the carrier frequency, I is the modulation index, and f_{m} is the modulating frequency. The modulation index I represents the ratio of the peak deviation of modulation to the modulating frequency. It is clearly seen that when $I = 0$, the output is the sinusoidal $y(n) = A(n)\sin(2\pi f_{\mathrm{c}} n)$ corresponding to zero modulation. Note that there is a slight discrepancy between Figure 2.1 (a) and Equation 2.1, since in the equation the phase and not the frequency is being modulated. However, since these presentations are frequently encountered in literature, e.g., in (De Poli, 1983; Roads, 1995), they are also adopted here. Holm (1992) and Bate (1990) discuss the effect of phase and differences between implementations of the simple FM algorithm.

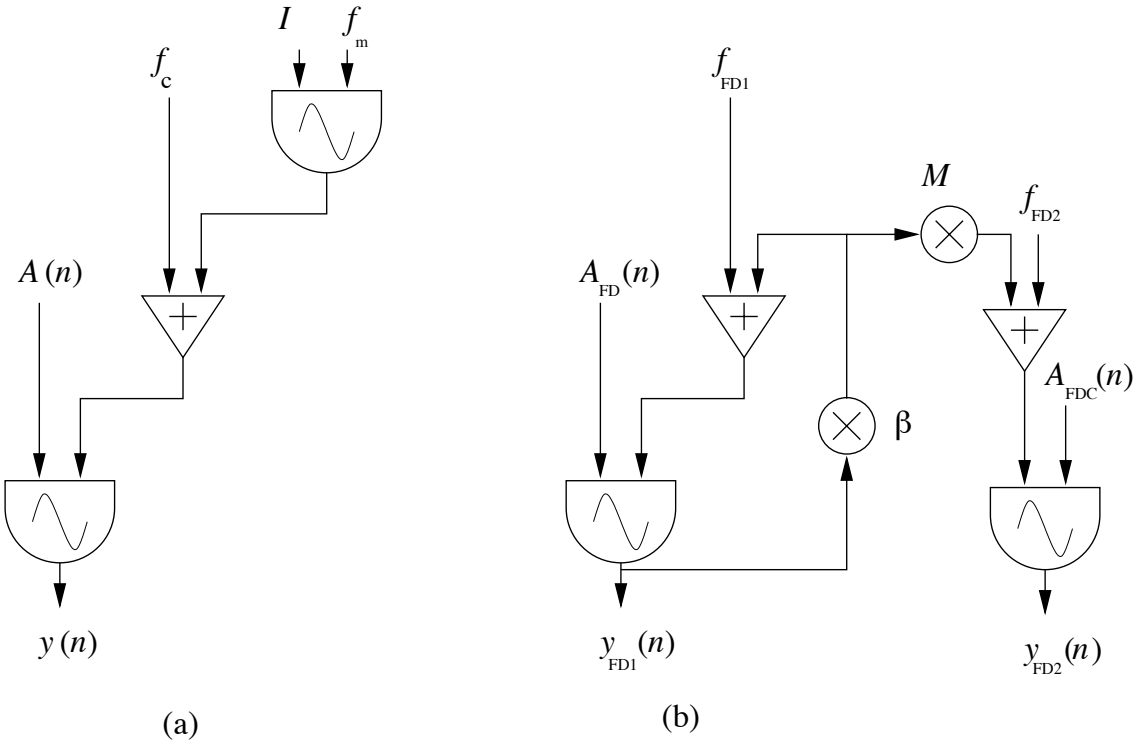

Figure 2.1: (a): A simple FM synthesis instrument. (b) one-oscillator feedback system with output $y_{\mathrm{FD1}}(n)$ and two-oscillator feedback system with output $y_{\mathrm{FD2}}(n)$, after (Roads, 1995).

The expression of the output signal in Equation 2.1 can be developed further

(Chowning, 1973; De Poli, 1983)

$$y(n) = \sum_{k=-\infty}^{\infty} J_k(I) \sin |2\pi(f_c + k f_m n)|, \tag{2.2}$$

where J_k is the *Bessel function* of order k. Inspection of Equation 2.2 reveals that the frequency-domain representation of the signal $y(n)$ consists of a peak at f_c and additional peaks at frequencies

$$f_n = f_c \pm n f_m, \quad n = 1, 2, \cdots,$$

as pictured in Figure 2.2. Part of the energy of the carrier waveform is distributed to the side frequencies f_n. Note that Equation 2.2 allows the partials be determined analytically.

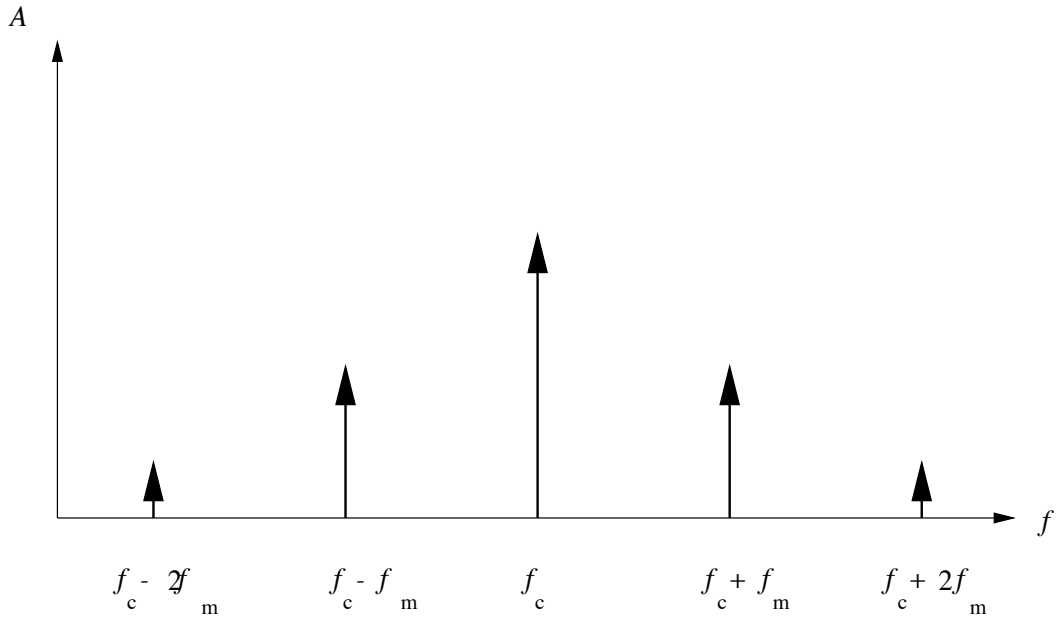

Figure 2.2: Frequency-domain presentation of FM synthesis.

A harmonic spectrum is created when the ratio of carrier and modulator frequency is a member of the class of ratios of integers, i.e,

$$\frac{f_c}{f_m} = \frac{N_1}{N_2}, \quad N_1, N_2 \in Z.$$

Otherwise, the spectrum of the output signal is inharmonic. Truax (1977) discusses the mapping of frequency ratios into spectral families.

2.1.2 Feedback FM

In simple FM, the amplitude ratios of harmonics vary unevenly when the modulation index I is varied. Feedback FM can be used to solve this problem (Tomisawa, 1981). Two feedback FM systems are pictured in Figure 2.1 (b). The one-oscillator feedback FM system is obtained from the simple FM by replacing the frequency modulation

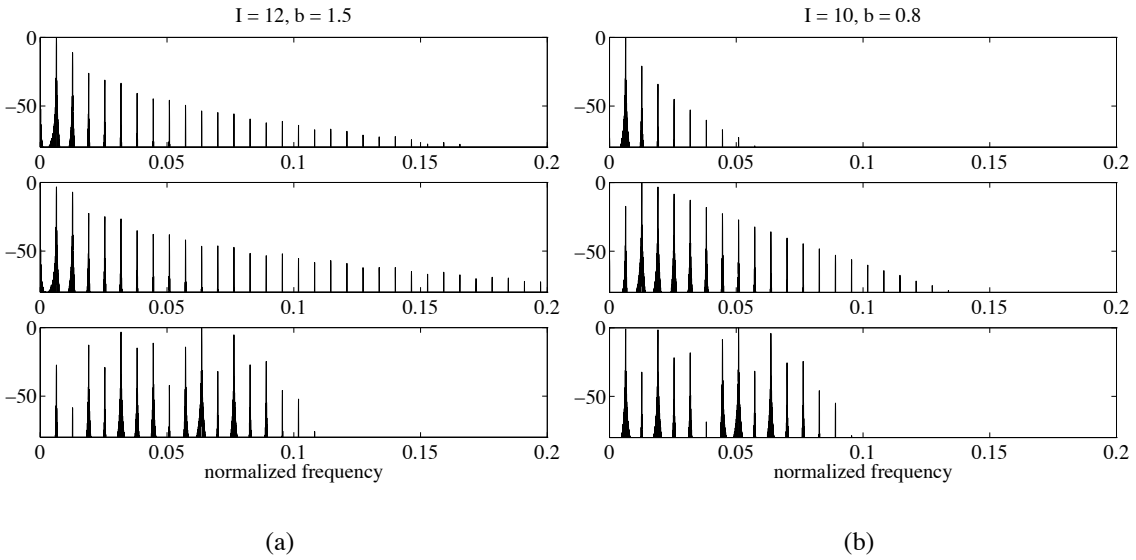

Figure 2.3: A comparison of three different FM techniques. Spectra of one-oscillator feedback FM are presented on top, those of two-oscillator feedback FM in the middle, and spectra of simple FM on the bottom. The frequency values, f_{FD1} and f_{FD2}, of the oscillators in the feedback system are equal. The modulation index M is set to 2. Parameter b is the feedback coefficient.

oscillator by a feedback connection from the output of the system. The two-oscillator system uses a feedback connection to drive the frequency modulation oscillator.

Figure 2.3 shows the effect of the feedback connections. The spectra of signals produced by the two feedback systems as well as the spectra of the signal produced by the simple FM are computed for two sets of parameters in figures 2.3 (a) and (b). The more regular behavior of the harmonics in the feedback systems is clearly visible. Furthermore, it can be observed that the two-oscillator system produces more harmonics for the same parameters.

2.1.3 Other Developments of the Simple FM

Roads (1995) gives on overview of different methods based on simple FM. The first commercial FM synthesizer, the GS1 digital synthesizer, was introduced by Yamaha after developing the FM synthesis method patented by Chowning further. The first synthesizer was very expensive, and it was only after introduction of the famous DX7 synthesizer that FM became the dominating sound synthesis method for years. It is still used in many synthesizers, and in "SoundBlaster-compatible" computer sound cards, chips, and software. Yamaha has patented the feedback FM method (Tomisawa, 1981).

2.2 Waveshaping Synthesis

Waveshaping synthesis, also called *nonlinear distortion*, is a simple sound synthesis method using a nonlinear *shaping function* to modify the input signal. First exper-

iments on waveshaping were made by Risset in 1969 (Roads, 1995). Arfib (1979) and Le Brun (1979) developed independently the mathematical formulation of the waveshaping. Both also performed some empirical experiments with the method.

In the most fundamental form, waveshaping is implemented as a mapping of a sinusoidal input signal with a nonlinear distortion function w. Examples of these mappings are illustrated in Figure 2.4. The function w maps the input value $x(n)$ in the range $[-1, 1]$ to an output value $y(n)$ in the same range. Waveshaping can be very easily implemented by a simple table lookup, i.e., the function w is stored in a table which is then indexed with $x(n)$ to produce the output signal $y(n)$.

Both Arfib (1979) and Le Brun (1979) observed that the ratios of the harmonics could be accurately controlled by using Chebyshev polynomials as distortion functions. The Chebyshev polynomials have the interesting feature that when a polynomial of order n is used as a distortion function to a sinusoidal signal with frequency ω, the output signal will be a pure sinusoid with frequency $n\omega$. Thus, by using a linear combination of Chebyshev polynomials as the distortion function, the ratio of the amplitudes of the harmonics can be controlled. Furthermore, the signal can be maintained bandlimited, and the aliasing of harmonics can be avoided. See (Le Brun, 1979) for discussion on the normalization of the amplitudes of the harmonics.

The signal obtained by the waveshaping method can be postprocessed, e.g., by amplitude modulation. This way the spectrum of the waveshaped signal has components distributed around the modulating frequency f_m spaced at intervals f_0, the frequency of the undistorted sinusoidal signal. If the spectrum is aliased, an inharmonic signal may be produced. See (Arfib, 1979) for more details and (Roads, 1995) for references on other developments of the waveshaping synthesis.

2.3 Karplus-Strong Algorithm

Karplus and Strong (1983) developed a very simple method for surprisingly high-quality synthesis of plucked string and drum sounds. The Karplus-Strong (KS) algorithm is an extension to the simple wavetable synthesis technique where the sound signal is periodically read from a computer memory. The modification is to change the wavetable each time a sample is being read. A block diagram of the simple wavetable synthesis and a generic design of the Karplus-Strong algorithm are shown in Figure 2.5 (a) and (b), respectively. In the KS algorithm the wavetable is initialized with a sequence of random numbers, as opposed to wavetable synthesis where usually a period of a recorded instrument tone is used.

The simplest modification that produces useful results is to average two consecutive samples of the wavetable as shown in Figure 2.5 (c). This can be written as

$$y(n) = \frac{1}{2}[y(n-P) + y(n-P-1)], \tag{2.3}$$

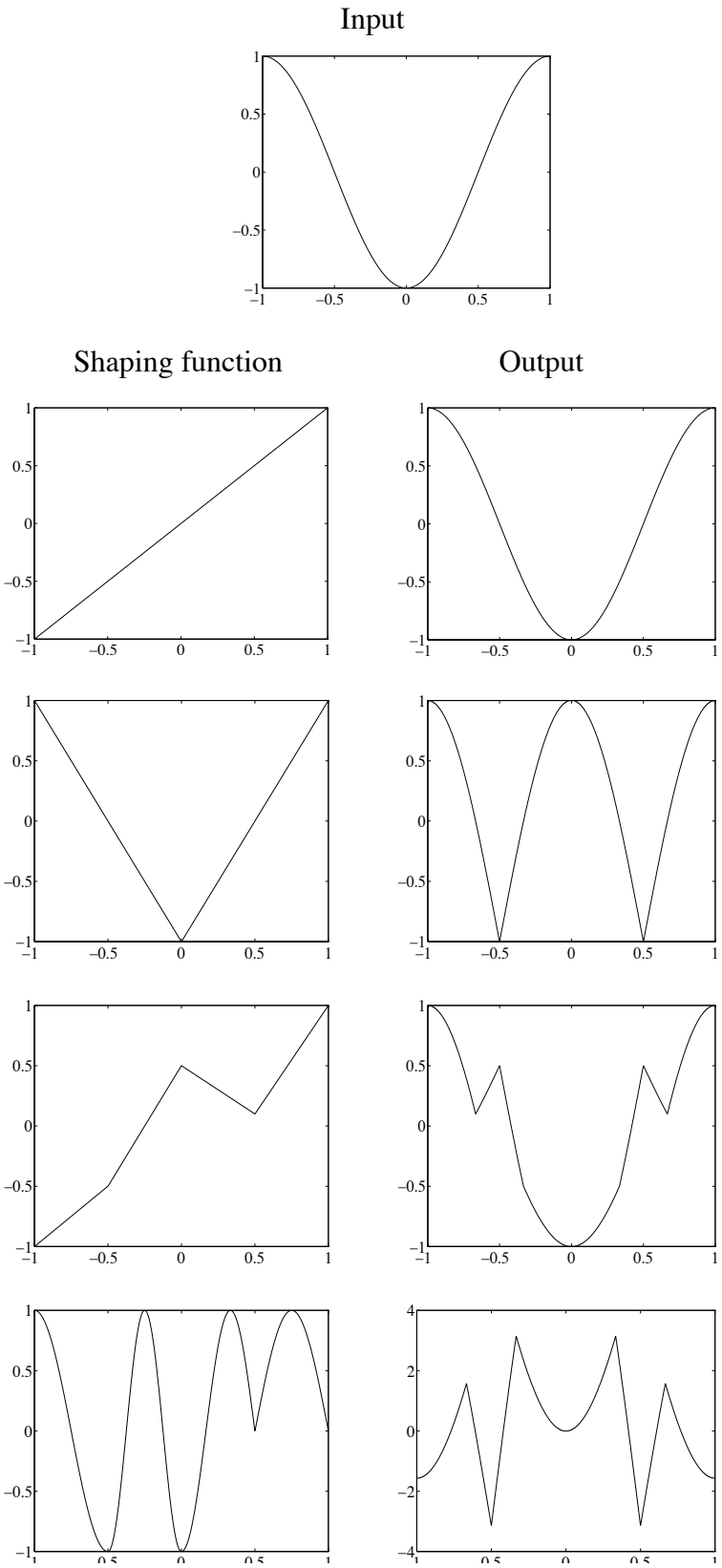

Figure 2.4: Waveshaping with four different shaping functions. The input function is presented on the top.

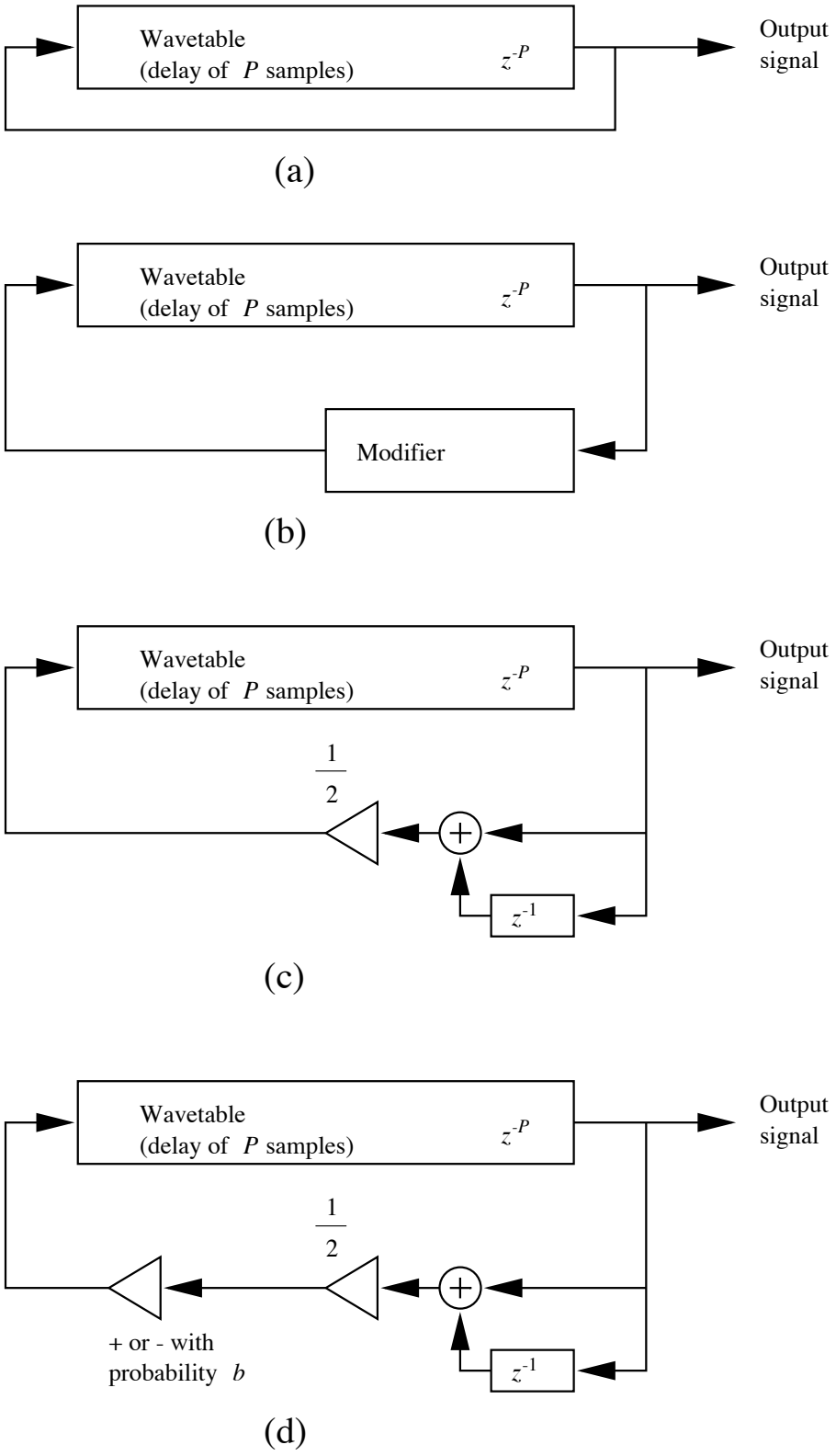

Figure 2.5: The Karplus-Strong algorithm. The simple wavetable synthesis is shown in (a), a generic design with an arbitrary modification function in (b), a Karplus-Strong model for plucked string tones in (c), and a Karplus-Strong model for percussion instrument tones in (d), after (Karplus and Strong, 1983).

where P is the delay line length. The transfer function of the simple modifier filter is

$$H(z) = \frac{1}{2}(1 + z^{-1}). \tag{2.4}$$

This is a lowpass filter and it accounts for the decay of the tone. A multiply-free structure can be implemented with only a sum and a shift for every output sample. This structure can be used to simulate plucked string instrument tones.

The model for percussion timbres is shown in Figure 2.5 (d). Now the output sample $y(n)$ depends on the wavetable entries by

$$y(n) = \begin{cases} \frac{1}{2}[(n-P) + y(n-P-1)], & \text{if } r < b, \\ -\frac{1}{2}[(n-P) + y(n-P-1)], & \text{if } r > b, \end{cases} \tag{2.5}$$

where r is a uniformly distributed random variable between 0 and 1 and b is a parameter called the blend factor. When $b = 1$, the algorithm reduces to that of Equation 2.3. When $b = \frac{1}{2}$, drum-like timbres are obtained. With $b = 0$, the entire signal is negated every $p + \frac{1}{2}$ samples and a octave lower tone with odd harmonics only is produced.

The KS algorithm is basically a comb filter. This can be seen by examining the frequency response of the algorithm. In order to compute the frequency response, we assume that we can feed a single impulse into the delay line that has been initialized with zero values. We then compute the output signal and obtain a frequency domain representation from the response, i.e., we interpret the output signal as the impulse response of the system. The corresponding frequency response is depicted in Figure 2.6. Notice the harmonic structure and that the magnitude of the peaks decreases with frequency, as expected.

Karplus and Strong (1983) propose modifications to the algorithm including excitation with a nonrandom signal. A physical modeling interpretation of the Karplus-Strong algorithm is taken by Smith (1983) and Jaffe and Smith (1983). Extensions to the Karplus-Strong algorithm are presented in Section 5.3.

2.4 Sampling Synthesis

Sampling synthesis is a method in which recordings of relatively short sounds are played back (Roads, 1995). Digital sampling instruments, also called samplers, are typically used to perform pitch shifting, looping, or other modification of the original sound signal (Borin et al., 1997b).

Manipulation of recorded sounds for compositional purposes dates back to the 1920's (Roads, 1995). Later, magnetic tape recording permitted cutting and splicing of recorded sound sequences. Thus, editing and rearrangement of sound segments was available. In 1950 Pierre Schaeffer founded the Studio de Musique Concrète at Paris and began to use tape recorders to record and manipulate sounds (Roads, 1995). Analog samplers were based on either optical discs or magnetic tape devices.

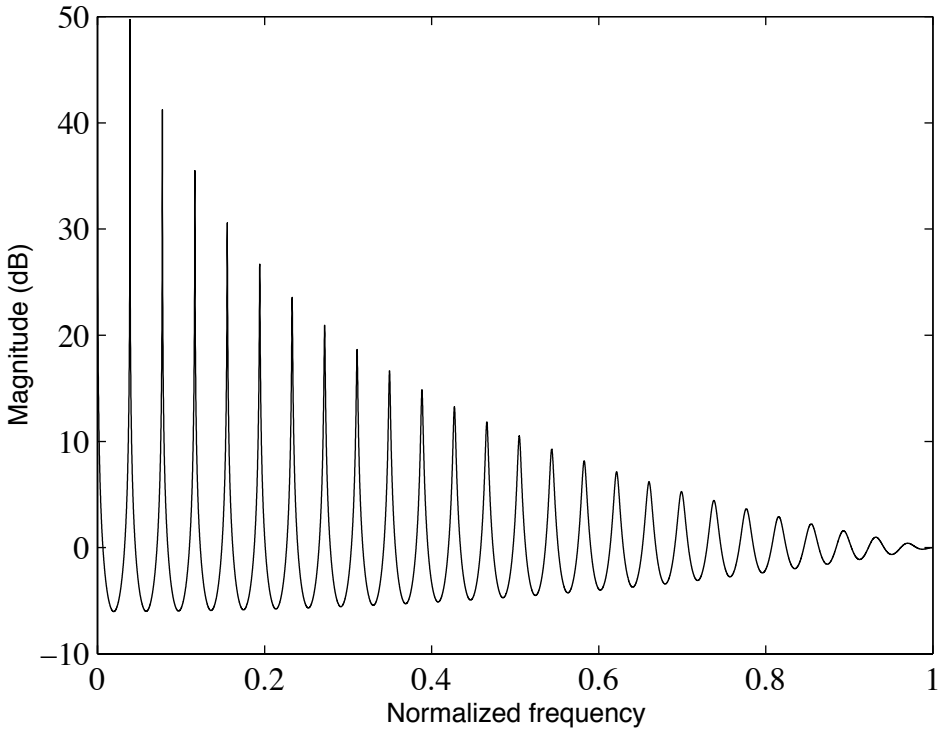

Figure 2.6: Frequency response of the Karplus-Strong model.

Sampling synthesis typically uses signals of several seconds. The synthesis itself is very efficient to implement. In its simplest form, it consists only of one table lookup and pointer update for every output sample. However, the required amount of memory storage is huge. Three most widely used methods to reduce the memory requirements are presented in the following. They are looping, pitch shifting, and data reduction (Roads, 1995). The interested reader should also consult a work by Bristow-Johnson (1996) on wavetable synthesis.

2.4.1 Looping

One obvious way of reducing the memory usage in sampling synthesis is to apply looping to the steady state part of a tone (Roads, 1995). With a number of instrument families the tone stays relatively constant in amplitude and pitch after the attack, until the tone is released by the player. The steady-state part can thus be reproduced by looping over a short segment between so called loop points. After the tone is released the looping ends and the sampler will play the decay part of the tone.

The samples provided with commercial samplers are typically pre-looped, i.e., the loop points are already determined for the user. For new samples the determination of the looping points has to be done by the user. One method is to estimate the pitch of the tone and then select a segment of length of a multiple of the wavelength of the fundamental frequency. This kind of looping technique tends to create tones with smooth looping part and constant pitch (Roads, 1995). If the looping part is too short, an artificial sounding tone can be produced because the time-varying

qualities of the tone are discarded. The looping points can also be *spliced* or *cross-faded* together. A splice is simply a cut from one sound to the next and it is bound to produce a perceivable click unless done very carefully. In cross-fading the end of a looping part is faded out as the beginning of the next part is faded in. Even more sophisticated techniques for the determination of good looping points are available, see (Roads, 1995) for more information.

2.4.2 Pitch Shifting

In less expensive samplers there might not be enough memory capacity to store every tone of an instrument, or not all the notes have been recorded. Typically only every third or fourth semitone is stored and the intermediate tones are produced by applying pitch shifting to the closest sampled tone (Roads, 1995). This reduces the memory requirements by a factor of three or four, thus the data reduction is significant.

Pitch shifting in inexpensive samplers is typically carried out using simple time-domain methods that affect the length of the signal. The two methods usually employed are: varying the clock frequency of the output digital-to-analog converter and sample-rate conversion in the digital domain (Roads, 1995). More elaborate method for pitch shifting exist, see (Roads, 1995) for references. One way of achieving sampling rate conversion is to use interpolated table reads with adjustable increments. This can be done using *fractional delay filters* described in (Välimäki, 1995; Laakso et al., 1996).

2.4.3 Data Reduction

In many samplers the memory requirements are tackled by data reduction techniques. These can be divided into plain data reduction where part of the information is merely discarded and data compression where the information is packed into a more economical form without any loss in the signal quality.

Data reduction usually degrades the perceived audio quality. It consists of simple but crude techniques that either lower the dynamic range of the signal by using less bits to represent each sample or reduce the sampling frequency of the signal. These methods decrease the signal-to-noise ratio or the audio bandwidth, respectively. More elaborate methods exit and these usually take into account the properties of the human auditory system (Roads, 1995).

Data compression eliminates the redundancies present in the original signal to represent the information more memory-efficiently. Compression should not degrade the quality of the reproduced signal.

2.5 Multiple Wavetable Synthesis Methods

Multiple wavetable synthesis is a set of methods that have in common the use of

multiple wavetables, i.e. sound signals stored in a computer memory. The most widely used methods are *wavetable cross-fading* and *wavetable stacking* (Roads, 1995). Horner et al. (1993) introduce methods obtaining optimal wavetables to match signals of existing real instruments.

In wavetable cross-fading the tone is produced from several sections each consisting of a wavetable that is multiplied with an amplitude envelope. These portions are summed together so that a sound event begins with one wavetable that is then cross-faded to the next. This procedure is repeated over the course of the event (Roads, 1995). A common way to use wavetable cross-fading is to start a tone with a rich attack, such as a stroke or a pluck on a string, and then cross-fade this into a sustain part of a synthetic waveform (Roads, 1995).

Wavetable stacking is a variation of the additive synthesis discussed in Section 3.1. Several arbitrary sound signals are first multiplied with an amplitude envelope and then summed together to produce the synthetic sound signal. Using wavetable stacking, hybrid timbres can be produced combining elements of several recorded or synthesized sound signals. In commercial synthesizers usually from four to eight wavetables are used in wavetable stacking.

In (Horner et al., 1993) methods for matching the time-varying spectra of a harmonic wavetable-stacked tone to an original are presented. The objective is to find wavetable spectra and the corresponding amplitude envelopes that produce a close fit to the original signal. First, the original signal is analyzed using an extension of the McAulay-Quatieri (McAulay and Quatieri, 1986) (see Section 3.4) analysis method. A *genetic algorithm* (GA) and *principal component analysis* (PCA) are applied to obtain the basis spectra. The amplitude envelopes are created by finding a solution that minimizes a least squares error. The method produced good results with four wavetables when the GA was applied (Horner et al., 1993).

2.6 Granular Synthesis

Granular synthesis is a set of techniques that share a common paradigm of representing sound signals by "sound atoms" or grains. Granular synthesis originated from the studies by Gabor in the late 40's (Cavaliere and Piccialli, 1997; Roads, 1995). The synthetic sound signal is composed by adding these elementary units in the time domain.

In granular synthesis one sound grain can have duration ranging from one millisecond to more than a hundred milliseconds and the waveform of the grain can be a windowed sinusoid, a sampled signal, or obtained from a physics-based model of a sound production mechanism (Cavaliere and Piccialli, 1997). The granular techniques can be classified according to how the grains are obtained. In the following a classification derived from one given by Cavaliere and Piccialli (1997) is presented and the techniques of each category are shortly described.

2.6.1 Asynhcronous Granular Synthesis

Asynchronous granular synthesis (AGS) has been developed by Roads (1991, 1995). It is a method that scatters sound grains in a statistical manner over a region in the time-frequency plane. The regions are called sound *clouds* and they form the elementary unit the composer works with (Roads, 1995). A cloud is specified by the following parameters: start time and duration of a cloud, grain duration, density of grains, amplitude envelope and bandwidth of the cloud, waveform of each grain, and spatial distribution of the cloud.

The grains of a cloud can all have similar waveforms or a random mixture of different waveforms. A cloud can also mutate from grains with one waveform to grains with another over the duration of the cloud. The duration of a grain effects also its bandwidth. The shorter a grain is, the more it is spread in the frequency domain. Pitched sounds can be created with low bandwidth clouds. Roads (1991, 1995) gives a more detailed discussion on the parameters of the AGS.

AGS is effective in creating new sound events that are not easily produced by musical instruments. On the other hand, simulations of existing sounds are very hard to achieve with AGS. In the following discussion granular synthesis methods better suited for that are presented.

2.6.2 Pitch Synchronous Granular Synthesis

Pitch synchronous granular synthesis (PSGS) is a method developed by De Poli and Piccialli (1991). The method is also discussed in (Cavaliere and Piccialli, 1997) and briefly in (Roads, 1995). In PSGS grains are derived from the *short-time Fourier transform* (STFT). The signal is assumed to be (nearly) periodic and, first, the fundamental frequency of the signal is detected. The period of the signal is used as the length of the rectangular window used in the STFT analysis. When used synchronously to the fundamental frequency, the rectangular window has the attractive property of minimizing the side effects that occur with windowing, i.e., the spectral energy spread.

After windowing, a set of analysis grains are obtained in such a way that each grain corresponds to one period of the signal. From these analysis grains, impulse responses corresponding to prominent content in the frequency domain representation are derived. Methods for the system impulse response estimation include *linear predictive coding* (LPC), and interpolation of the frequency domain representation of a single period of the signal (Cavaliere and Piccialli, 1997).

In the resynthesis stage, a train of impulses is used to drive a set of parallel FIR filters obtained from the system impulse response. The period of the pulse train is obtained from the detected fundamental frequency. See (De Poli and Piccialli, 1991) for transformations that can create variations to the produced signal.

2.6.3 Other Granular Synthesis Methods

Some sound synthesis methods presented elsewhere in this document can also be interpreted as special cases of granular synthesis. These include the wave-function synthesis (Section 3.8.1) and VOSIM (Section 3.8.2). In fact, all methods that use the overlap-add technique for synthesizing sound signals can be thought of as being instances of granular synthesis.

Another possibility is to apply the *wavelet transform* to obtain a multiresolution representation of the signal. See (Evangelista, 1997) for a discussion on wavelet representations of musical signals.

3. Spectral Models

Spectral sound synthesis methods are based on modeling the properties of sound waves as they are perceived by the listener. Many of them also take the knowledge of psychoacoustics into account. Spectral sound synthesis methods are general in that they can be applied to model a wide variety of sounds.

In this chapter, three traditional linear synthesis methods, namely, *additive synthesis*, *the phase vocoder*, and *source-filter synthesis*, are first discussed. Second, *McAulay-Quatieri algorithm*, *Spectral Modeling Synthesis* (SMS), *Transient Modeling Synthesis* (TMS), and the inverse-FFT based additive synthesis method *(FFT^{-1} synthesis)* are described. Finally, two methods for modeling the human voice are shortly addressed. These methods are the *CHANT* and the *VOSIM*.

3.1 Additive Synthesis

Additive synthesis is a method in which a composite waveform is formed by summing sinusoidal components, for example, harmonics of a tone, to produce a sound (Moorer, 1985). It can be interpreted as a method to model the time-varying spectra of a tone by a set of discrete lines in the frequency domain (Smith, 1991).

The concept of additive synthesis is very old and it has been used extensively in electronic music; see (Roads, 1995, pp. 134–136) for references and historical treatment. In 1964 Risset (1985) applied the method for the first time to reproduce sounds based on the analysis of recorded tones. With this application to trumpet tones, the control data was reduced by applying piecewise-linear approximation of the amplitude envelopes. Many of the modern spectral modeling methods use additive synthesis in some form. These methods are discussed later in this chapter. A block diagram of additive synthesis with slowly-varying control functions is depicted in Figure 3.1.

In additive synthesis, three control functions are needed for every sinusoidal oscillator: the amplitude, frequency, and phase of each component. In many cases the phase is left out and only the amplitude and frequency functions are used. The

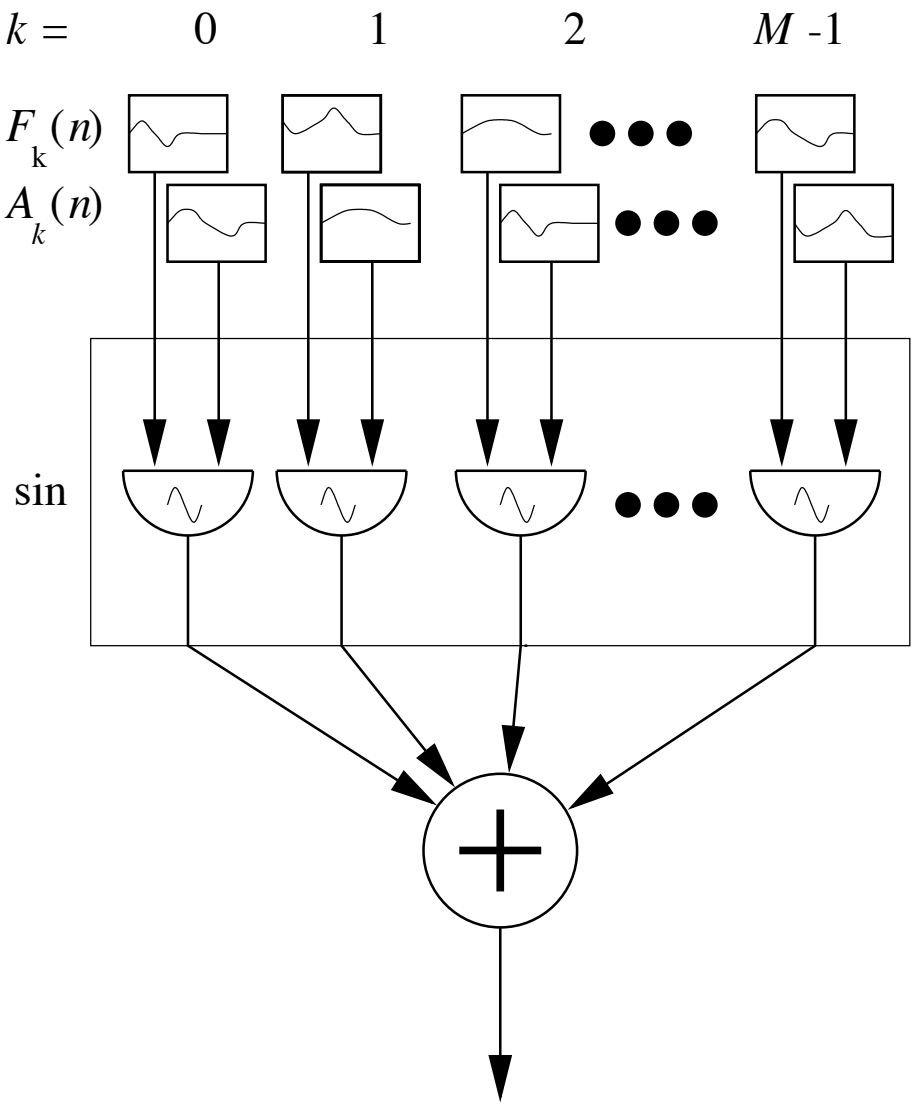

Figure 3.1: Time-varying additive synthesis, after (Roads, 1995).

output signal $y(n)$ is the sum of the components and can be represented as

$$y(n) = \sum_{k=0}^{M-1} A_k(n) \sin[2\pi F_k(n)], \tag{3.1}$$

where T is the sampling interval, n is the time index, M is the number of the sinusoidal oscillator, ω_k is the radian frequency of the oscillator, $A_k(n)$ is the time varying amplitude of the k^{th} partial and $F_k(n)$ is the frequency deviation of the k^{th} partial. If the tone is harmonic, ω_k is a multiple of the fundamental radian frequency ω_0, i.e., $\omega_k = k\omega_0$. $A_k(n)$ and $F_k(n)$ are assumed to be slowly time-varying.

The control functions can be obtained with several procedures (Roads, 1995). One is to use arbitrary shapes, for instance, some composers have tracked the shapes of mountains or urban sky lines. The functions can also be generated using composition programs. An analysis method can be applied to map a natural sound into a series of control functions. Such a system is pictured in Figure 3.2. The *Short Time Fourier Transform* (STFT) is calculated from the input signal. Harmonics

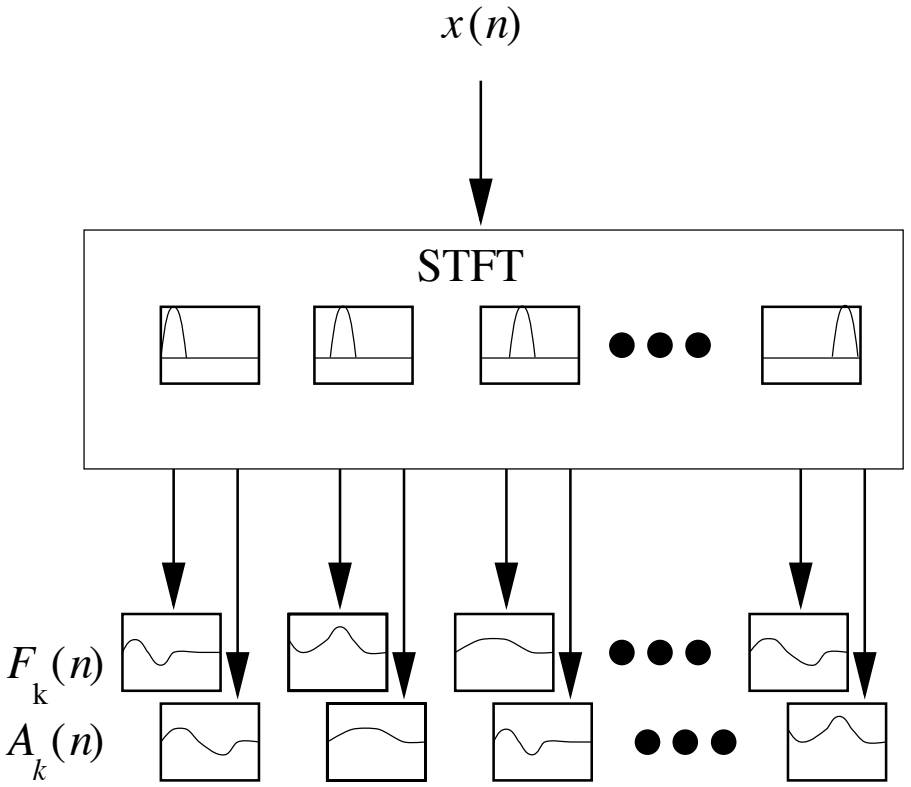

Figure 3.2: The additive analysis technique, after (Roads, 1995). A STFT is calculated from the input signal. Frequency and amplitude trajectories in the time domain are formed.

are mapped to peaks in the frequency domain, and their frequency and amplitude functions can be detected from the series of STFT frames. These control functions can be used directly to synthesize tones in a system of Figure 3.1.

The main drawbacks of the additive synthesis are the enormous amount of data involved and the demand for a large number of oscillators. The method gives best results when applied to harmonic or almost harmonic signals where little noise is present. Synthesis of noisy signals requires a vast number of oscillators. A method for the reduction of control data is discussed in the following subsection.

3.1.1 Reduction of Control Data in Additive Synthesis by Line-Segment Approximation

There are several ways to reduce the amount of control data needed (Roads, 1995, p. 149), (Moorer, 1985). The main criteria for the data reduction method are 1) to retain the intuitively appealing form of the control data, i.e., the composer has to be able to easily modify the control data to obtain musically interesting effects on sound, and 2) to preserve the original sound in the absence of transformation.

Line-segment approximation can be utilized to obtain a set of piecewise linear curves approximating the frequency and the amplitude control functions. The

method has been used by Risset (1985), and it is also discussed by Moorer (1985) and Strawn (1980). The idea of the line-segment approximation is to fit a set of straight lines to each control function in such a way that the curve obtained resembles the original curve. An example of the line-segment approximation is illustrated in Figure 3.3 where the amplitude trajectory of the 4^{th} partial of a flute tone has been approximated using line segments.

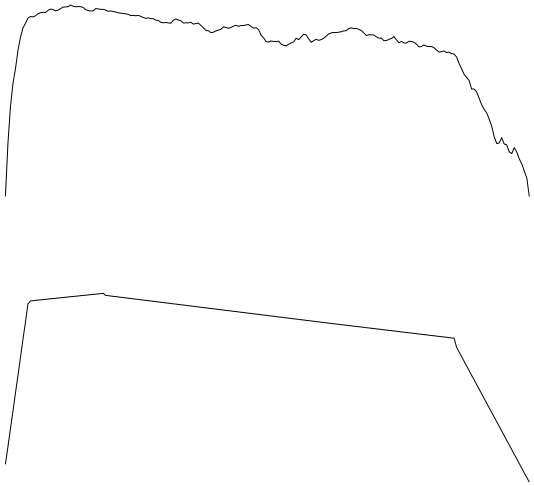

Figure 3.3: The line-segment approximation in additive synthesis.

When 32 partials of a tone are approximated by line-segment approximation using ten segments with 16 bit numbers for each partial, the result is approximately 5120 bits of data for a half-second tone. The same tone when using sampling rate of 44 100 Hz and 16 bit samples amounts to 352 800 bits. Thus the data reduction ratio is about 1 to 69.

3.2 The Phase Vocoder

The phase vocoder was developed at Bell laboratories and was first described by Flanagan and Golden (1966). All vocoders present the input signal in multiple parallel channels, each of which describes the signal in a particular frequency band. Vocoders simplify the complex spectral information and reduce the amount of data needed to present the signal.

In the original channel vocoder (Dudley, 1939) the signal is described as an excitation signal and values of short time amplitude spectra measured at discrete frequencies. The phase vocoder, however, uses *complex short time spectra* and thus preserves *the phase information* of the signal.

The analysis part of the method can be considered to be either a bank of bandpass filters or a short-time spectrum analyzer. These viewpoints are mathematically equivalent for, in theory, the original signal can be reproduced undistorted (Gordon and Strawn, 1985; Dolson, 1986). Portnoff (1976) gives a mathematical treatment on the subject, and he also shows that the phase vocoder can be formulated as an

identity system in the absence of parameter modifications. An introductory text of the phase vocoder can be found in (Serra, 1997a).

The implementation of the phase vocoder using the STFT is computationally more efficient than using a filter bank, since the complex spectrum can be evaluated with the *fast Fourier transform* (FFT) algorithm. Detailed discussions on the phase vocoder and practical implementations are given by Portnoff (1976), Moorer (1978), and Dolson (1986). Code for implementing the phase vocoder can be found in (Gordon and Strawn, 1985) and (Moore, 1990).

The phase vocoder is pictured in Figure 3.4. The input signal is divided into equally spaced frequency bands. This can be done by applying the STFT to the windowed signal. Each bin of the STFT frame corresponds to the magnitude and phase values of the signal in that frequency band at the time of the frame.

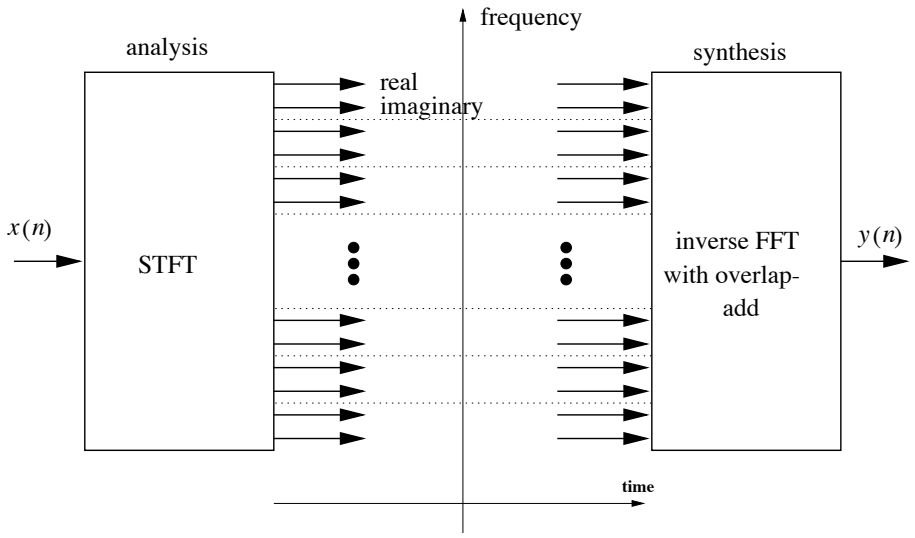

Figure 3.4: The phase vocoder, after (Roads, 1995). First the STFT is calculated from the input signal. The signal is now presented as a multiple series of complex number pairs corresponding to the signal components in each frequency band. The output signal is composed by calculating the inverse FFT for each frame and by using the overlap-add method to reconstruct the signal in the time domain.

Time scaling and pitch transposition are effects that can be easily performed using the phase vocoder (Dolson, 1986; Serra, 1997a). Time-varying filtering can also be utilized (Serra, 1997a). Time scaling is done by modifying the hop size in the synthesis stage. If the hop size is increased, each STFT frame will effectively sound longer and the produced signal is a stretched version of the original. If the hop size is reduced the opposite occurs. The modification of the hop size has to be taken into account in the analysis stage by choosing a window that minimizes the side effects. Otherwise, some output samples are given more weight and the synthetic signal is amplitude modulated. The phase values need also to be compensated for in the modification stage. The phase values have to be multiplied by a scaling factor in order to retain the correct frequency. Pitch shifting without changing the temporal evolution can be accomplished by first modifying the time scale by the desired pitch-scaling factor and then changing the sampling rate of the signal

correspondingly. See (Serra, 1997a) for examples and more details on time-scale modifications and problems that arise when the frequency resolution of the STFT analysis is not sufficient. The problem of "phasiness" in time-scale modifications is discussed by Laroche and Dolson (1997) where also a phase synchronization is proposed.

The phase vocoder works best when used with harmonic and static or slowly changing tones. It has difficulties with noisy and rapidly changing sound signals. These signals can be modeled better with a tracking phase vocoder or the spectral modeling synthesis described in Section 3.5.

3.3 Source-Filter Synthesis

In source-filter synthesis the sound waveform is obtained by filtering an excitation signal with a time-varying filter. The method is sometimes called subtractive synthesis. The technique has been used especially to produce synthetic speech; see, e.g., (Moorer, 1985) for more details and references, but also for musical applications (Roads, 1995).

A block diagram of the method is depicted in Figure 3.5. The idea is to have a broadband or harmonically rich excitation signal which is filtered to get the desired output, as opposed to additive synthesis where the waveform is composed as a sum of simple sinusoidal components. In theory, an arbitrary periodic bandlimited waveform can be generated from a train of impulses by filtering. Complex waveforms are simple to generate by using a complex excitation signal. A new method to generate bandlimited pulse trains is introduced by Stilson and Smith (1996).

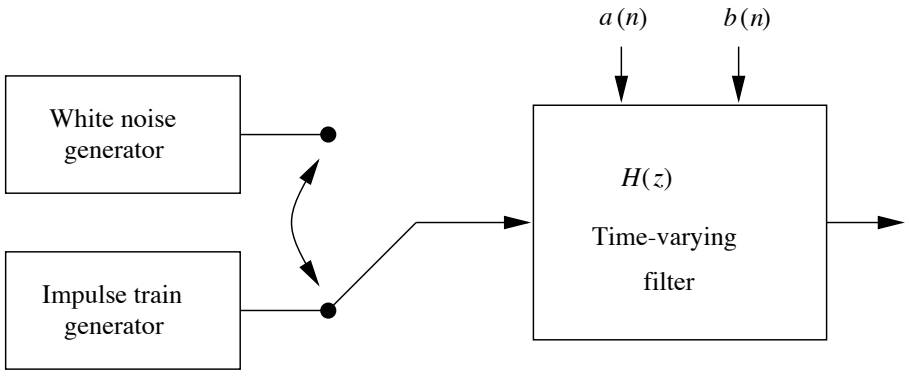

Figure 3.5: Source-filter synthesis. The transfer function of the time-varying filter $H(z)$ is described by filter coefficients $a(n)$ and $b(n)$.

The human voice production mechanism can be approximated as an excitation sound source feeding a resonating system. When source-filter synthesis is used to synthesize speech, the sound source generates either a periodic pulse train or white noise depending on whether the speech is voiced or unvoiced, respectively. The filter

$$H(z) = \frac{\sum_{k=0}^{K} b_k z^{-k0}}{1 + \sum_{l=1}^{L} a_l z^{-l}} \tag{3.2}$$

models the resonances of the vocal tract. The coefficients $a(n)$ and $b(n)$ of the filter vary with time thus simulating the movements of lips, the tongue and other parts of the vocal tract. The periodic pulse train simulates the glottal waveform. Many traditional musical instruments have a stationary or slowly time-varying resonating system, and source-filter synthesis can be used to model such instruments. The method has also been used in analog synthesizers. When applied to speech or singing, the method can be interpreted as physical modeling of the human sound production mechanism.

The excitation signal and the filter coefficients fully describe the output waveform. If only a wideband pulse train and noise is used, it is enough to decide between unvoiced and voiced sounds. If the pulse form is fixed, only the period, i.e., the fundamental frequency of the pulse train, remains to be determined.

Detection of the pitch is not a trivial problem and it has been studied extensively mainly by speech researchers. Pitch detection methods can be divided into five categories: time-domain methods, autocorrelation-based methods, adaptive filtering, frequency-domain methods, and models of the human ear. These are discussed, e.g., in (Roads, 1995) with references.

The filter coefficients can be efficiently computed by applying *linear predictive* (LP) analysis. The basic idea of LP is that it is possible to design an all-pole filter whose magnitude frequency response closely matches that of an arbitrary sound. The difference between STFT and LP is that LP measures the envelope of the magnitude spectrum whereas the STFT measures the magnitude and phase at a large number of equally spaced points. LP is a parametric method whereas STFT is non-parametric and LP is optimal in that it is the best match of the spectrum in the minimum-squared-error sense. The method is discussed in detail by Makhoul (1975).

The fundamental frequency of the waveform depends only on the fundamental frequency of the pulse train. So the timing and the fundamental frequency can be varied independently. Also, the synthesis system can be excited with a complex waveform, thus creating new sounds that have characteristics of the excitation sound as well as the resonating system.

Although, in theory, arbitrary signals can be produced, source-filter synthesis is not a very robust representation for generic wideband audio or musical signals. Ways to improve the sound quality are discussed by Moorer (1979).

3.4 McAulay-Quatieri Algorithm

An analysis-based representation of sound signals suitable for *additive synthesis* has been presented by McAulay and Quatieri (1986). The McAulay-Quatieri (MQ) algorithm originated from research of speech signals but it was already reported in the first study that the algorithm is capable of synthesizing a broader class of sound signals (McAulay and Quatieri, 1986). Other algorithms of parameter estimation

for additive synthesis exist (Risset, 1985; Smith and Serra, 1987) and the MQ and related algorithms have been utilized in many spectral modeling systems (Serra and Smith, 1990; Fitz and Haken, 1996).

In the MQ algorithm the original signal is decomposed into signal components that are resynthesized as a set of sinusoids. The k^{th} signal component at time location l is represented as a set of triplets $\{A_k^l, \omega_k^l, \phi_k^l\}$ constituting three types of trajectories, namely, amplitude, frequency, and phase trajectories, that are used in the synthesis stage. The time locations l are determined by the hop size parameter N_{hop} of the STFT as

$$l = nN_{\text{hop}},\ n = 0, 1, 2, \ldots$$

The MQ algorithm can be programmed to adapt to the analysis signal, e.g., the number of detected signal components and the hop size parameter can vary in time.

The method is efficient in presenting harmonic or voiced signals with little noise or transitions. If noisy or unvoiced signals are to be reproduced, a large number of sinusoids is needed. In the example described in the study by McAulay and Quatieri (1986), the maximum number of the detected peaks was set to 80 with the sampling frequency of 10 kHz and hop size of 10 ms. It was shown that waveforms of harmonic signals are reproduced accurately, whereas the reproduced waveforms of noisy signals do not resemble the original well.

The analysis part of the MQ algorithm uses the STFT to obtain a representation for each signal component. The input signal $x(n)$ is windowed in the time domain to a length ranging typically between 4 and 30 ms. A *discrete Fourier transform* (DFT) is computed from the windowed signal $x_{\text{w}}(n)$. Peaks in the complex spectrum $X_{\text{w}}(n)$ corresponding to sinusoidal signal components are detected and they are used to obtain the amplitude, frequency, and phase trajectories that compose the signal representation. The analysis steps are elaborated further in the following subsections.

3.4.1 Time-Domain Windowing

Choosing a proper window function is a compromise between the width of the main lobe (frequency resolution of each signal component) and attenuation of the side lobes (spreading in the frequency domain). A detailed discussion of window functions is given by Harris (1978) and Nuttall (1981). In the original study, McAulay and Quatieri utilize a Hamming window. In the frequency domain, it has a 43 dB attenuation of the largest side lobe and an asymptotic decay of 6 dB/octave (Nuttall, 1981). The same window function is also used in the examples presented in this section.

The length of the window function determines the time resolution of the analysis. The length of the Hamming window function should be at least 2 1/2 times the period of the fundamental frequency (McAulay and Quatieri, 1986). The window length can be time-varying to adapt to the analyzed signal.

It is beneficial to increase the frequency resolution of the DFT by increasing

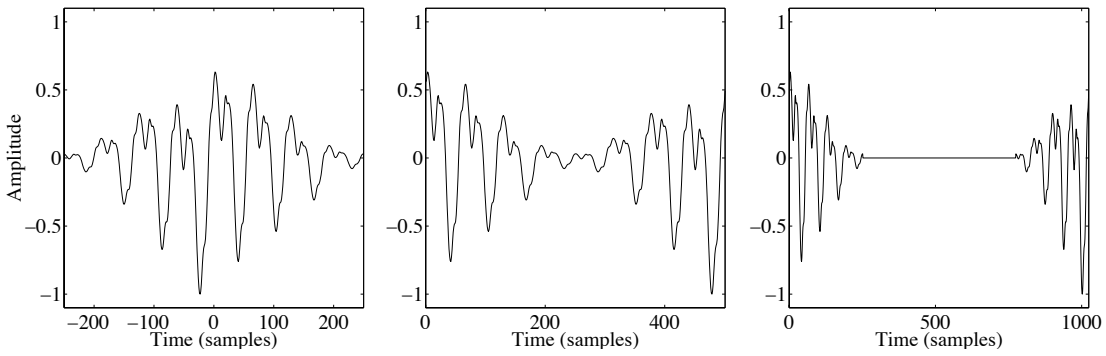

Figure 3.6: An example of zero-phase windowing. On the left a signal is windowed about the time origin. An equivalent signal for the DFT is displayed in the middle. In zero padding the zeros are inserted in the middle of the signal as shown on the right.

the length of the windowed signal by concatenating the windowed signal with zeros (Smith and Serra, 1987). This is called zero padding and typically the length of the windowed signal is increased to a power of two to allow for the use of the *fast Fourier transform* (FFT), an efficient algorithm for the computation of the DFT. Note, however, that the frequency resolution in the analysis is further improved by applying an interpolation scheme proposed by (Smith and Serra, 1987).

For the detection of the phase values it is important to use *zero-phase windowing* to avoid a linear trend in the phase spectra (Serra, 1989). An example of zero-phase windowing is shown in Figure 3.6. On the left a portion of a guitar signal is windowed using a Hamming window with a length of 501 samples, and the windowed signal is centered about the time origin at indices $-250, -249, \ldots, 251$. In practice, the circular properties of the DFT are used and the left half (indices $-250, \ldots, -1$) of the signal is positioned at time indices $252, \ldots, 501$, as shown in the middle of Figure 3.6. On the right the signal is zero-padded to a length of 1024. Notice that the zeros are inserted in the middle of the wrapped signal.

3.4.2 Computation of the STFT

The STFT is composed as a series of DFTs computed on windowed signal portions separated in time by the hop size parameter N_{hop}. A value varying between $N_{\text{win}}/2$ and $N_{\text{win}}/16$ is typically used for the hop size parameter, where N_{win} is the length of the time-domain analysis window.

A DFT is performed on the zero-phase-windowed and zero-padded signal $x_{\text{w}}(n)$. The DFT returns a complex sequence $X_{\text{w}}(k)$ of length of the original signal. Sequence $X_{\text{w}}(k)$ is a frequency domain representation of the signal, and it is centered around the frequency origin. As a result of the analyzed signal being real, the values at positive and negative frequencies are complex conjugates, i.e., $X_{\text{w}}(k) = X_{\text{w}}^*(-k)$. In the following we will only consider the values of $X_{\text{w}}(k)$ at positive frequencies. Sequence $X_{\text{w}}(k)$ is interpreted as magnitude and phase spectra of the windowed signal by changing to *polar coordinates*. An example of a single STFT frame is shown in Figure 3.8 where the magnitude (top) and phase (bottom) spectra of a windowed

guitar signal are plotted.

3.4.3 Detection of the Peaks in the STFT

The peaks in the magnitude spectrum correspond to prominent signal components that are modeled as sinusoidal signals. In general the determination of whether a peak is a prominent one is rarely trivial. In the case of an harmonic tone, the harmonic structure of the magnitude spectrum can be exploited. The fundamental frequency of the recorded signal is estimated and it suffices to search for the local maxima of each magnitude spectrum in the vicinities of the multiples of the fundamental frequency.

A peak detection is best performed in the dB scale (Serra, 1989). A local maximum in the vicinity of a harmonic frequency can be detected by first determining the range of the search. Typically, a the peak corresponding to the k^{th} partial is searched for in the range $[(k-1/4)\hat{f}_0, (k+1/4)\hat{f}_0]$. The maximum value in this range is detected and if it is a local maximum, it is marked as a peak. This procedure is carried out for every partial in every frame.

3.4.4 Removal of Components below Noise Threshold Level

The peaks detected in the previous section will contain values that do not correspond to the sinusoidal components. It is therefore essential to remove the detected peaks with values below a chosen noise threshold value. Typically, this value should be frequency dependent. The noise level can be estimated in the recorded signals, e.g., in the pauses of speech.

If a single tone is analyzed, the sinusoidal components can be detected starting from the end of the signal (Smith and Serra, 1987). Then, the amplitude values can be set to a zero value before a distinctive signal component is found.

3.4.5 Peak Continuation

After the peaks below the noise threshold level are removed, a peak continuation algorithm is utilized to produce the amplitude and frequency trajectories corresponding to the sinusoidal components of the original signal. It is assumed that the sinusoids are fairly stationary between frames, and thus the algorithm assigns a peak for an existing trajectory if their frequency values are close enough. A parameter for the maximum frequency deviation f_{D} between consecutive frames is used as a limiting criterion. If there is no existing trajectory for that component in the previous frame, a new trajectory is started, i.e., it is "born" (McAulay and Quatieri, 1986). This is done by creating a triplet in the previous frame with zero amplitude, the same frequency, and a phase value that is computed by subtracting a phase shift in one frame from the detected phase value. Similarly, if no peak matching an existing trajectory is found, that trajectory is "killed" (McAulay and Quatieri, 1986). In this case a triplet with zero amplitude, the same frequency, and a shifted phase value is inserted in the next frame.

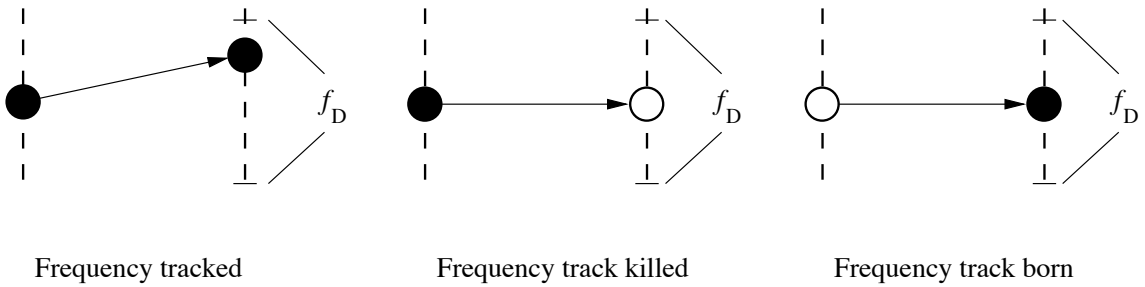

Figure 3.7: An example of the peak continuation algorithm, after (McAulay and Quatieri, 1986). On the left a match is found, and the peak is assigned to the track. In the middle no peak within the maximum deviation f_D is found and the track is killed. On the left, a peak is detected that does not match any peaks in the previous frame and a track is born.

An example of the "nearest-neighbor" peak continuation algorithm is illustrated in Figure 3.7. On the left, a frequency value is detected that is within the frequency deviation threshold f_D and the peak is assigned to the corresponding track. In the middle, no peak with a frequency value within the limit is found, and the track is killed. On the right, a new track is born, i.e., a peak is detected that does not correspond to any of the peaks in the previous frame.

3.4.6 Peak Value Interpolation and Normalization

A better frequency resolution of the peak detection can be obtained by applying a parabolic interpolation scheme proposed by Smith and Serra (1987) and detailed by Serra (1989). In parabolic interpolation a parabola is fitted to the three points consisting of the maximum and the adjacent values. A point corresponding to the maximum value of the parabola is detected. The point yields the amplitude and the frequency values of the corresponding peak. The phase value is detected in the phase spectrum by interpolating linearly between the adjacent frequency points enclosing the location of the peak.

An example of detection of the peaks is shown in Figure 3.8. The peaks above -60 dB in magnitude are detected and denoted with a cross ($\times$) in the magnitude and phase spectra. A zoom to the phase spectrum in Figure 3.9 shows the efficiency of the zero-phase windowing. The phase values are almost constant in the vicinity of an harmonic component. This greatly reduces the estimation error of the detected phase value.

The effect of the time-domain windowing has to be compensated for in the amplitude values. The normalization factor c_w of the window function can be computed by solving (Serra, 1989)

$$c_w \sum_{-\infty}^{\infty} w(n) = c_w \sum_{m=0}^{N-1} w(m) = 1,$$

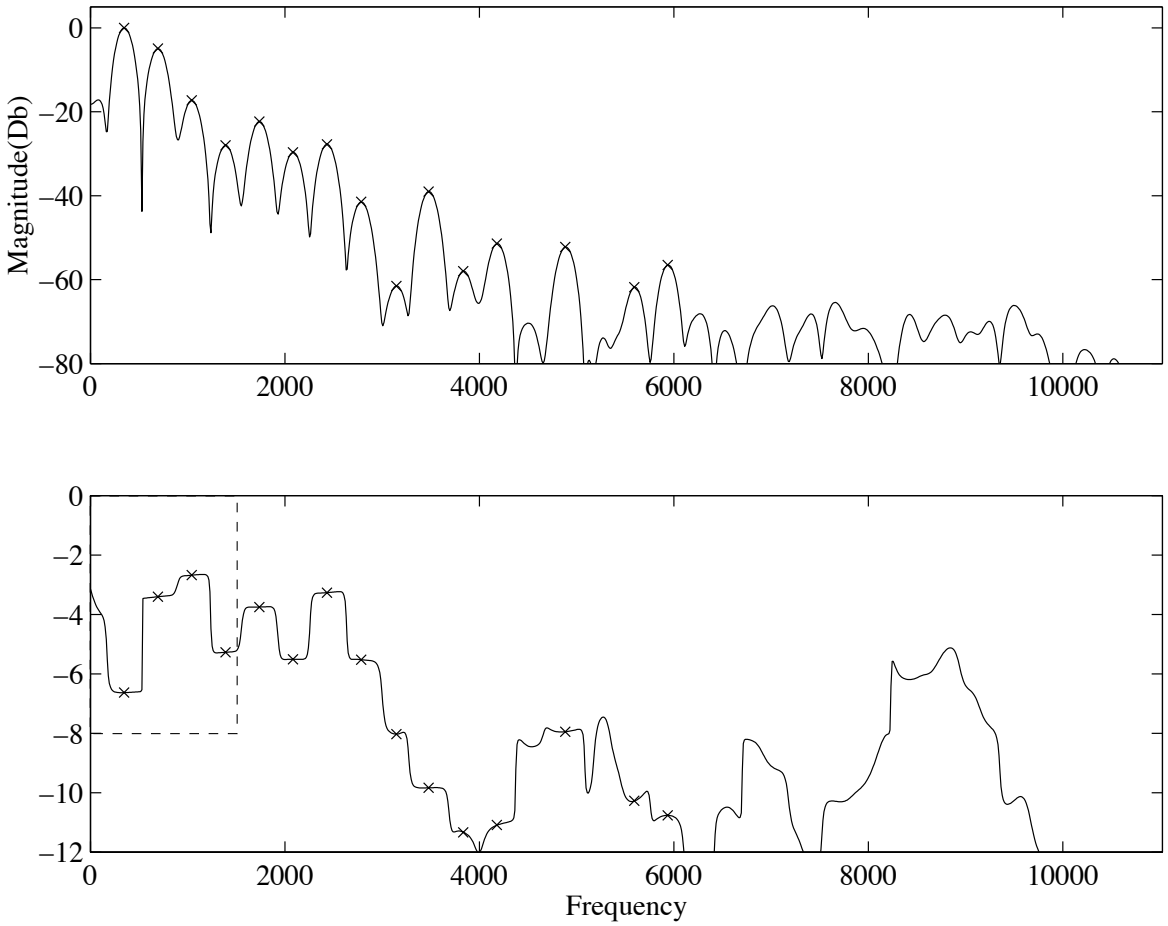

Figure 3.8: Magnitude (top) and phase (bottom) spectra corresponding to a frame in STFT. The locations of peaks corresponding to harmonic components are denoted with ×. A zoom to the dashed box in the phase spectrum is shown in Figure 3.9.

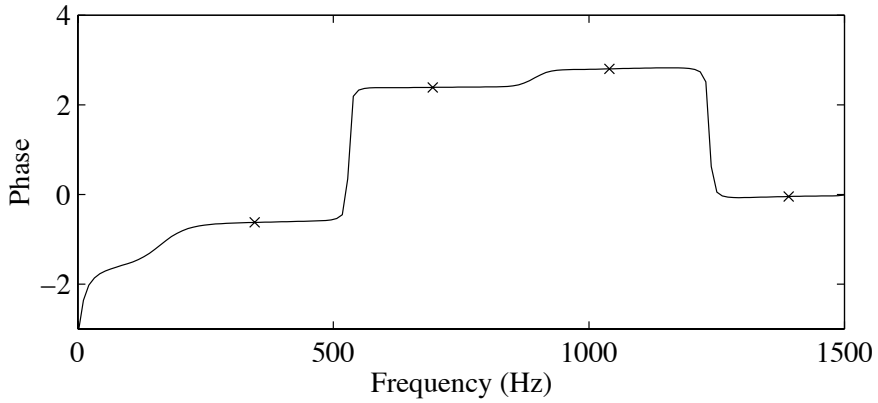

Figure 3.9: A detail of phase spectra in a STFT frame. Zero-phase windowing yields flat portions of the phase spectrum in the vicinity of a harmonic component.

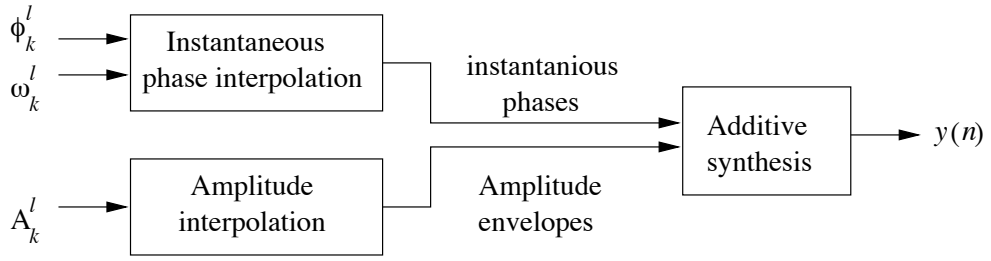

Figure 3.10: Additive synthesis of the sinusoidal signal components, after (McAulay and Quatieri, 1986). Linear interpolation is used for the amplitude envelope and cubic interpolation for the instantaneous phase of each partial

which yields

$$c_{\mathrm{w}} = \frac{1}{\sum_{m=0}^{N-1} w(m)}. \tag{3.3}$$

Furthermore, in the DFT half of the energy of each sinusoid is on the negative frequencies and thus the amplitude value of the detected peak has to be multiplied by a factor of 2. The overall normalization factor is thus

$$c = \frac{2}{\sum_{m=0}^{N-1} w(m)},$$

where $w(m)$ the window function of length N.

3.4.7 Additive Synthesis of Sinusoidal Components

The additive synthesis of the sinusoidal signal components is pictured in Figure 3.10. In this case a phase-included additive synthesis is used, i.e., the signal is approximated as

$$x(n) \approx \tilde{x}(n) = \sum_{k=1}^{N_{\mathrm{sig}}(n)} \tilde{A}_k(n) \cos(\tilde{\theta}_k(n)), \tag{3.4}$$

where $\tilde{A}_k(n)$ is the amplitude envelope and $\tilde{\theta}_k(n)$ is the instantaneous phase of the k^{th} signal component. Notice that the number of signal components $N_{\mathrm{sig}}(n)$ may depend on time n. This implies that the number of signal components adapts to the analyzed signal.

The analysis stage provides the amplitude, the frequency, and the phase trajectories of the signal components. The values of each triplet $\{A_k^l, \omega_k^l, \phi_k^l\}$ correspond to the detected values of amplitude, frequency and phase of the k^{th} signal component at frame l. They are separated in time by an amount determined by the hop size parameter N_{hop} of the STFT. The trajectories have to be interpolated from frame to frame in order to obtain the amplitude envelopes and the instantaneous

phases for additive synthesis. Amplitude trajectory A_k^l of the k^{th} signal component is interpolated linearly from frame $l-1$ to frame l to obtain instantaneous amplitude

$$\tilde{A}_k(m) = A_k^{l-1} + \frac{A_k^l - A_k^{l-1}}{N_{\text{hop}}} m, \quad m = 0, 1, \ldots, N_{\text{hop}} - 1, \tag{3.5}$$

This procedure is applied to all frame boundaries to obtain the amplitude envelopes $\tilde{A}_k(n)$ for the additive synthesis.

Both the detected frequency and phase affect the instantaneous phase $\tilde{\theta}_k(m)$. Thus there are four variables, namely, ω_k^{l-1}, φ_k^{l-1}, ω_k^l, and φ_k^l, that have to be involved in the interpolation. As proposed by McAulay and Quatieri (1986), cubic interpolation can be used with

$$\tilde{\theta}_k(m) = \zeta + \gamma m + \delta m^2 + \eta m^3. \tag{3.6}$$

This equation is solved as

$$\tilde{\theta}_k(m) = \varphi_k^{l-1} + \omega_k^{l-1} m + \delta m^2 + \eta m^3, \tag{3.7}$$

where

$$\begin{bmatrix} \delta(M) \\ \eta(M) \end{bmatrix} = \begin{bmatrix} \frac{3}{N_{\text{hop}}^2} & \frac{-1}{N_{\text{hop}}} \\ \frac{-2}{N_{\text{hop}}^3} & \frac{1}{N_{\text{hop}}^2} \end{bmatrix} \begin{bmatrix} \theta_k^l - \theta_k^{l-1} - \omega_k^{l-1} T + 2\pi M \\ \omega_k^l - \omega_k^{l-1} \end{bmatrix} \tag{3.8}$$

The value of M is chosen so that the instantaneous phase function is maximally smooth. This is done by taking M to be the integer value closest to x, when (McAulay and Quatieri, 1986)

$$x = \frac{1}{2\pi}[\theta_k^{l-1} + \omega_k^{l-1} - \theta_k^l + \frac{N_{\text{hop}}}{2}(\omega_k^l - \omega_k^{l-1})] \tag{3.9}$$

The instantaneous phase $\tilde{\theta}_k(m)$ is obtained by applying Equation 3.7 to all frame boundaries. The synthetic signal is now computed as

$$x_{\text{sin}}(n) = \sum_{k=1}^{N_{\text{sin}}(n)} \tilde{A}_k(n) \cos(\tilde{\theta}_k(n)). \tag{3.10}$$

The residual signal corresponding to the stochastic component (Serra, 1989) is obtained as

$$x_{\text{res}}(n) = x(n) - x_{\text{sin}}(n). \tag{3.11}$$

The stochastic signal contains information on both the steady-state noise and rapid transients in the signal.

3.5 Spectral Modeling Synthesis

The Spectral Modeling Synthesis (SMS) technique was developed in the late 1980's at CCRMA, Stanford University. Serra (1989) developed a method for decomposing

a sound signal into deterministic and stochastic components. The deterministic component can be obtained by using the MQ algorithm (McAulay and Quatieri, 1986), (Section 3.4) or by using a magnitude-only analysis. The deterministic part is subtracted from the original signal either in the time or the frequency domain to produce a residual signal which corresponds to the stochastic component. The residual signal can be represented efficiently using methods discussed in this section. In (Serra and Smith, 1990) a detailed discussion of the magnitude-only analysis/synthesis is given and a description of that system will be given here. The method is also discussed in (Serra, 1997b). The analysis scheme with phase included can be used to obtain the residual signal by a time-domain subtraction, as discussed in Section 3.4. This method is used in various musical analysis applications, including analysis of recorded plucked string tones to derive proper excitation signals for physical modeling of plucked string tones (Tolonen and Välimäki, 1997). The interested reader is also referred to work by Evangelista (1993, 1994) where a wavelet representation is introduced that is suitable for representing separately pseudo-periodic and aperiodic components of a signal.

The SMS technique is based on the assumption that the input sound can be represented as a sum of two signal components, namely, the deterministic and the stochastic component. By definition a deterministic signal is any signal that is fully predictable. The SMS model, however, restricts the deterministic part to sinusoidal components with piecewise linear amplitude and frequency variations. This affects the generality of the model and some sounds cannot be accurately modeled by the technique. In the method the stochastic component is described by its power spectral density. Therefore, it is not necessary to preserve phase information of the stochastic component. The stochastic component can be efficiently represented by the magnitude spectrum envelope of the residual of each DFT frame.

The SMS model consists of an analysis part and a synthesis part described in the following two subsections.

3.5.1 SMS Analysis

The analysis part is used to map the input signal from the time domain into the representation domain, as is depicted in Figure 3.11. The stochastic representation is given by the spectral envelopes of the stochastic component of the input signal. The envelopes are calculated from each DFT frame and they can be efficiently described using a piece-wise linear approximation (Serra and Smith, 1990). The deterministic representation is composed of two trajectories, the frequency and the magnitude trajectory.

The analysis part is fairly similar to that of the MQ algorithm. The first step is to calculate the STFT of each windowed portion of the signal. The STFT produces a series of complex spectra from which the magnitude spectra is calculated. From each spectrum the prominent peaks are detected and the peak trajectories are obtained utilizing a peak continuation algorithm.

The stochastic component is obtained by subtracting the deterministic compo-

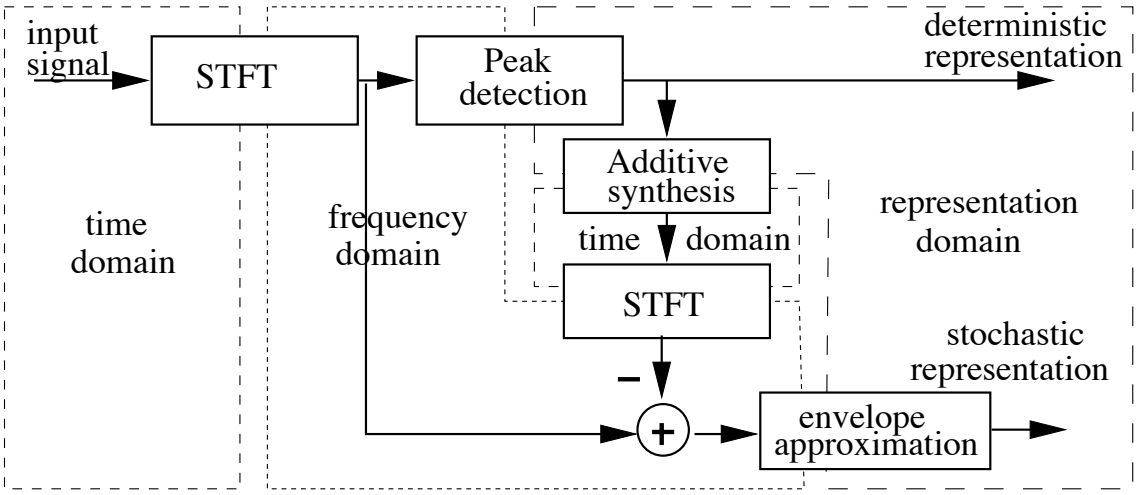

Figure 3.11: The analysis part of the SMS technique, after (Serra and Smith, 1990).

nent from the signal in the frequency domain. First, the deterministic waveform is computed from the peak trajectories. Then the STFT of the deterministic waveform is calculated similarly to the one obtained from the original signal. By calculating the difference of the magnitude spectra of the input and the deterministic signal, the corresponding magnitude spectrum of the stochastic component is obtained for each windowed waveform portion. The envelopes of these spectra are then approximated using a line-segment approximation. These envelopes form the stochastic representation.

3.5.2 SMS Synthesis

The synthesis part of the technique maps a signal from the representation domain into the time domain. This process is illustrated in Figure 3.12. The deterministic component of the signal is obtained by a magnitude-only additive synthesis. An optional transformation can be used to alter the synthesized signal. This allows the production of new sounds using the information of the analyzed signal, for example, the duration of the signal (tempo) can be varied without changing the peak frequencies (key) of the signal. Similarly, the frequencies can be transposed without influencing the duration.

The stochastic signal is computed from the spectral envelopes, or their modifications, by calculating an inverse STFT. The phase spectra are generated using a random number generator.

The SMS method is very efficient in reducing the control data and computational demands. The method is general and can be applied to many sounds. There are some problems with the use of STFT; is not sufficiently well time-localized and short transient signals will be spread in the time domain (Goodwin and Vetterli, 1996).

In the next section, a method for improving the accuracy on transient signals is presented.

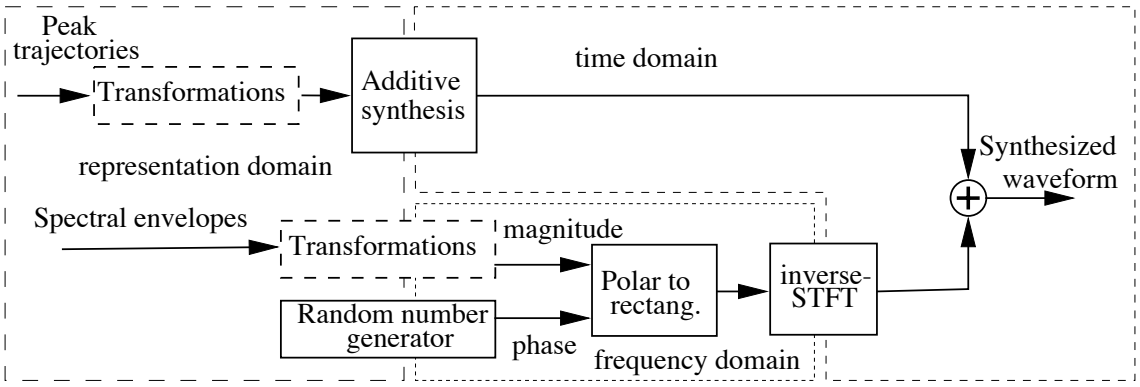

Figure 3.12: The synthesis part of the SMS technique, after (Serra and Smith, 1990).

3.6 Transient Modeling Synthesis

An extension to Spectral Modeling Synthesis discussed in the previous section is presented by Verma et al. (1997). In this approach, the residual signal obtained by subtracting the sinusoidal model from the original signal is represented in two parts, transients and steady noisy components. Transient Modeling Synthesis (TMS) provides a parametric representation of the transient components.

TMS is based on the duality between the time and the frequency domains (Verma et al., 1997). Transient signals are impulsive in the time domain, and thus they are not in a form that is easily parameterizable. However, with a suitable transformation, impulsive signals are presented as frequency domain signals that have a sinusoidal character. This implies that sinusoidal modeling can be applied in the frequency domain to obtain a parametric representation of the impulsive signal.

In the next subsection, the principles utilized in TMS are presented. Second, the structure of the TMS system is described.

3.6.1 Transient Modeling with Unitary Transforms

The idea is to apply sinusoidal modeling on a frequency domain signal, that corresponds to rapid changes in the time domain signal. For sinusoidal modeling we wish to have a real-valued signal. Thus, in this case the DFT is not an appropriate choice since it produces a complex-valued spectrum. The *discrete cosine transform* (DCT) provides a mapping in which an impulse in the time domain maps into a real-valued sinusoid in the frequency domain. The DCT is defined as

$$C(k) = \alpha(k) \sum_{n=0}^{N-1} x(n) \cos\left[\frac{(2n+1)k\pi}{2N}\right], \quad n, k \in 0, 1, 2, \ldots N-1, \tag{3.12}$$

where N is the length of the transformed signal $x(n)$. Coefficients $\alpha(k)$ are

$$\alpha(k) = \begin{cases} \sqrt{\frac{1}{N}} & \text{for} k = 0, \\ \sqrt{\frac{2}{N}} & \text{for} k = 1, 2, \ldots, N-1 \end{cases} \tag{3.13}$$

It is obvious from Equation 3.12 that if $x(n) = \delta(n - l)$, the frequency-domain representation is

$$C(k) = \cos\left[\frac{(2l+1)\pi}{2N}k\right],$$

i.e., it is a sinusoid with a period depending on the location l of the time-domain impulse $\delta(n - l)$. Thus, Equation 3.12 implies that impulsive time-domain signals, e.g., corresponding to attacks of tones, produce a DCT that has strong sinusoidal components, whereas steady-state signals produce a DCT with little or no sinusoidal components.

Equation 3.12 is exemplified in Figures 3.13 and 3.14. On the top and in the middle of Figure 3.13 an impulse-like time-domain signal and its DCT are illustrated, respectively. The DCT is clearly a sinusoid with an amplitude envelope that varies with frequency. The waveform of the DCT can be represented by applying sinusoidal modeling. Notice that in this case the sinusoidal analysis is performed on a frequency domain signal. On the bottom of Figure 3.13, the magnitude of the complex-valued DFT computed on the sinusoidal DCT is presented. Notice that only values corresponding to the positive indexes of the DFT are shown. This plot shows that the period of the DCT corresponds to the location of the impulse.

To demonstrate the duality principle applied in TMS, similar plots corresponding to a slowly-varying signal are presented in Figure 3.14. In this case, an exponentially decaying sinusoid (top) produces an impulsive DCT (middle). Again, the magnitude of the DFT (bottom) computed on the DCT closely follows the amplitude envelope of the original signal. Observe that in both Figures 3.13 and 3.14 the DFT does not provide a parametric representation of the transients in the residual signal. The magnitude plots are only shown to clarify the unitary transforms applied in the TMS.

3.6.2 TMS System

As mentioned above, TMS is an extension to the SMS system discussed in Section 3.5 in that the residual signal is further decomposed into two components corresponding to transient and noisy parts of the original signal. In this context, only the extension part of the TMS is presented. A block diagram of the system is illustrated in Figure 3.15 (Verma et al., 1997). First, a block DCT is computed on the residual signal. The length of the DCT block is chosen to be sufficiently large so that the transients are compact entities within the block. A block size of one second has found to be a good choice (Verma et al., 1997). The transient detection block is optional and it can be used to determine the regions of interest in the sinusoidal analysis. The SMS is applied to the frequency domain DCT signal, and the obtained representation is used to synthesize the transients and subtract them from the residual signal in the time domain. The residual signal is now expressed as components corresponding to slowly-varying noise and transients. The analysis steps are elaborated further in the following discussion.

The transient detection block is optional and the system can operate without it. However, it is useful since if the approximate locations of the transients in the time

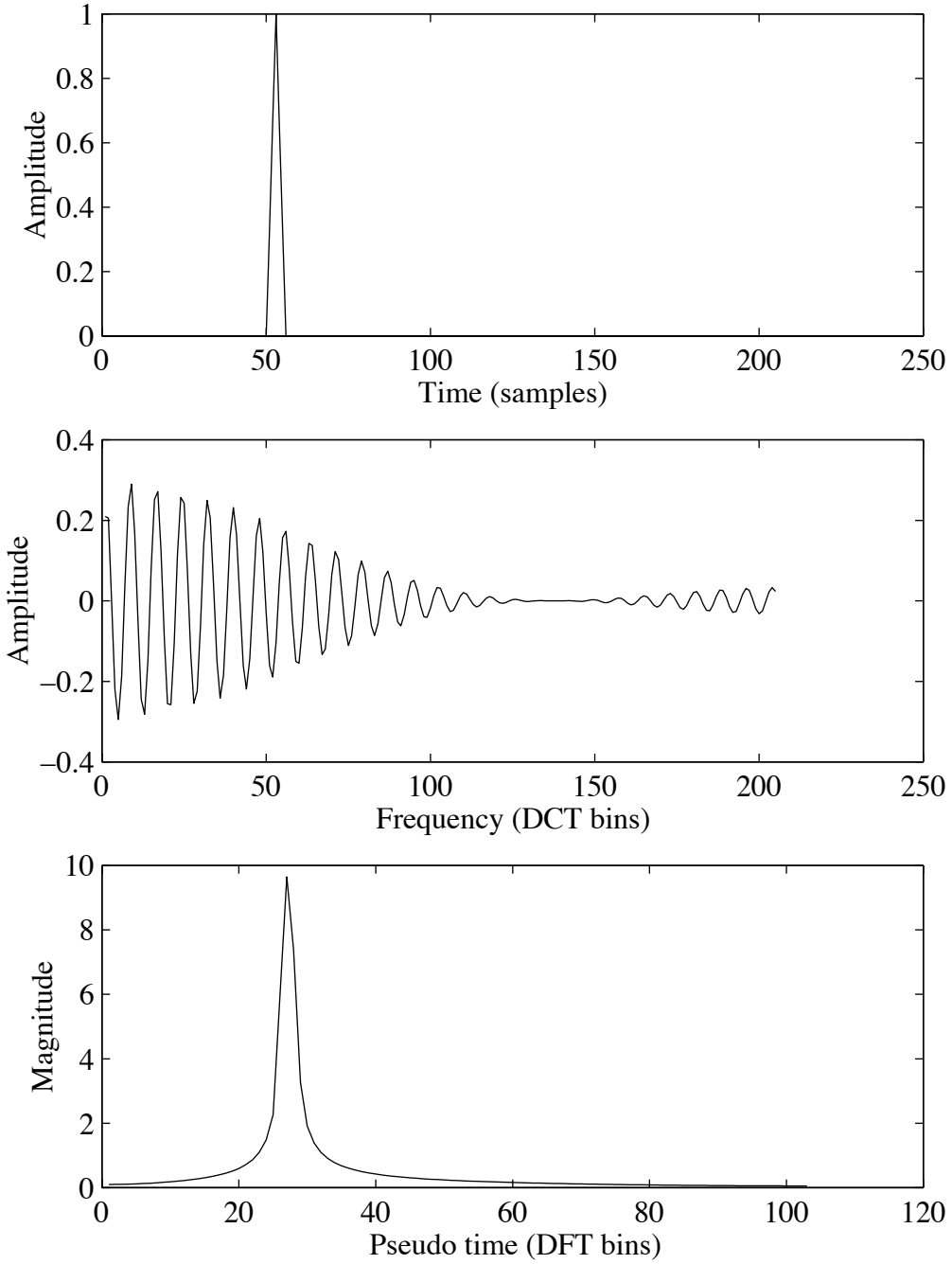

Figure 3.13: An example of TMS. An impulsive signal (top) is analyzed. A DCT (middle) is computed, and an DFT (magnitude in bottom) is performed in the DCT representation.

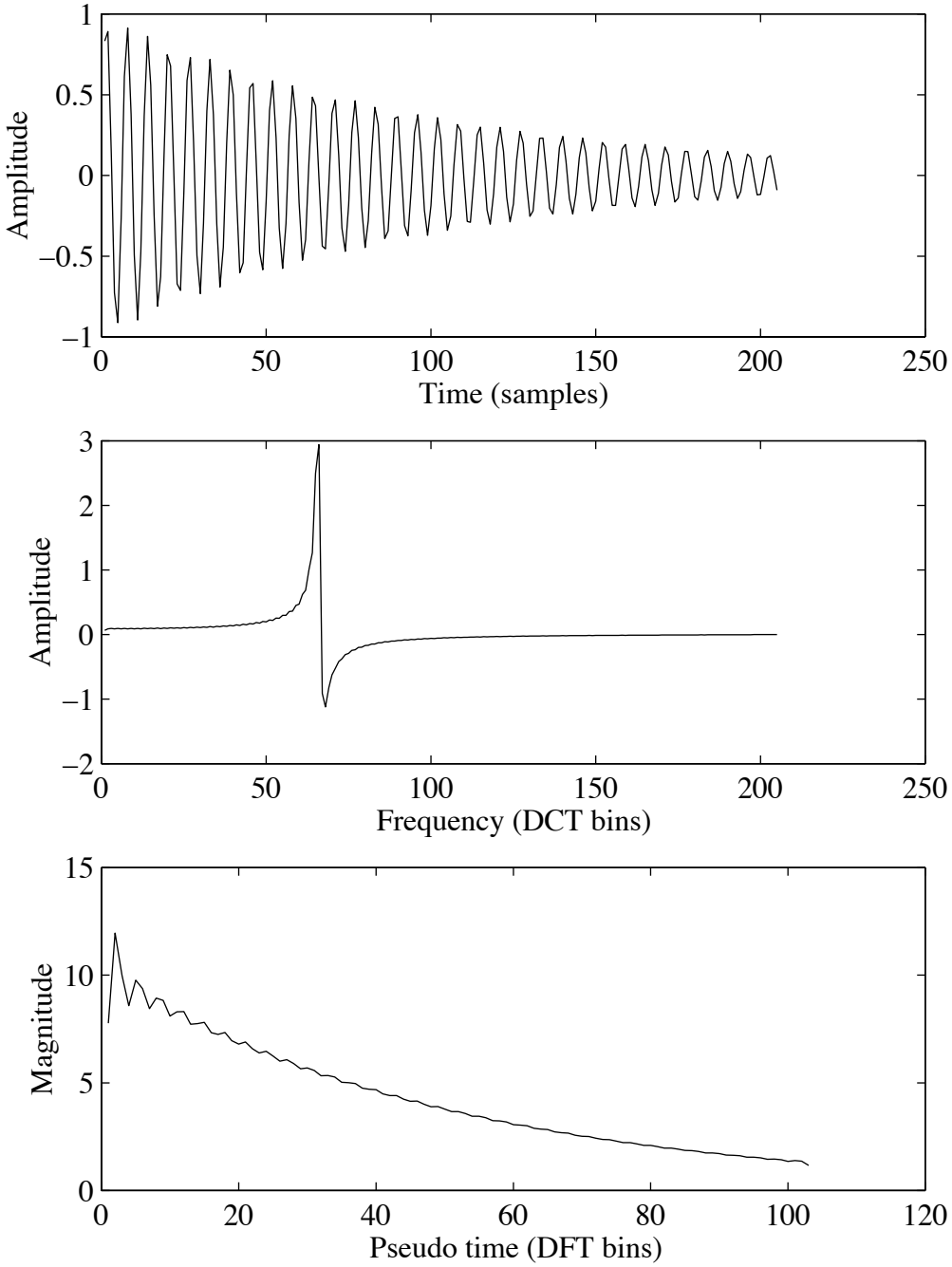

Figure 3.14: An example of TMS. A slowly-varying signal (top) is analyzed. A DCT (middle) is computed, and an DFT (magnitude in bottom) is performed in the DCT representation.

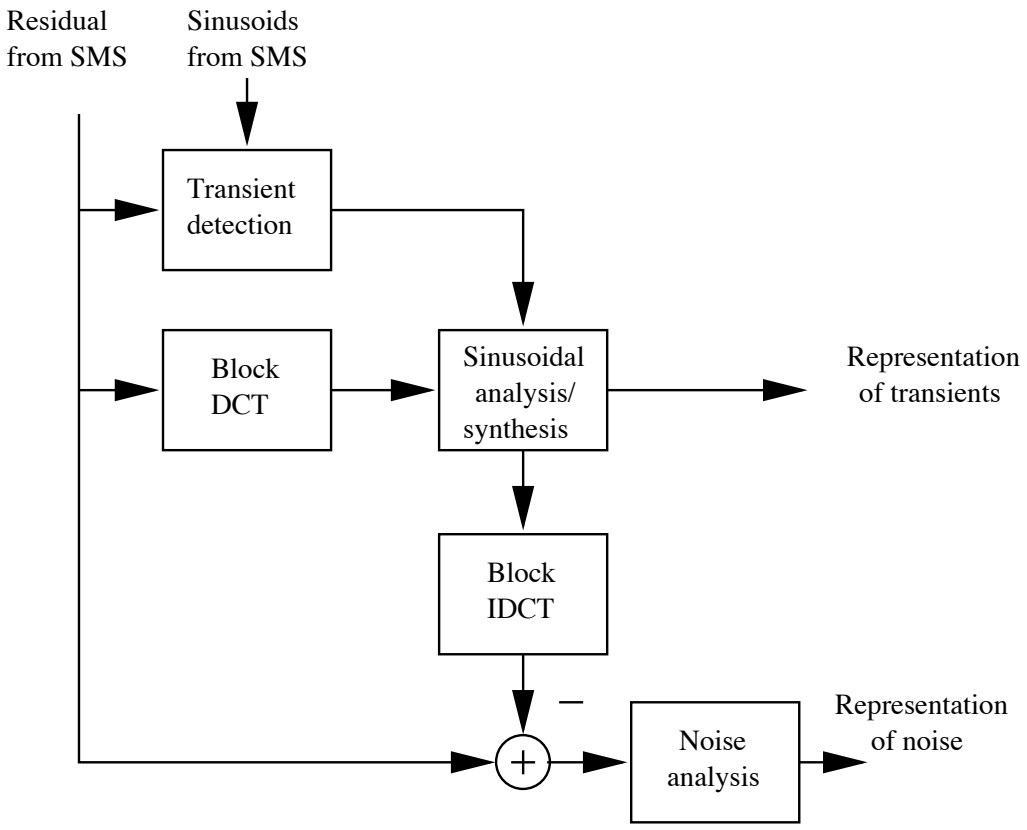

Figure 3.15: A block diagram of the transient modeling part of the TMS system, after (Verma et al., 1997).

domain are known, the sinusoidal modeling operating on the DCT can be restricted to only select those components that correspond to the transient positions. The transients are detected in the residual signal by computing a ratio of the energies of the residual and sinusoidal signals as a function of time (Verma et al., 1997). In practice this is done within a DCT block by first computing the energies of the sinusoidal and residual signals as

$$E_{\text{sin}} = \sum_{n=0}^{N-1} |x_{\text{sin}}(n)|^2 \quad \text{and} \quad E_{\text{res}} = \sum_{n=0}^{N-1} |x_{\text{res}}(n)|^2, \tag{3.14}$$

where N is the length of the DCT. The instantaneous energies of the sinusoidal and the residual signal are approximated by computing the energy within a short window that is slid in time within the DCT block. This is expressed as

$$e_{\text{sin}}(k) = \sum_{n=k-\frac{L}{2}}^{k+\frac{L}{2}} |x_{\text{sin}}(n)|^2 \tag{3.15}$$

and

$$e_{\text{res}}(k) = \sum_{n=k-\frac{L}{2}}^{k+\frac{L}{2}} |x_{\text{res}}(n)|^2, \tag{3.16}$$

for $k = 0, N_{\text{hop}}, 2N_{\text{hop}}, ..., N-1$, where L is the length of the sliding window, N_{hop} is the hop size parameter, and $x(n)$ is the signal within the DCT block zero-padded in a manner that it is defined in the region of computation.

The locations of the transients are determined to be in the vicinity of positions k where the ratio of normalized instantaneous energies of the residual and the sinusoidal signal is above a given threshold value. This is expressed explicitly as

$$\frac{e_{\text{res}}(k)/E_{\text{res}}}{e_{\text{sin}}(k)/E_{\text{sin}}} > R_{\text{thr}} \tag{3.17}$$

After the locations of the transients have been detected, the sinusoidal model is restricted to estimating periodic spectral components corresponding to the estimated locations. If the transient detection is not used, SMS is applied on the whole period range of the spectral representation. The spectral modeling parameters are used to resynthesize the transient signal components, and subtract them from the residual signal. The obtained signal lacks the rapid variations and can therefore be approximated as slowly-varying noise.

3.7 Inverse FFT (FFT^{-1}) Synthesis

Inverse FFT (FFT^{-1}) synthesis is presented in (Rodet and Depalle, 1992a) and (Rodet and Depalle, 1992b). In this method, additive synthesis is used in the frequency domain, i.e., all the signal components are added together as spectral envelopes composing a series of STFT frames. The waveform can be constructed by calculating the inverse FFT of each frame. The *overlap-add* method is used to attach the consecutive frames to each other.

Sinusoidal signals are simple to represent in the frequency domain. A windowed sinusoid in the frequency domain is a scaled and shifted version of the DFT of the window function. For the synthesis method to be computationally efficient, the DFT of the windowing function should have low sidelobes, i.e., it should have few significant values (Rodet and Depalle, 1992a). On the other hand, the frequency and the amplitude of the sinusoid in consecutive frames are linearly interpolated. This requirement yields a triangular window. The DFT of a triangular window, however, has quite significant sidelobes and is not appropriate. A solution to this problem is to use two windows, one in the frequency domain and one in the time domain (Rodet and Depalle, 1992a).

Using the FFT^{-1} synthesis, quasiperiodic signals can be easily composed. The parameters, namely, the frequency and the amplitude, are intuitive, although it is useful to apply higher level controls in order to efficiently create complex sounds with many partials. It is straightforward to add additional noise of arbitrary shape in the frequency domain representation. This is done by adding STFTs of desired noise in frequency domain representation of the signal under construction (Rodet and Depalle, 1992a).

There are several methods to improve the problems which arise mainly from the interpolation between consecutive frames. These are discussed in (Goodwin and Rodet, 1994) and (Goodwin and Gogol, 1995).

3.8 Formant Synthesis

In many cases it is useful to inspect spectral envelopes, i.e., a more general view of the spectra instead of the fine details provided by the Fourier transform. A central concept of spectral envelopes is a *formant*, which corresponds to a peak in the envelope of the magnitude spectrum. A formant is thus a concentration of energy in the spectrum. It is defined by its center frequency, bandwidth, amplitude, and envelope. Formants are useful for describing many musical instrument sounds but they have been used extensively for synthesis of speech and singing. See (Roads, 1995) for more details and references on the use of formants in sound synthesis.

In this section two sound synthesis methods based on formants are discussed. Formant Wave-Function synthesis is used in the CHANT program to produce high quality synthetic singing. VOSIM is a method for creating synthetic sound by trains of pulses of a simple waveform. These methods can also be interpreted as granular synthesis methods as both of them use short grains of sounds to produce the output signal. See Section 2.6 for a discussion and references of granular synthesis methods.

3.8.1 Formant Wave-Function Synthesis and CHANT

The formant wave-function synthesis has been developed at IRCAM, Paris, France (Rodet, 1980). The method starts from the premise that the production mechanism of many of the real-world sound signals can be presented as an excitation function and a filter (Rodet, 1980). The method assumes that the filter is linear and the excitation signal is composed of pulses of impulses or arches. The fundamental frequency of the tone is then readily determined as the period of the train of excitation pulses. In general, the response of the filter can be interpreted as a sum of responses of a set of parallel filters each of which corresponds to a formant in the synthesized waveform. The impulse responses of the parallel filters can be determined by analyzing one period of a recorded signal by linear prediction (Rodet, 1980).

The main elements of the formant wave-function synthesis are the *formant wave-functions* (French: fonction d'onde formantique, FOF) described by Rodet (1980). Each FOF corresponds to a formant or a main mode of the synthesized signal and it is obtained by analyzing a recorded signal as explained above. FOFs are computed in the time domain. A typical FOF ($s(k)$) is pictured in Figure 3.16 and it can be written as

$$\begin{aligned} s(k) &= 0 \quad \text{for} \quad k < 0 \\ s(k) &= \frac{1}{2}[1 - cos(\beta k)]e^{\alpha k} \sin(\omega k + \Phi) \quad \text{for} \quad 0 \leq k \leq \pi/\beta \\ s(k) &= e^{\alpha k} \sin(\omega k + \Phi) \quad \text{for} \quad k > \pi/\beta, \end{aligned} \tag{3.18}$$

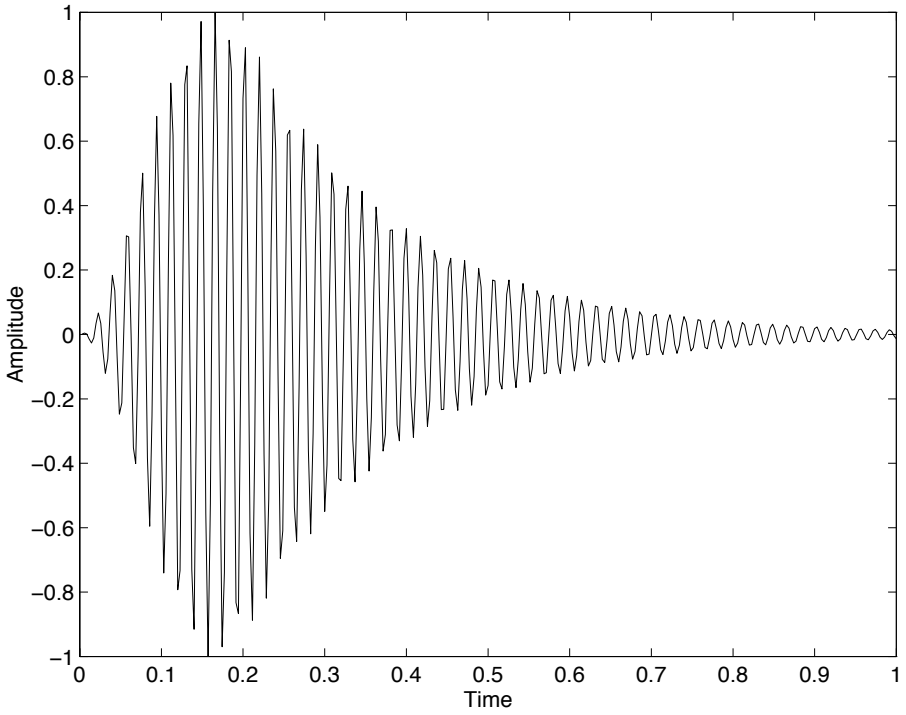

Figure 3.16: A typical FOF.

where ω is the center frequency, $\alpha\pi$ is the 3 dB bandwidth, parameter β governs the skirt width, and Φ is the initial phase. Naturally, the amplitude of the FOF can also be modified.

A FOF synthesizer is constructed by connecting FOF generators in parallel. The synthesizer can be controlled via instructions from the CHANT program. The user can utilize high-level commands and achieve comprehensive control without having to adjust the low-level parameters directly.

The CHANT program was originally written to produce high-quality singing voices, but it can also be employed to synthesize musical instruments (Rodet et al., 1984). It employs semiautomatic analysis of the spectrum of recorded sounds, extraction of gross formant characteristics, and fundamental-frequency estimation (Rodet, 1980). The program is discussed in detail in (Rodet et al., 1984). Sound examples of synthesized singing can be found in (Bennett and Rodet, 1989).

3.8.2 VOSIM

VOSIM (VOice SIMulation) is developed by Kaegi and Tempelaars (1978). It starts from the idea of presenting a sound signal as a set of tone bursts that have a variable duration and delay.

The pulses used in VOSIM are of fixed waveform. The VOSIM time function consists of N pulses that are shaped like squared sinusoids. The pulses are of equal duration T with decreasing amplitude (starting from value A) and followed by a delay M. Each pulse is obtained from the previous pulse by multiplying with a constant factor b. Such a time function is pictured in Figure 3.17. The five

parameters presented above are the primary parameters of VOSIM. For vibrato, frequency modulation, and noise sounds the delay M is modulated. Three more parameters are required: S is the choice of random or sine wave, D is the maximum deviation of M, and NM is the modulation rate. Four additional variables allow for transitional sounds: NP is the number of transition periods, and DT, DM, and DA the positive or negative increments of T, M, and A, respectively.

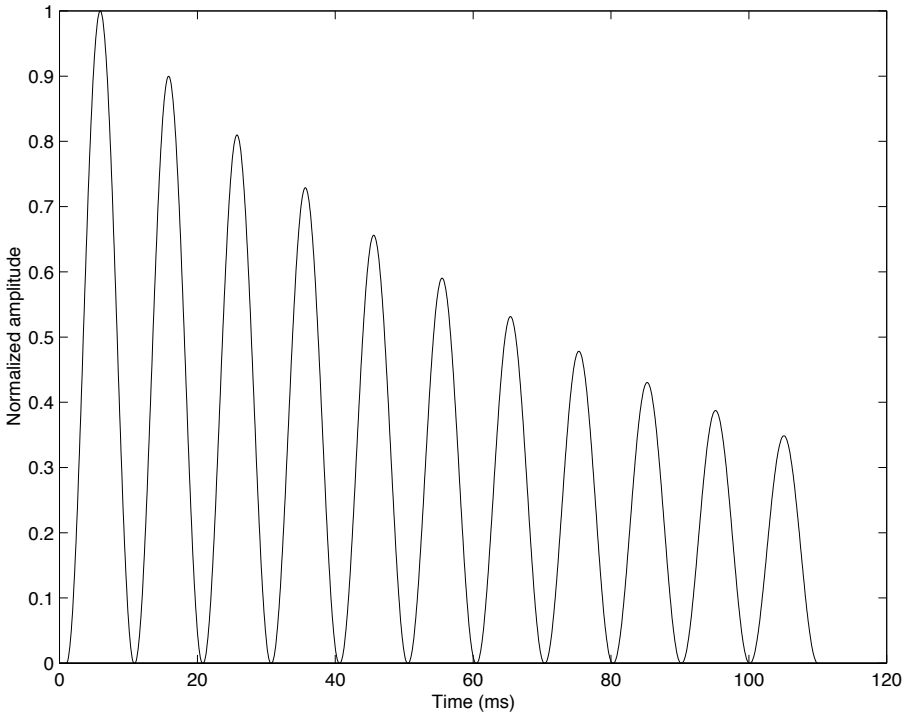

Figure 3.17: The VOSIM time function. $N = 11$, $b = 0.9$, $A = 1$, $M = 0$, and $T = 10$ ms.

4. Physical Models

Physical modeling of musical instruments has evolved to one of the most active fields in sound synthesis, musical acoustics, and computer music research. Physical modeling applications gain popularity by giving users better tools for controlling and producing both traditional and new synthesized sounds. The user is provided with a sense of a real instrument.

The aim of a model is to simulate the fundamental physical behavior of an actual instrument. This is done by employing the knowledge of the physical laws that govern the motions and interactions within the system under study, and expressing them as mathematical formulae and equations. These mathematical relationships provide the tools for physical modeling.

There are two main motivations for developing physics-based models. The first is that of science, i.e., models are used to gain understanding of physical phenomena. The other is production of synthesized sound. From the days of first physics-based models researchers and engineers have utilized them for sound synthesis purposes (Hiller and Ruiz, 1971a).

Physical modeling methods can be divided into five categories (Välimäki and Takala, 1996).

1. Numerical solving of partial differential equations
2. Source-filter modeling
3. Vibrating mass-spring networks
4. Modal synthesis
5. Waveguide synthesis

Waveguide synthesis is one of the most widely used physics-based sound synthesis methods in use today. It is very efficient in simulating wave propagation in one-dimensional homogeneous vibratory systems. The method is very much digital signal processing oriented and a number of real-time implementations using waveguide synthesis exists. Waveguide synthesis and single delay loop (SDL) models are discussed further in Chapter 5.

Source-filter models have been used especially for modeling the human sound production mechanism. The interaction of the vocal chords and the vocal tract is modeled as a feedforward system. Effective digital filtering techniques for source-filter modeling have been developed especially for speech transmission purposes. The technique is basically a physical modeling interpretation of the source-filter synthesis presented in Section 3.3.

The modeling methods simulate the system either in the time or the frequency domain. The frequency domain methods are very effective for models of linear systems. Musical instruments cannot in general be approximated accurately as being linear. Nonlinear systems make the frequency-domain approach infeasible. All the methods presented here model the system under study in the time domain.

This chapter starts with describing three physical modeling methods that use numerical acoustics. First, models using finite difference methods are represented. Applications to string instruments as well as to mallet percussion instruments are presented. Second, modal synthesis is discussed. Third, CORDIS is a system of modeling vibrating objects by mass-spring networks.

The interested reader is also referred to an interesting web site by De Poli and Rocchesso: `http://www.dei.unipd.it/english/csc/papers/dproc/dproc.html`

4.1 Numerical Solving of the Wave Equation

In this section, modeling methods based on finite difference equations will be discussed. These methods have been used especially for string instruments.

The method is in general applicable to any vibrating object, i.e., a string, a bar, a membrane, a sphere, etc. (Hiller and Ruiz, 1971a). The basic principle is to obtain mathematical equations that describe the vibratory motion in the object under study. These wave equations are then solved in a finite set of points in the object, thus obtaining a difference equation. The use of difference equations leads to a recurrence equation that can be interpreted as a simulation of the wave propagation in the vibrating object. The finite difference method is computationally most efficient with one-dimensional vibrators as the computational demands rapidly increase with introduction of more dimensions. The number of points in space increases proportional to the power of the number of dimensions. Furthermore, the number of computational operations for each point is increased, and the effective sampling frequency is increased. Digital waveguide meshes presented in Section 5.2 are DSP formulations of difference equations in two and three dimensions.

Hiller and Ruiz (1971a) were the first to take the approach of solving the differential equations of a vibrating string for sound synthesis purposes. They developed models of plucked, struck, and bowed strings. The stiffness of the string was modeled, as well as the frequency-dependent losses. Hiller and Ruiz (1971b) were able to produce synthesized sound and plots of the resulting waveforms by a computer program. Since that pioneer work developments have been made in modeling the

excitation, e.g., the interaction of the hammer and the piano strings, see (Chaigne and Askenfelt, 1994a) for references.

More recently Chaigne has been studying finite difference methods with applications to modeling of the guitar, the piano, and the violin (Chaigne et al., 1990), (Chaigne and Askenfelt, 1994a), (Chaigne and Askenfelt, 1994b). He has taken similar approaches to modeling a vibrating bar with application to the xylophone (Chaigne and Doutaut, 1997).

In this section a difference equation with initial and boundary conditions for a damped stiff string is first introduced. Then similar treatment is given for a vibrating bar. Finally, the synthesized waveforms are compared with original recordings of real instrument sounds.

4.1.1 Damped Stiff String

The model of a vibrating string presented here includes modeling the stiffness of the string as well as the frequency-dependent losses due to the friction with air, viscosity, and finite mass of the string. It describes the transversal wave motion of the string in a plane. The wave equation for the model is (Chaigne and Askenfelt, 1994a)

$$\frac{\partial^2 y}{\partial t^2} = c^2 \frac{\partial^2 y}{\partial x^2} - \varepsilon c^2 L^2 \frac{\partial^4 y}{\partial x^4} - 2b_1 \frac{\partial y}{\partial t} + 2b_3 \frac{\partial^3 y}{\partial t^3} + f(x, x_0, t), \tag{4.1}$$

where y is the displacement of the string, x the axis along the string, c the transverse wave velocity of the string, ε the stiffness parameter, L the string length, b_1 and b_3 the loss parameters, and $f(x, x_0, t)$ the excitation acceleration applied at point x_0. The excitation term is actually a force density term normalized by the mass density of the string so that the term will give acceleration of the string at point x.

The stiffness parameter ε is given by

$$\varepsilon = \kappa^2 \frac{ES}{TL^2}, \tag{4.2}$$

where κ is the radius of gyration of the string, E the Young's modulus, S the area of the string cross section, and T the string tension.

In Equation 4.1, the two partial time derivative terms of odd order model the frequency-dependent losses, i.e., the decay of the vibration. The decay is an effect of several physical phenomena. The effect of each phenomenon can be hard to separate from the total decay, and it will not be attempted here. However, some qualitative interpretations can be made. In the low-frequency range, the main causes for losses are the air resistance and the resistive impedances at the ends of the string (Chaigne, 1992). In the high-frequency range, the damping is mainly created by internal losses in the string, such as the viscoelastic losses in nylon strings discussed by Chaigne (1991). The parameters for the losses, b_1 and b_3, are obtained via the analysis of real instrument tones. The model does not try to model the individual physical processes that cause the dissipation of energy separately. The frequency-dependent

decay rate is given by

$$\sigma = \frac{1}{\tau} = b_1 + b_3\omega^2. \tag{4.3}$$

The string is excited by the force density term $f(x, x_0, t)$. It is assumed that the force density does not propagate along the string, thus time and space dependence can be separated in order to get

$$f(x, x_0, t) = f_H(t)g(x, x_0). \tag{4.4}$$

The term $g(x, x_0)$ can be understood as a spatial window which distributes the excitation energy to the string. This window smoothes the applied excitation, e.g., hammer strike on a piano string, so that artifacts that occur in the solution because of discontinuities will be eliminated.

The force density term $f_H(t)$ is related to the force $F_H(t)$ exerted in the excitation by

$$f_H(t) = \frac{F_H(t)}{\mu \int_{x_0-\delta x}^{x_0+\delta x} g(x, x_0)dx}, \tag{4.5}$$

where the effective length of the string section interacting with the exciter is $2\delta x$, and μ is the linear mass density of the string.

4.1.2 Difference Equation for the Damped Stiff String

The difference equation for the stiff damped string is obtained by discretizing the time and space by taking (Chaigne and Askenfelt, 1994a)

$$x_k = k\Delta x, \quad k \in [0, \frac{L}{\Delta x}] \tag{4.6}$$

and

$$t_n = n\Delta t, \quad n = 0, 1, 2, \ldots \tag{4.7}$$

The Δt and Δx are related by

$$c\frac{\Delta t}{\Delta x} = r \leq 1.$$

The condition $r = 1$ gives the exact solution with no numerical dispersion (Chaigne, 1992). However, r equals unity only in the case of an ideal string. For values $r < 1$ numerical dispersion will be present in the model. This will not be discussed further in this context, see (Chaigne, 1992) for more details. The main variable of interest will be the discretized transversal string displacement denoted $y(k, n) = y(k\Delta x, n\Delta t)$ for convenience. The derivation of the difference equation approximating Equation 4.1 is given by Hiller and Ruiz (1971a) and will not be

repeated here. However, it should be noted that for the sake of efficiency in computation one further simplification is made. The third order time derivative term in Eq. 4.1 would yield the following approximation with time $t_n = n$ as central point:

$$\frac{\partial^3 y}{\partial t^3} \approx y(k, n+2) - 2y(k, n+1) + 2y(k, n-1) - y(k, n-2), \tag{4.8}$$

i.e., the need for implicit methods. This can be overcome by noticing that the magnitude of the term $2b_3 \frac{\partial^3 y}{\partial t^3}$ is relatively small, and by reducing the number of time steps by employing the recurrence equation for the ideal string:

$$y(k, n+1) = y(k+1, n) + y(k-1, n) - y(k, n-1). \tag{4.9}$$

Using this equation to simplify Eq. 4.8 will not increase the number of time or space steps involved in the recurrence equation. The general recurrence equation is now given by

$$\begin{aligned} y(k, n+1) \;=\; & a_1 y(k, n) + a_2 y(k, n-1) \\ & + a_3[y(k+1, n) + y(k-1, n)] \\ & + a_4[y(k+2, n) + y(k-2, n)] \\ & + a_5[y(k+1, n-1) + y(k-1, n-1) + y(k, n-2)] \\ & + [\Delta t^2 N F_H(n) g(k, i_0)]/M_S, \end{aligned} \tag{4.10}$$

where the coefficients a_1 to a_5 are given with Equations 4.11.

$$\begin{aligned} a_1 &= (2 - 2r^2 + b_3/\Delta t - 6\varepsilon N^2 r^2)/D, \\ a_2 &= (-1 + b_1 \Delta t + 2b_3/\Delta t)/D \\ a_3 &= r^2(1 + 4\varepsilon N^2)/D, \\ a_4 &= (b_3/\Delta t - \varepsilon N^2 r^2)/D \\ a_5 &= (-b_3/\Delta t)/D, \\ \text{where} & \\ D &= 1 + b_1 \Delta t + 2b_3/\Delta t \quad \text{and} \quad r = c\Delta t/\Delta x \end{aligned} \tag{4.11}$$

Figure 4.1 shows the how displacement $y(k, n+1)$ depends on previous values of displacement when using Eq. 4.10. This equation can be directly utilized to compute the displacements of the chosen discrete points on the string.

4.1.3 The Initial Conditions for the Plucked and Struck String

The initial conditions are given for models of the guitar and the piano. The conditions are very dissimilar and correspond to the excitation by either plucking or striking the string.

Plucked String

Excitation by plucking is the simplest case and its initial conditions are given directly by Equation 4.5. The spatial window $g(x, x_0)$ and the string section affected are

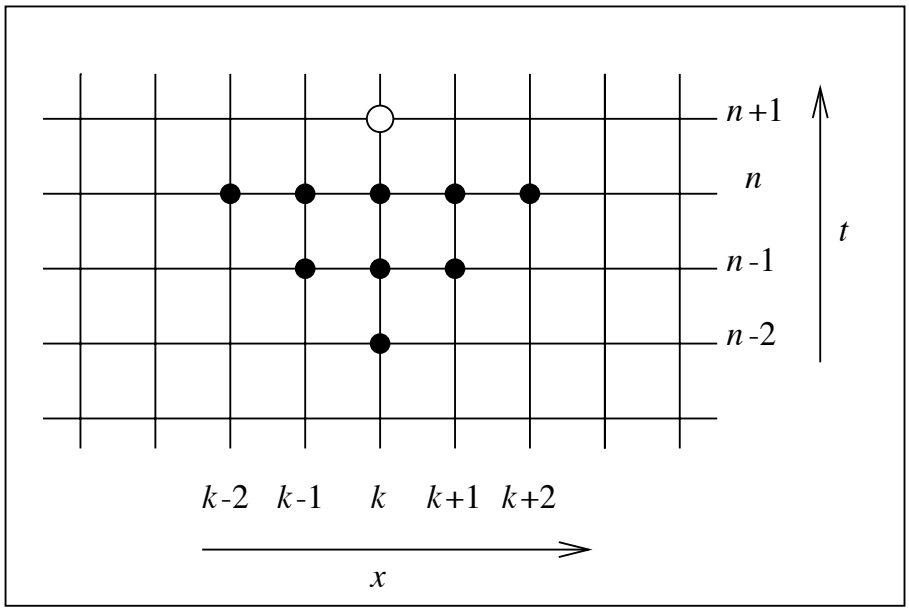

Figure 4.1: Dependence of displacement $y(k, n+1)$ on previous values of the displacement, after (Chaigne, 1992). The next value marked with ○ will depend on those points in time and space marked with •.

determined mainly by the type of the pluck. The velocity of the pluck determines the time distribution of the excitation. Naturally, these are not mutually independent. It may be helpful to consider the plucking event as being mapped to a force density distribution that can then be separated to parts depending on space and time. A more detailed model of the plucking event including modeling the finger-string interaction is given by Chaigne (1992).

For the guitar, the initial condition is introduced by rewriting the last term on the right hand side of Eq. 4.10

$$\Delta t^2 \frac{N}{m_S} F(n) g(k, i_0), \tag{4.12}$$

where N is the number of points on the string, m_S is the mass of the string, $F(n)$ is the force applied by finger or plectrum, and $g(k, i_0)$ is the discretized spatial window (Chaigne et al., 1990).

Struck String

For the development of an expression of the initial conditions for the piano an assumption of zero initial velocity and displacement of the string is made by Chaigne and Askenfelt (1994a). This assumption is made only for the sake of simplicity in discussing the initial condition, the model has no restrictions on the initial condition. With the string at rest at $t = 0$ we have

$$y(k, 0) = 0.$$

One further assumption is needed for Equation 4.10 to be applicable to the first three time steps. The calculation involves the states of the string at three past time

steps. Thus $y(k,1)$ is estimated by using approximated Taylor series to obtain

$$y(k,1) = \frac{y(k+1,0) + y(k-1,0)}{2}.$$

Now for the displacement of the hammer at time $n = 1$ we calculate

$$\eta(1) = V_{H0}\Delta t,$$

where V_{H0} is the hammer velocity at $t = 0$, and for the force exerted by the hammer

$$F_H(1) = K|\eta(1) - y(k-1,0)|^p. \tag{4.13}$$

Note that the force term at $t = 1$ is computed using the initial velocity, i.e., a unit delay is introduced in order for the force to be computable. Borin et al. (1997a) propose a more elaborate method for eliminating delay-free loops in discrete-time models. Interestingly, they also apply the method for modeling the hammer-string interaction.

Continuing with the treatment of Chaigne and Askenfelt (1994a), the displacement $y(k,2)$ is computed using a simplified version of Eq. 4.10

$$y(k,2) = y(k-1,1) + y(k+1,1) - y(k,0) + \frac{\Delta t^2 N F_H(1)}{M_H}. \tag{4.14}$$

Here the stiffness and damping terms are neglected in order to limit the space and time dependence, i.e., no terms with $n = 2$ are included. For the hammer, displacement $\eta(2)$ and force $F_H(2)$ are computed by

$$\begin{aligned} \eta(2) = &\quad 2\eta(1) - \eta(0) - \tfrac{\Delta t^2 F_H(1)}{M_H} \\ F_H(2) = &\quad K|\eta(2) - y(k_0,2)|^p. \end{aligned} \tag{4.15}$$

The effect of the simplifications is discussed by Chaigne and Askenfelt (1994a).

After the displacements $y(k,n)$ are known to first three time samples, it is possible to start using the recurrence formula of Eq. 4.10 directly. The force $F_H(n)$ is assumed to be known, and its effect for the string is taken into account until time n when

$$\eta(n+1) < y(k_0, n+1).$$

After this the string is left free for vibrations unless recontact of the hammer is modeled. The force density term $f(x, x_o, t)$ can be applied at the string at any time.

4.1.4 Boundary Conditions for Strings in Musical Instruments

Terminations of strings in musical instruments are not completely rigid. For example, in the guitar the bridge has a finite impedance, and the finger terminating the string against the fingerboard is far from rigid. The boundary conditions are given for the guitar, the piano, and the case of the violin is discussed briefly.

The boundary conditions for plucked, bowed, and struck string instruments, like the guitar, the violin, and the piano, can be described by one of the three models

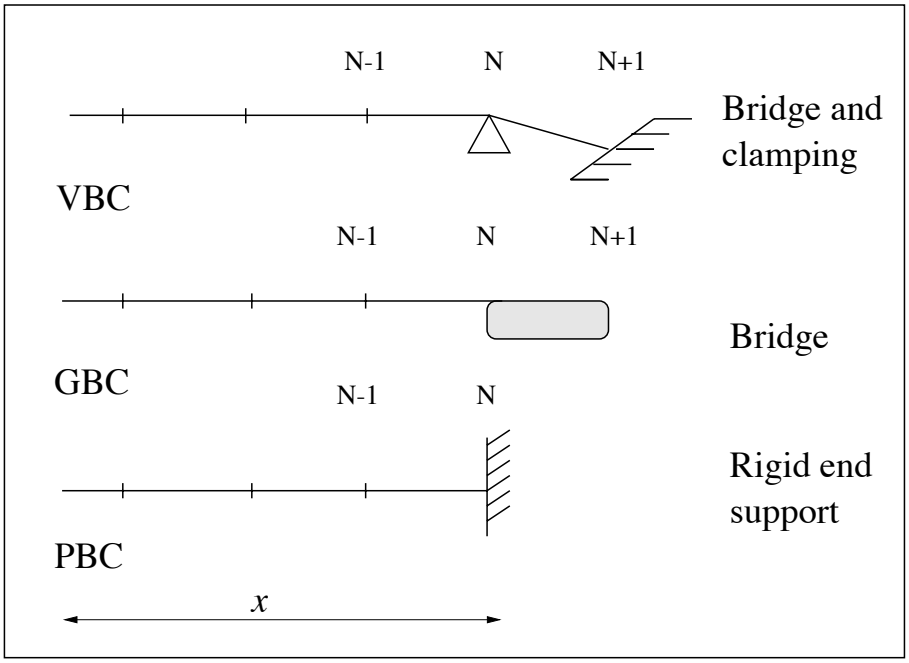

Figure 4.2: Models for boundary conditions of string instruments, after (Chaigne, 1992). VBC: Violin-like boundary condition. GBC: Guitar-like boundary condition. PBC: Piano-like boundary condition.

presented in Figure 4.2 (Chaigne, 1992). Here point N on the string corresponds to the point of the bridge. For the violin-like boundary conditions it is assumed that the displacement $y(N, n)$ of the string is non-zero. Furthermore, the displacement of the string at the point $x = N + 1$ is taken to be much smaller than at $x = N$. If the distance between the bridge and the clamping position is greater than the space-step, e.g., $p\Delta x$, the boundary condition can be written as $y(N + p, n) = 0$, instead of $y(N + 1, n) = 0$. Then the expressions for the intermediate points would have to be developed.

In the guitar-like boundary condition the string is clamped just behind the bridge, so that the distance between the bridge and the clamping position is usually above the audible wavelength range of a human. Thus, it can be assumed that

$$y(N, n) = y(N + 1, n), \tag{4.16}$$

i.e., $y(N, n)$ denotes the displacement of the string as well as the resonating box at the bridge. This allows for modeling of the coupling of the bridge and the resonating body by using measured values of the input admittance at the guitar bridge. Modeling the resonances and the radiated sound pressure is discussed by Chaigne (1992).

The piano string is assumed to be hinged at both ends yielding the following boundary conditions (Fletcher and Rossing, 1991):

$$\begin{aligned} y(0, t) = y(L, t) = 0 \\ \frac{\partial^2 y}{\partial x^2}(0, t) = \frac{\partial^2 y}{\partial x^2}(L, t) = 0 \end{aligned} \tag{4.17}$$

For the model of the piano the boundary conditions are obtained by discretizing Eqs. 4.17 and they can be expressed as:

$$y(0,n) = y(N,n) = 0 \tag{4.18}$$

$$y(-1,n) = -y(1,n) \quad \text{and} \quad y(N+1,n) = -y(N-1,n) \tag{4.19}$$

The string is coupled to the soundboard at point N. If the frequency-dependent properties of the coupling effect are desired, the second condition in Eq. 4.19 can be replaced with a difference equation approximating the differential equation governing the coupling. Equation 4.19 is important for deriving expressions for string motion at points $y = -1$ and $y = N + 1$, for these points are not explicitly included in the model. These points are needed for the calculation of the displacement of the string at points $y = 1$ and $y = N - 1$ because the differential equation for the string is of fourth order, i.e., the recurrence equation for point k depends on points $k - 2$ and $k + 2$.

4.1.5 Vibrating Bars

An approach similar to the case of string is taken by Chaigne and Doutaut (1997) for the vibrating bar. Theoretical treatment of vibrating bars is given by, e.g., Morse and Ingard (1968), and mallet percussion instruments are discussed by Fletcher and Rossing (1991).

It is assumed that the vertical component $w(x,t)$ of the displacement of a xylophone bar is given by the two following equations:

$$M(x,t) = EI(x)(1 + \eta\frac{\partial}{\partial t})\frac{\partial^2 w(x,t)}{\partial x^2}, \tag{4.20}$$

and

$$\frac{\partial^2 w(x,t)}{\partial t^2} = \frac{1}{\rho S(x)}\frac{\partial^2 M(x,t)}{\partial x^2} - \gamma_B\frac{\partial w(x,t)}{\partial t} - \frac{\chi}{M_B}w(x,t) + f(x,x_0,t). \tag{4.21}$$

$M(x,t)$ is the bending moment and $I(x)$ the moment about the x axis. $S(x,t)$ is the cross sectional area of the bar. E is the Young's modulus and ρ the density of the vibrating bar. The coefficients η and γ_B account for losses. They are obtained by analyzing the decay times of partials on real instruments. Estimation of the stiffness coefficient χ is obtained by measuring the natural frequency of a spring-mass system composed of the bar with mass M_B, and the supporting cord.

The model for the interaction between the bar and the mallet is similar to the one used for the hammer-string interaction in the piano model with force

$$f_H(t) = \frac{F_M(t)}{\rho S(x)\int_{x_0-\delta x}^{x_0+\delta x} g(x,x_0)dx}, \tag{4.22}$$

where $S(x_0)$ is the cross section of the bar at point x_0, and ρ is the density of the bar. The spatial smoothing of the impact is obtained by employing a spatial window as in Eq. 4.4.

The impact force is given by Eq. 4.13 with $p = 3/2$. The non-integer exponent $3/2$ is now derived from the general theory of elasticity, as opposed to the case of the piano where analysis of experimental data must be used, see (Chaigne and Doutaut, 1997, Appendix A) for derivation. The stiffness coefficient K is obtained by analysis of experimental data.

This interaction model is able to simulate three important physical aspects of the instrument:

1. The introduction of kinetic energy, localized in time and space, into the vibrating system.
2. The influence of the initial velocity on both the contact duration and the impact force due to the nonlinear force-deformation law. This determines the spectrum of the tone.
3. The influence of the stiffness of the two materials in contact, which strongly determines the tone quality of the initial blow, i.e., the attack.

These principles apply to the model of the piano as well.

For the numerical formulation of the xylophone model, the same principles as in the case of the string are employed. However, the explicit computation scheme already used in the guitar and piano models is applicable only to a simple case of a uniform bar with constant cross-sectional area. This is the only model discussed in this context. Chaigne and Doutaut (1997) discuss also the more demanding model of a bar with a variable section.

The differential equation for the uniform bar is

$$\frac{\partial^2 w(x,t)}{\partial t^2} = -a^2[\frac{\partial^4 w(x,t)}{\partial x^4} + \eta\frac{\partial^5 w(x,t)}{\partial t \partial x^4}] - \gamma_B \frac{\partial w(x,t)}{\partial t} - \frac{\chi}{M_B} w(x,t) + f(x, x_0, t), \tag{4.23}$$

where $a^2 = EI/\rho S$.

The recurrence equation approximating Eq. 4.23 is given by (Chaigne and Doutaut, 1997)

$$\begin{aligned} y(k,n+1) \;=\; & c_1 w(k,n) + c_2 w(k,n-1) \\ & + c_3[w(k+2,n) - 4w(k+1,n) - 4w(k-1,n) + w(k-2,n)] \\ & + c_4[w(k+2,n-1) - 4w(k+1,n-1) \\ & - 4w(k+1,n-1) + w(k-2,n-1)] \\ & + c_5 F_M(n) g(k, i_0)], \end{aligned} \tag{4.24}$$

where

$$c_1 = \frac{2 - 6r^2(1+\beta) - (\Delta t \omega_B)^2}{1+\gamma}, \quad c_2 = \frac{-1 + \gamma + 6\beta r^2}{1+\gamma}$$
$$c_3 = \frac{-r^2(1+\beta)}{1+\gamma}, \quad c_4 = \frac{\beta r^2}{1+\gamma}, \quad c_5 = \frac{N}{M_B f_s^2 (1+\gamma)}, \tag{4.25}$$

where

$$\beta = \eta f_s, \quad \omega_B^2 = \frac{\chi}{M_B}, \quad \gamma = \frac{\gamma_B}{2 f_s} \quad \text{and} \quad r = a\frac{N^2}{f_s L^2} \tag{4.26}$$

It can be shown that the explicit scheme remains stable if the number of spatial points N is (Chaigne and Doutaut, 1997, Appendix B)

$$N \leq M_{\text{MAX}} = \frac{3}{4}\sqrt{\frac{\pi f_s}{f_1(1+\eta f_s)}},$$

where f_1 is the frequency of the lowest partial. For wooden bars, the order of magnitude for the term $\eta f_s = 10^{-2}$. It can be seen that according to this stability criterion the maximum number of spatial points is roughly proportional to the square root of the sampling frequency. Thus, to double the spatial resolution, sampling frequency of four times the original is required. Furthermore, it can be observed that there is an asymptotic limit $\frac{3}{4}\sqrt{\pi/f_1\eta}$ for N_{MAX} as f_s increases. The maximum spatial resolution obtained by using the sampling frequency of 192 kHz is equal to 1 cm.

A comparison of the original measured signals and those obtained with the model of variable cross-sectional area is discussed in the next section.

4.1.6 Results: Comparison with Real Instrument Sounds

The models described in the previous sections have been evaluated and compared to real instruments by Chaigne and Askenfelt (1994b), Chaigne et al. (1990), and Chaigne and Doutaut (1997). This is important not only for the validation of the models, but also for studying the contribution of each individual physical parameter to the signal. Typically the effect of a single parameter on produced sound can be hard to establish by observing the instrument or the produced sound.

Only a short qualitative comparison between measured and simulated signals is given in this section. References to detailed presentations of each instrument are given in the corresponding subsection.

The Piano

Chaigne and Askenfelt (1994b) give a detailed and systematic discussion on the comparison of real signals to those obtained by simulation.

The string velocities were computed for bass (C2), midrange (C4), and treble (C7) tones. The overall result is that the model is capable of producing the waveforms quite well over the whole register of the piano, including the attack transients. Some small discrepancies in the bass range can be caused by the non-rigid termination of a real piano string. The model does not attempt to take this phenomenon into account.

The spectra of the string velocities with notes played at different ranges with different dynamics show a good behavior of the model. The spectra show increased spectral content with increased hammer velocity, as expected. Large and audible differences were observed above the first 5-7 partials, although these discrepancies had little effect to the waveforms.

The Guitar

The guitar and the corresponding finite difference model are compared by Chaigne et al. (1990). The waveforms of a vibrating guitar string were obtained by a simple electrodynamic method. A concentrated magnetic field was applied perpendicular to the vibrating string. The generated voltage proportional to the string velocity at the point of the magnetic field was measured between the string ends.

It was observed that the measured and simulated waveforms were similar. Furthermore, the influence of the body response was more clearly visible in the measured signal.

The Xylophone

The measurements and comparison were conducted by Chaigne and Doutaut (1997). In this case the acceleration of the mallet's head was measured. The corresponding force signal was derived by multiplication by the equivalent mass of the mallet. The acceleration of the chosen point on the bar was either measured with the help of an accelometer or derived from the velocity signal obtained by a laser vibrometer.

Two different types of mallets were simulated: a soft mallet with rubber head, and a hard mallet with boxwood head. For both mallets signals of weak (*piano*) and strong (*mezzo-forte*) impact were measured and simulated. Three comparisons were made: bar accelerations, impact forces, and bar acceleration spectra.

For a weak impact with a soft mallet the general shape and amplitude of the waveform of the bar accelerations were similar. However, the upper partials seemed to be damped more rapidly in the measured acceleration. For a strong impact both the magnitude and the shape of the signals were very similar. The model seems to work better with hard mallets because the bar acceleration waveforms show a good match with both the weak and the strong impact.

The magnitude of the impact forces on the bar showed that the order of magnitude for both shapes and amplitudes are reproduced fairly well for the soft mallet.

The impact durations were systematically shorter by approximately 20 %. With hard mallets the impact durations were identical, as well as the shapes and magnitudes of the force signals.

The frequency-domain comparison of the bar acceleration signal showed again better match with the hard mallet. The first three partials were almost identical with a discrepancy of less than or equal to 2 dB. With the soft mallet the third partial is approximately 15 dB below the corresponding partial of the measured signal.

For a detailed comparison and discussion on the cause of the discrepancies, see (Chaigne and Doutaut, 1997).

4.2 Modal Synthesis

The modal synthesis method has been developed mainly at IRCAM in Paris, France (Adrien, 1989), (Adrien, 1991). They have produced a commercial software application Modalys (Eckel et al., 1995), formerly called Mosaïc (Morrison and Adrien, 1993). With this application the user can simulate vibrating structures. The user describes the structure under study for the program, and the program computes the modal data and outputs the signal observed at a point defined by the user.

Modal synthesis is based on the premise that any sound-producing object can be represented as a set of vibrating substructures which are defined by modal data (Adrien, 1991). Substructures are coupled and they can respond to external excitations. These coupling connections also provide for the energy flow between substructures. Typical substructures are:

- bodies and bridges
- bows
- acoustic tubes
- membranes and plates
- bells

The simulation algorithm uses the information of each substructure and their interactions.

The method is general as it can be applied to structures of arbitrary complexity. The computational effort needed increases rapidly with complexity thus setting the practical limits of the method. Next, the formulation of modal data of a substructure is presented. Then an application to real musical instrument is shortly discussed.

4.2.1 Modal Data of a Substructure

The modal data for a substructure consists of the frequencies and damping coefficients of the structure's resonant modes and of the shapes of each of the modes (Adrien, 1991). A vibrating mode is essentially a particular motion in which every point of the structure vibrates with same frequency. It should be noted that an arbitrary motion of a structure can be expressed as a sum of the contributions by the modes as can be done by Fourier series expansion.

The modes are excited by an external force applied at a given point on the structure. The excitation energy is distributed to the modes depending on the form of the excitation. It is assumed that there exists no exchange of energy between the modes. In practice, the vibration pattern is never fully described by a single mode, but it is a sum of an infinite series of vibrating modes. This accounts for an infinite number of degrees of freedom in a continuous structure. For numeric computation of vibration of the structure to become realizable, the continuous structure must be spatially divided into a finite set of points.

Given a set of N points on a structure a number of N modes can be represented. Each mode is described by its resonant frequency ν_m, and damping coefficients ξ_m. The $N \times N$ *modes' shape matrix* $[\Phi_{m,k}]$ describes the relative displacements of the N points in each mode. Column m of the modes' shape matrix corresponds to the contribution of mode m to the displacements of the N points. Each mode can then be presented as a second-order resonator connected in parallel with the others as pictured in Fig. 4.3.

The modal data can be obtained analytically for simple vibrating structures. The expressions for the modal data for each mode can be obtained from the differential equation system governing the motion of the simple vibrating system. For complex structures direct computation of modal data is not possible, and analysis based on measuring experiments must be utilized. Modal analysis is used extensively in aircraft and car industry, and thus the tools are efficient and available. They typically consist of excitation and pickup devices, signal processing hardware and software for Fourier transforms and polynomial extraction of modal data. Similar methods have been used for parameter calibration of other physical models (Välimäki et al., 1996).

The method is similar for mechanical and acoustical systems. In mechanical systems the deflections in the modes' shape matrix are the actual displacements of the points on the surface of the vibrating structure. In acoustical systems elements of the modes' shape matrix correspond to the deflections of sound pressure or particle velocity.

4.2.2 Synthesis using Modal Data

Modal synthesis is very powerful in that all vibrating structures can be described using the same equations. These equations describe the response of the structure to an excitation applied at a given point. For a mechanical structure partitioned to N

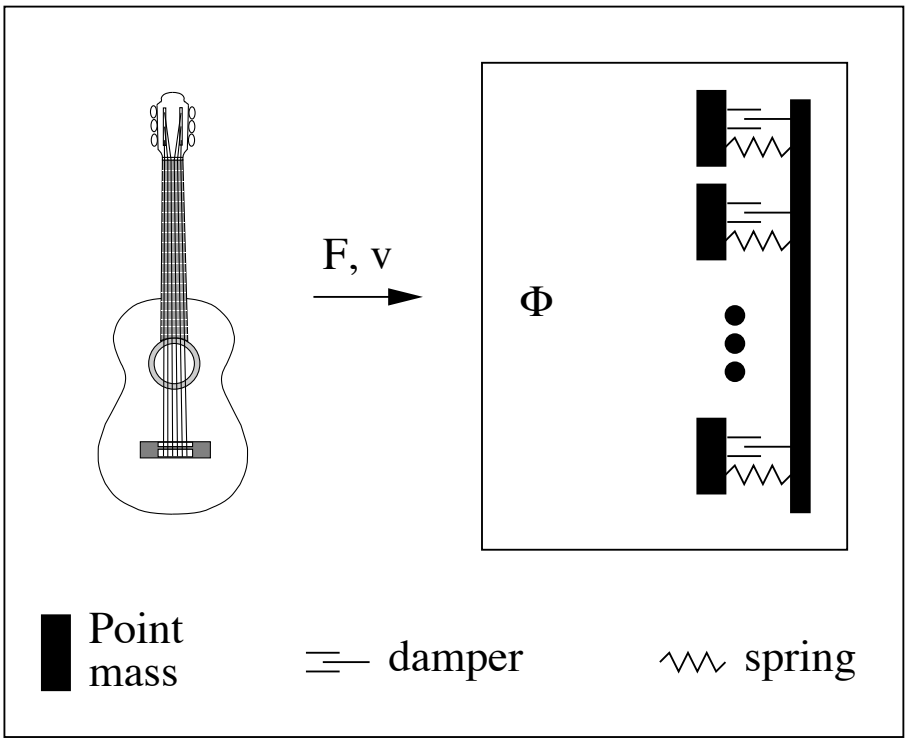

Figure 4.3: A modal scheme for the guitar. A complex vibrating structure is represented as a set of parallel second-order resonators responding to the external force F, and contributing to the resulting velocity v.

points the equation for the instantaneous velocity of the k^{th} point is (Adrien, 1991)

$$\frac{\partial y_{k,t+1}}{\partial t} = \sum_{m=1}^{N} \Phi_k^m \frac{\sum_{l=1}^{P} \Phi_l^i F_{l,t+1}^{\text{ext}} + \frac{\partial \varphi_{m,t}}{\partial t}\frac{1}{\Delta t} - \omega_m^2 \varphi_{m,t}}{\frac{1}{\Delta t} + 2\omega_m \xi_m + \omega_m^2 \Delta t}, \tag{4.27}$$

where Φ_k^m is the contribution of the m^{th} mode to the deflection of point k on the structure, $F_{l,t+1}^{\text{ext}}$ is the instantaneous external force on point l of the structure, Δt is the time step, and ω_m, ξ_m, and φ_m the angular frequency, the damping coefficient, and the instantaneous deflection associated with the m^{th} mode.

A similar equation can be applied to acoustic systems with the external forces $F_{l,t+1}^{\text{ext}}$ replaced by external flows $U_{l,t+1}^{\text{ext}}$. The density of air is denoted by ρ_0. The equation becomes

$$p_{k,t+1} = \rho_0 \frac{\partial y_{k,t+1}}{\partial t} = \sum_{m=1}^{N} \Phi_k^m \frac{\sum_{l=1}^{P} \Phi_l^i U_{l,t+1}^{\text{ext}} + \frac{\partial \varphi_{m,t}}{\partial t}\frac{1}{\Delta t} - \omega_m^2 \varphi_{m,t}}{\frac{1}{\Delta t} + 2\omega_m \xi_m + \omega_m^2 \Delta t}, \tag{4.28}$$

If all instantaneous external excitations are known, the velocities of the modes and thus the velocities of all of the points can be calculated. However, typically only the excitation corresponding to control and driving data are known, and other forces of flows have to be determined. These forces or flows implement the coupling, i.e., the energy flow, between substructures. The couplings are often nonlinear. The reed/air column interaction in woodwind instruments is an example of coupling of linear systems governed by Equations 4.27 and 4.28 respectively. The coupling

equation involves the flow entering the bore U_0^{ext}, and the pressure difference between the mouth and the bore $P_m - P_0$, the position of the reed ξ, the Backus constant B, and the additional flow $S_0\xi$ due to the displacement of the reed. The interaction shifts between two regions (Adrien, 1991)

$$\begin{aligned} \text{Open reed} \quad & U_{0,t+1}^{\text{ext}} = B\sqrt[3]{(p_{m,t+1} - p_{0,t+1})^2\xi^4} + S_0\xi_{t+1} \\ \text{Closed reed} \quad & \begin{aligned} U_{0,t+1}^{\text{ext}} &= 0 \\ \xi_{t+1} &= 0 \end{aligned} \end{aligned} \tag{4.29}$$

4.2.3 Application to an Acoustic System

When the modal synthesis method is applied to a simple acoustical system consisting of a conical tube with a simple reed mouthpiece and five holes, six equations of the form of Eq. 4.28 are utilized, one for the cone, and one for each hole. The reed is presented as a mechanical system with nonlinear coupling to the cone with Equations 4.28 and 4.29. The interactions between substructures involve flow conservation. Using this principle, it is possible to eliminate all pressure terms from the equations and present the equations in a 6×6 matrix form. For a description of the matrix equations see (Adrien, 1991).

The modal synthesis method provides many possible output signals. It is interesting, at least for research purposes, to try to recreate the acoustic field of a real instrument. This can be done by utilizing a body of a real instrument to radiate the created acoustic signal. In the case of the violin, the string, the bridge, and the exciter is modeled as usual, but the body is replaced by an infinite impedance in the model. The sound outputs obtained at the foot of the bridge are used as force signals to drive shakers at the foot of a real instrument bridge. The implicit assumption made above is that the body does not act as a load for the strings and therefore it does not affect the attenuation and phase of the partials in bow-string interaction (Rocchesso, 1998). Adrien (1991) gives a detailed discussion of the simulated signals, but lacks comparison to measured real instrument signals.

4.3 Mass-Spring Networks – the CORDIS System

Cadoz et al. (1983) attempt to model the acoustical system under study using simple ideal mechanical elements, such as masses, dampers and springs. They aim to develop a paradigm which can be applied to an arbitrary acoustic system. The CORDIS system was the first system capable of producing sound based on a physical model in real time (Florens and Cadoz, 1991). In this section, first the basic elements of the system are described. Second, application to modeling a plucked string is discussed.

4.3.1 Elements of the CORDIS System

The most primitive and fundamental elements of the system are the following:

1. point masses
2. ideal springs
3. ideal dampers

When these components are combined and connected in sufficient number, reproduction of spatial continuum and an acoustical signal should be possible at a desired sampling rate (Florens and Cadoz, 1991). The object under study is then approximated as a set of these elements discretely distributed over its surface. A major simplification is obtained by taking each element as being one-dimensional, i.e., each element can only move or act in one dimension. For modeling interactions that vary in time, e.g., bowing, striking by a hammer, or plucking, a conditional link is introduced. It consists of a spring and a damper with adjustable parameters connected in parallel.

The mathematical presentations for the elements are simple and they are given by Florens and Cadoz (1991) with

$$\text{Mass}: \quad F = m\frac{\partial^2 x}{\partial t^2} \tag{4.30}$$

$$\text{Spring}: \quad F_1 = F_2 = -K(x_1 - x_2) \tag{4.31}$$

$$\text{Damper}: \quad F_1 = F_2 = -Z(\frac{\partial x_1}{\partial t} - \frac{\partial x_2}{\partial t}) \tag{4.32}$$

$$\text{Conditional link}: \quad F_1 = F_2 = -K(x_1 - x_2) - Z(\frac{\partial x_1}{\partial t} - \frac{\partial x_2}{\partial t}), \tag{4.33}$$

where F is the force driving the mass, F_1 and F_2 are the forces at points x_1 and x_2 of the spring, damper, or the condition link, Z is the friction coefficient, and K the spring coefficient.

The same equations may be obtained in discretized form by taking

$$\frac{\partial x(n)}{\partial t} \longrightarrow x(n) - x(n-1)$$

and

$$\frac{\partial^2 x(n)}{\partial t^2} \longrightarrow \frac{\partial x(n)}{\partial t} - \frac{\partial x(n-1)}{\partial t},$$

thus

$$\text{Mass}: \quad F(n) = m[x(n) - 2x(n-1) + x(n-2)] \tag{4.34}$$

$$\text{Spring}: \quad F_1(n) = F_2(n) = -K[x_1(n) - x_2(n)] \tag{4.35}$$

$$\text{Damper}: \quad F_1(n) = F_2(n) = -Z[x_1(n) - x_1(n-1) - x_2(n) + x_2(n-1)] \tag{4.36}$$

$$\text{Conditional link}: \quad F_1(n) = F_2(n) = -K[x_1(n) - x_2(n)] - Z[x_1(n) - x_1(n-1) - x_2(n) + x_2(n-1)] \tag{4.37}$$

Temporal discretization introduces an error to the frequency of each of the modeled harmonic in the form

$$\Delta\omega \approx \frac{\omega^2 \Delta T^2}{24},$$

Since the sampling frequency f_s is the reciprocal of the time step ΔT, the error can be reduced by increasing the sampling frequency. Using a sampling frequency of three times the frequency of the highest harmonic component guarantees a maximum error of 5 percent on all partials.

The vibrating string is modeled with N masses connected with $N-1$ identical parallel springs and dampers as illustrated in Figure 4.4. This continuum of points on the string is connected at both ends to rigid end supports with a damper and a spring in parallel. In this case there will be N harmonics present in the signal.

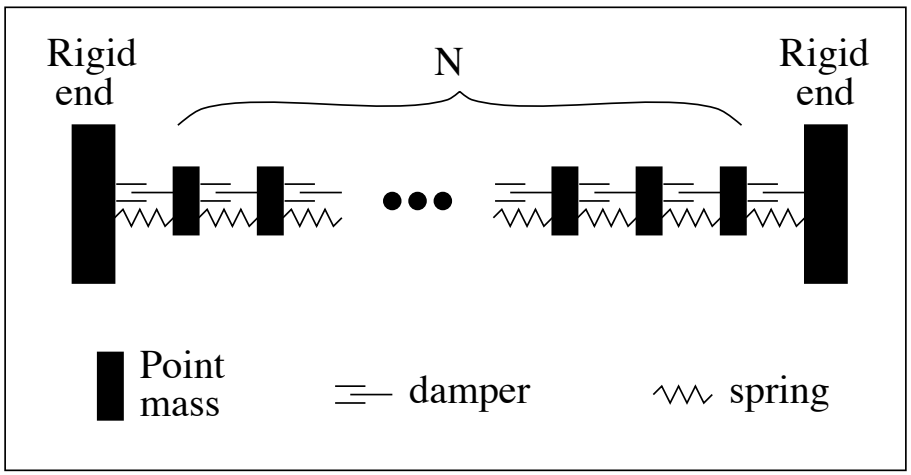

Figure 4.4: A model of a string according to the CORDIS system. N point masses are connected to each other and to rigid end supports with a damper and a spring in parallel at each connection.

The creators of CORDIS have developed a system called ANIMA for two and three dimensional elements (Florens and Cadoz, 1991).

4.4 Comparison of the Methods Using Numerical Acoustics

In this section the three presented methods are compared by discussing the application of each method to an acoustic system: the guitar.

Using finite difference equations for simulating the vibration of a string is presented in Section 4.1. The finite difference method is very accurate in reproducing the original waveform if the model parameters are correct. The approach is interesting especially in a scientific sense, because the vibratory motion can be observed at any discrete point on the string. Furthermore, the parameters of the model are the actual parameters of the real instrument, such as the stiffness and the loss parameters of the string, and the input admittance at the bridge. These parameters can be obtained via measurements and analysis on both the instrument and the signals produced by it.

For real-time sound synthesis purposes, the finite difference model is not very attractive. The model can only be applied to a simple structure, such as a vibrating string, in real-time; including a guitar body in the model would imply the need for hybrid systems. An estimation on the complexity of the computation can be obtained by inspecting Equation 4.10. For each of the N points on a string, five multiplications and eight summations are needed, with an additional multiplication and summation if an excitation is applied at that point. For good spatial resolution, N needs to be large. So several hundreds of operations are needed for every output sample. Also, numerical dispersion might be a problem with the FD method (Rocchesso, 1998). An example program for simulating string vibration as well as the effect of each individual parameter is written by Kurz and Feiten (1996). The program for Silicon Graphics workstations can be downloaded at `ftp://ftp.kgw.tu-berlin.de/pub/vstring/`.

With modal synthesis the guitar can be divided into three substructures, one for every functional part of the instrument, namely, the excitation, the vibrating strings, and the body radiating the sound field. The excitation substructure only interacts with other parts when the string is being plucked. The excitation can be applied at any points on other two substructures. The vibrating string is simulated with N parallel independent second-order resonators, each producing one harmonic component. The resonators can be computationally efficiently implemented, but a large number of them are needed for high-quality synthesis. The model for the body of the instrument is obtained by modal analysis of the structure. This is a very time-consuming process, especially for a complex structure, such as the violin (Adrien, 1991).

If a body of a real instrument is used as a transducer, the radiated sound field, produced by the vibrating string coupled to the body, can be simulated. Naturally, this method can also be used with other methods capable of producing a driving force signal at the bridge.

The CORDIS system divides each vibrating structure into idealized elements, i.e., point masses vibrating in one direction connected with ideal dampers and springs. A vibrating string is thus simulated with N point masses connected together with $N-1$ links composed of a damper and a string connected in parallel. This structure is capable of producing N harmonics. The number of computational operations for each cell is relatively low. An estimation can be made by analyzing Equations 4.34 - 4.36. One output sample requires approximately $3N$ multiplications, and $6N$ summations. Unfortunately, an estimation on the number of points needed for the simulation of a guitar body was not available.

To draw a summary, several observations are made. The finite difference method can be used for simulating vibrations on essentially one dimensional objects very accurately. The other methods attempt to be more general at the cost of the accuracy and the detailed mathematic presentation of the vibratory phenomena. The finite difference method and the modal synthesis method provide tools for the study of real instruments. None of the methods is very well applicable for real-time sound synthesis purposes. The first reason is the computational cost when high-quality

synthesis is desired. Second, the parameters of the model are non-intuitive in a musical sense, and they are hard to control in the same way the actual instrument is controlled, especially in real-time performance situations. Finally, sound synthesis methods with more efficient computation and control exist, especially for string instruments and woodwinds with conical bores.

5. Digital Waveguides and Extended Karplus-Strong Models

Digital waveguides and *single delay loop* (SDL) models are the most efficient methods for physics-based real-time modeling of musical instruments. High-quality models exist for a number of musical instruments, and the research in this field is active.

In this chapter the digital waveguides are first discussed. Second, waveguide meshes, which are 2D and 3D models, are presented. The equivalence of the bi-directional digital waveguide model and the SDL model is detailed by (Karjalainen et al., 1998) and it will be described shortly. The last sections of the chapter present a case study of modeling the acoustic guitar using *commuted waveguide synthesis*.

5.1 Digital Waveguides

The concept of digital waveguides has been developed by Smith (1987, 1992, 1997). Digital waveguides and methods based on finite differences are closely related in that they both start from the premise of solving the wave equation. We recall from Section 4.1 that with the finite difference method the wave equation is solved in a set of discrete points on the vibrating object. At every time period a physical variable, such as displacement, is computed for every point. This implies that the vibratory motion of the whole discretized vibrating object is readily observable. While this may be attractive for the study of the vibrating object and the vibratory motion, more efficient methods are needed for sound synthesis purposes.

5.1.1 Waveguide for Lossless Medium

The digital waveguide is based on a general solution of the wave equation in a one-dimensional homogeneous medium. The lossless wave equation for a vibrating string can be expressed as (Morse and Ingard, 1968)

$$K\frac{\partial^2 y}{\partial x^2} = \varepsilon\frac{\partial^2 y}{\partial t^2}, \tag{5.1}$$

where K is the string tension, ε the linear mass density, and y the displacement of the string. This equation is applicable to any lossless one dimensional vibratory

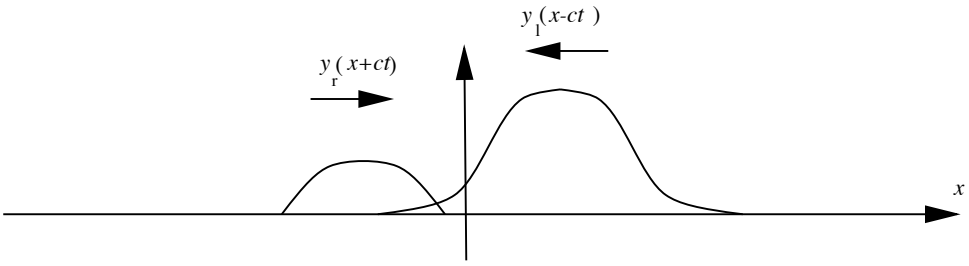

Figure 5.1: d'Alembert's solution of the wave equation.

motion, like that of the air column in the bore of a cylindrical woodwind instrument. Naturally in that case the parameters and the wave variables are interpreted accordingly. It can be seen by direct computation that the equation is solved by an arbitrary function of the form

$$\begin{aligned} y(x,t) &= y_{\mathrm{l}}(x-ct) \quad \text{or} \\ y(x,t) &= y_{\mathrm{r}}(x+ct), \end{aligned} \tag{5.2}$$

where

$$c = \sqrt{\frac{K}{\varepsilon}}.$$

The functions $y_{\mathrm{l}}(x-ct)$ and $y_{\mathrm{r}}(x+ct)$ can be interpreted as traveling waves going left and right, respectively. The general solution of the wave equation is a linear combination of the two traveling waves and it is pictured in Figure 5.1. This is the d'Alembert's solution to the wave equation.

The only restriction posed by the d'Alembert's solution is that the functions $y_{\mathrm{l}}(x-ct)$ and $y_{\mathrm{r}}(x+ct)$ have to be twice differentiable in both x and t. However, when the linear wave equation is developed for the real one-dimensional vibrator, the amplitude of the vibration is assumed to be small. Physically, in the case of a vibrating string this means that the slope of the vibrating string can only have values much lower than one. Similarly, vibrating air columns can exhibit only small variations of pressure around the static air pressure.

The digital waveguide is a discretization of the functions $y_{\mathrm{l}}(x-ct)$ and $y_{\mathrm{r}}(x+ct)$ and it is obtained by first changing the variables

$$x \longrightarrow x_m = mX$$

$$t \longrightarrow t_n = nT,$$

where T is the time step, X is the corresponding step in space, and m and n are the new integral-valued time and space variables. The new variables are related by

$$c = \frac{X}{T}.$$

Substitution of the new variables to the d'Alembert's solution of the wave equation

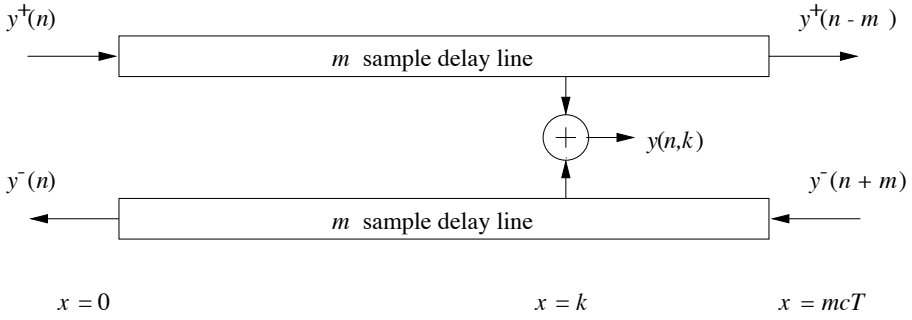

Figure 5.2: The one-dimensional digital waveguide, after (Smith, 1992).

yields

$$\begin{aligned} y(x_m, t_n) &= y_{\mathrm{r}}(t_n - \frac{x_m}{c}) + y_{\mathrm{l}}(t_n + \frac{x_m}{c}) \\ &= y_{\mathrm{r}}(nT - \frac{mX}{c}) + y_{\mathrm{l}}(nT + \frac{mX}{c}) \\ &= y_{\mathrm{r}}(T(n-m)) + y_{\mathrm{l}}(T(n+m)). \end{aligned} \tag{5.3}$$

Equation 5.3 can be simplified by defining

$$\begin{aligned} y^+(n) &= y_{\mathrm{r}}(nT) \quad \text{and} \\ y^-(n) &= y_{\mathrm{l}}(nT). \end{aligned} \tag{5.4}$$

In this notation, the + superscript denotes the traveling wave component going to the right and the − superscript the component going to the left.

Finally, the mathematical description of the digital waveguide is obtained with the two discrete functions $y^+(n-m)$ and $y^-(n+m)$ which can be interpreted as m-sample delay lines. The delay lines are pictured in Figure 5.2. The output from the waveguide at point k is obtained by summing the delay line variables at that point.

The solution to the one-dimensional wave equation provided by the waveguide is exact at the discrete points in the lossless case as long as the wavefronts are originally bandlimited to one half of the sampling rate. Bandlimited interpolation can be applied to estimate the values of the traveling waves at non-integral points of the delay lines. *Fractional delay filters* provide a convenient solution to bandlimited interpolation. See (Laakso et al., 1996) and (Välimäki, 1995) for more on fractional delay filters. A number of different physical quantities can be chosen as traveling waves. See Smith (1992, or 1995) for details on conversion between wave variables.

5.1.2 Waveguide with Dispersion and Frequency-Dependent Damping

In real vibrating objects, physical phenomena that account for attenuation of the vibratory motion are always present. These phenomena have to be incorporated in the model to obtain any realistic synthesis. In a general case, dispersion is also present. A wave equation that includes both the frequency-dependent damping and

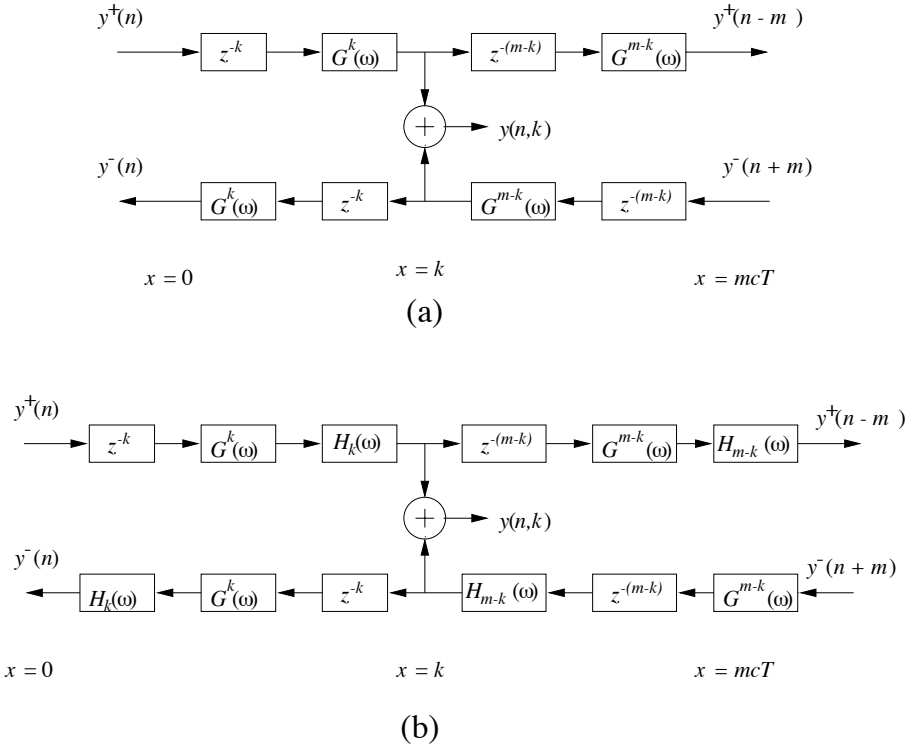

Figure 5.3: A lossy digital waveguide in (a). The frequency dependent gains $G(\omega)$ are lumped before observation points to obtain $G^k(\omega)$ in order to get more efficient implementation. In (b) dispersion is added in the form of allpass filters approximation the desired phase delay, after (Smith, 1995).

the dispersion is already presented for the vibrating string in Equation 4.1. The complete linear, time-invariant generalization of the wave equation for the lossy stiff string is described by Smith (1995).

A frequency-dependent gain factor $G(\omega)$ determines the frequency-dependent attenuation of the traveling wave for one time step. For a detailed derivation of an expression for $G(\omega)$ from the one-dimensional lossy wave equation, see (Smith, 1995, 1992). In the waveguide a gain factor that realizes $G(\omega)$ would have to be inserted between every unit delay. However, the system is linear and time-invariant and the gain factors can be commuted for every unobserved portion of the delay line. This is illustrated in Figure 5.3 (a) where the losses are consolidated before each observation point.

When the fourth-order derivative with respect to displacement y is present in the wave equation, the velocity of the traveling waves is not constant but it is dependent on the frequency. This is to say that the wavefront shape will be constantly evolving as the higher frequency components travel with a different velocity than the lower frequency components. This physical phenomenon is present in every physical string and it is called dispersion. The dispersion is mainly caused by stiffness of the string. For derivation of an expression for the frequency dependent velocity, see Smith (1995).

The dispersion can be taken into account in the waveguide model by inserting

an allpass filter before each observation point as is done in Figure 5.3 (b). The allpass filter $H_a(z)$ approximates the dispersion effect for a delay line of length a. Van Duyne and Smith (1994) present an efficient method for designing the allpass filter as a series of one-pole allpass filters. More recently, Rocchesso and Scalcon (1996) have presented a method to design an allpass filter based on analysis of the dispersion in recorded sound signals based on an allpass filter design method presented by Lang and Laakso (1994).

5.1.3 Applications of Waveguides

The digital waveguide has been applied to many sound synthesis problems (Smith, 1996). A short overview of applications in different instrument families is given.

The first physics-based approach to use digital filters to model a musical instrument was made for the violin by Smith (1983). Jaffe and Smith (1983) introduced several extensions to the Karplus-Strong algorithm that enable high-quality synthesis of plucked strings including an allpass filter in the delay loop to approximate the non-integral part of the delay.

Since those pioneer works, many improvements and further extensions have been presented for plucked string synthesis. These include Lagrange interpolation for fine-tuning the pitch and producing smooth glissandi (Karjalainen and Laine, 1991), and allpass filtering techniques to simulate dispersion caused by string stiffness (Smith, 1983), (Paladin and Rocchesso, 1992), and (Van Duyne and Smith, 1994). Commuted waveguide synthesis technique is an efficient way to include a high-quality model of an instrument body to waveguide synthesis. It has been proposed by (Smith, 1993) and (Karjalainen et al., 1993). Välimäki et al. (1995) have presented a method to produce smooth glissandi with allpass fractional delay filters. A parameter calibration method based on the STFT was developed by Karjalainen et al. (1993) and further elaborated by Välimäki et al. (1996). A similar approach is also made by Laroche and Jot (1992). These works are extended and an automated calibration system was implemented by Tolonen and Välimäki (1997). Multirate implementations of the string model and separate low-rate body resonators are presented by Smith (1993), Välimäki et al. (1996), and Välimäki and Tolonen (1997a, 1997b).

The plucked-string algorithm is also utilized to synthesize electric instrument tones. Sullivan (1990) extended the Karplus-Strong algorithm to synthesize electric guitar tones with distortion and feedback. Rank and Kubin (1997) have developed a model for slapbass synthesis.

Waveguide synthesis for the piano is presented by Smith and Van Duyne (1995) and Van Duyne and Smith (1995a) where a model of a piano hammer (Van Duyne and Smith, 1994), 2D digital waveguide mesh (Van Duyne and Smith, 1993a, see also Section 5.2), and allpass filtering techniques for simulating stiffness of the strings and the soundboard are combined together. Another development of the piano hammer model is presented by Borin and Giovanni (1996).

Waveguide synthesis has also been applied to several wind instruments. The clarinet was one of the first applications by Smith (1986), Hirschman (1991), Välimäki et al. (1992b), and Rocchesso and Turra (1993). A waveguide model for the flute has been proposed by Karjalainen and Laine (1991) and Välimäki et al. (1992a). Välimäki et al. (1993) propose a model for the finger holes in woodwind bores. Brass instrument tones have been simulated with waveguides by Cook (1991), Dietz and Amir (1995), Msallam et al. (1997), and Vergez and Rodet (1997). Cook (1992) has created a device that can control models of the wind instrument family.

SPASM is a DSP program by Cook (1993) to model the sound processing mechanism of a human in real time. It also provides a graphical user interface with an image of the vocal tract shape.

5.2 Waveguide Meshes

The digital waveguide presented in the previous section is very efficient in modeling one-dimensional vibrators. If modeling of vibratory motion in a 2D or 3D object is desired, the digital waveguide can be expanded to a waveguide mesh. Applications of waveguide meshes can be found, for instance, in modeling membranes, soundboards, cymbals, gongs, and room acoustics.

In this section, the two-dimensional waveguide mesh is discussed. Different implementation of the 2D waveguide mesh are given by Van Duyne and Smith (1993a, 1993b), Fontana and Rocchesso (1995), and Savioja and Välimäki (1997, 1996). Expansion to a three-dimensional mesh is relatively straightforward, as well as to the mathematically interesting N-dimensional mesh. An interesting 3D formulation not discussed here is the tetrahedral waveguide mesh presented by Van Duyne and Smith (1995b, 1996).

The traveling plane wave solution of the two-dimensional wave equation is given as (Morse and Ingard, 1968)

$$\frac{\partial^2 u(t,x,y)}{\partial t^2} = c^2\left[\frac{\partial^2 u(t,x,y)}{\partial x^2} + \frac{\partial^2 u(t,x,y)}{\partial y^2}\right] \tag{5.5}$$

$$\Leftrightarrow$$

$$u(t,x,y) = \int f_\alpha(x\cos(\alpha) + y\sin(\alpha) - ct)d\alpha. \tag{5.6}$$

where α denotes the direction of the plane wave. The integral involves an infinite number of traveling waves that are divided into components traveling in the x and y-directions.

5.2.1 Scattering Junction Connecting N Waveguides

To be able to formulate a waveguide mesh, a junction of waveguides needs to be developed. Connection of waveguides is pictured in Figure 5.4 where scattering junction S connects N bi-directional waveguides with impedances R_i, $i = 1, 2, \ldots, N$.

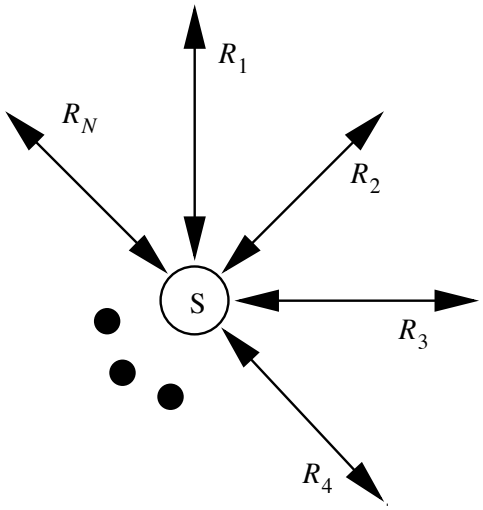

Figure 5.4: A scattering junction, after (Van Duyne and Smith, 1993b). N waveguides are connected together with no loss of energy.

For the connection to be physically meaningful, two conditions are required. The values of the wave variables, e.g., vibration velocities or sound pressures, have to be equal at the point of the junction

$$v_{\mathrm{S}} = v_1 = v_2 = \ldots = v_N\,, \tag{5.7}$$

where v_S is the value of the wave variable in the junction. Equation 5.7 states that the strings move together all the time. Second, the sum of the forces exerted by the strings or flows in the tubes must equal to zero

$$\sum_{k=1}^{N} f_k = 0. \tag{5.8}$$

Recalling from the previous section the definitions

$$v_k = v_k^+ + v_k^-, \quad f_k = f_k^+ + f_k^-, \quad f_k^+ = R_k v_k^+ \text{ and } f_k^- = -R_k v_k^-$$

the two constraints of Equations 5.7 and 5.8 can be developed further as

$$\begin{aligned}
\sum_{k=1}^{N} R_k v_k &= \sum_{k=1}^{N} R_k v_k^+ + \sum_{k=1}^{N} R_k v_k^- \\
&= \sum_{k=1}^{N} R_k v_k^+ + \sum_{k=1}^{N} R_k v_k^- + \overbrace{\sum_{k=1}^{N} R_k v_k^+ - \sum_{k=1}^{N} R_k v_k^-}^{\text{equals 0}} \\
&= 2\sum_{k=1}^{N} R_k v_k^+.
\end{aligned}$$

Now using $v_{\mathrm{S}} = v_k$ an expression for the wave variable at the junction is obtained

as

$$v_{\mathrm{S}} = \frac{2\sum_{k=1}^{N} R_k v_k^+}{\sum_{k=1}^{N} R_k}. \tag{5.9}$$

The outputs of the junction are obtained by applying $v_{\mathrm{S}} = v_k = v_k^+ + v_k^-$ as

$$v_k^- = v_{\mathrm{S}} - v_k^+ \tag{5.10}$$

5.2.2 Two-Dimensional Waveguide Mesh

The rectilinear waveguide mesh formulation of the two-dimensional wave equation consists of delay elements and 4-port scattering junctions. Such a system is pictured in Figure 5.5. The scattering junctions are marked with $\mathrm{S}_{l,m}$ where l denotes the index to the x-direction and m to the y-direction. The discrete time variable is n. The two delay elements between the ports of consecutive scattering junctions form a bi-directional delay unit.

If the medium is assumed isotropic, the impedances R_k are equal and the junction equations for junction $\mathrm{S}_{l,m}$, denoted S for convenience, are obtained from Equations 5.9 and 5.10 as

$$v_{\mathrm{S}}(n) = \frac{1}{2}\sum_{k=1}^{4} v_k^+(n) \tag{5.11}$$

and

$$v_k^-(n) = v_{\mathrm{S}}(n) - v_k^+(n), \quad k = 1, 2, 3, 4. \tag{5.12}$$

This formulation can be interpreted as a finite difference approximation of the two-dimensional wave equation as shown by Van Duyne and Smith (1993a, 1993b).

5.2.3 Analysis of Dispersion Error

The formulation of the waveguide mesh presented above has some drawbacks. The wave propagation speed and magnitude response depend on both the direction of wave motion and frequency. This can be illustrated by examining the two-dimensional discrete Fourier transform of the finite difference scheme. The 2D DFT produces a 2D frequency space so that each point (ξ_1, ξ_2) corresponds to a spatial frequency

$$\xi = \sqrt{\xi_1^2 + \xi_2^2}.$$

The coordinates ξ_1 and ξ_2 of the 2D frequency space are taken to correspond to x and y dimensions of the waveguide mesh, respectively.

The ratio of the actual propagation speed to the desired propagation speed in the rectilinear waveguide mesh can be computed as (Van Duyne and Smith, 1993a)

$$\frac{c'(\xi_1, \xi_2)}{c} = \frac{\sqrt{2}}{\xi T} \arctan \frac{\sqrt{4 - b^2}}{b}, \tag{5.13}$$

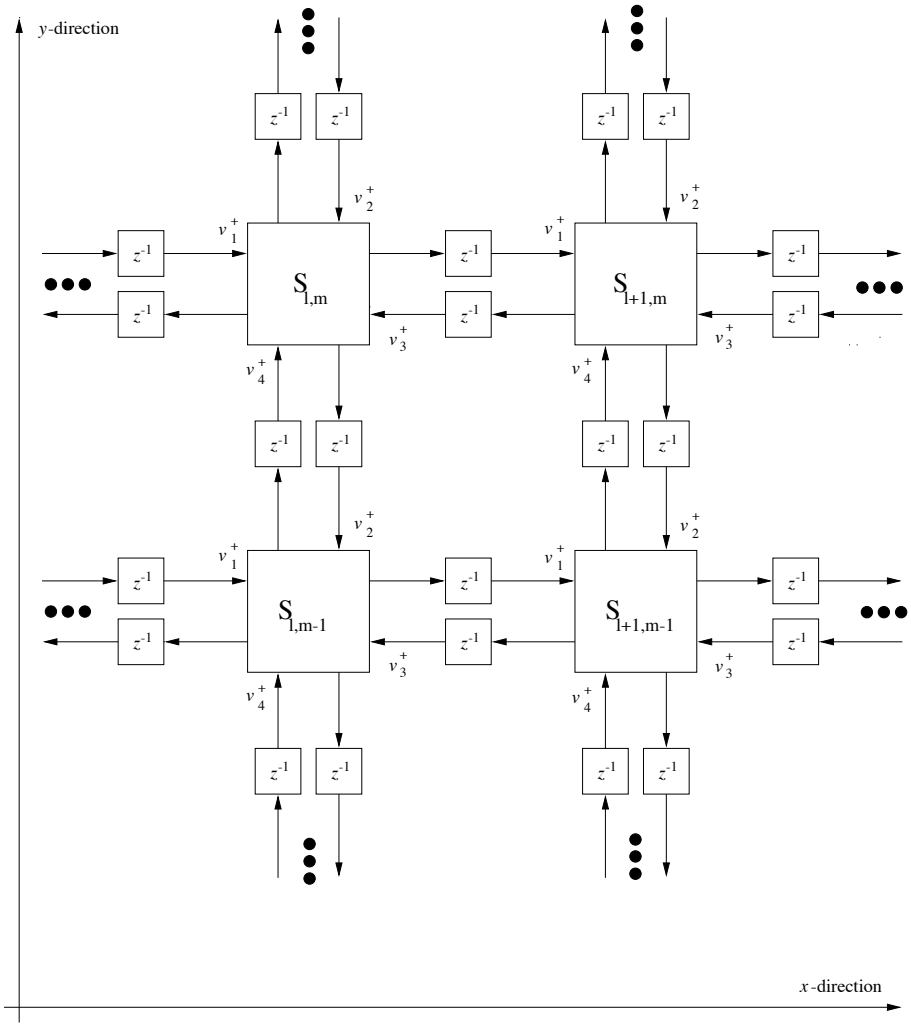

Figure 5.5: Block diagram of a 2D waveguide mesh, after (Van Duyne and Smith, 1993a).

where

$$b = \cos(\xi_1 T) + \cos(\xi_2 T), \tag{5.14}$$

and T is the sampling interval.

The effect of dispersion error can be suppressed using different types of waveguide formulations. Savioja and Välimäki (1996, 1997) propose to use an interpolated waveguide mesh that utilizes deinterpolation to approximate unit delays in the diagonal directions. Fontana and Rocchesso (1995) suggest a tessellation of the ideal membrane into triangles. The ratio of propagation speeds is pictured in Figure 5.6 as a function of frequency for four different types of waveguide formulations. In Figure 5.6 (a) (Van Duyne and Smith, 1993a), the speed ratio of the rectilinear formulation is pictured. In (b) and (c) the speed ratios of a hypothetical (non-realizable) 8-directional waveguide and a deinterpolated 8-directional waveguide are depicted (Savioja and Välimäki, 1997). In Figure 5.6 (d) the speed ratio of the triangular tessellation is illustrated (Fontana and Rocchesso, 1995).

The distance from center of the plots in Figures 5.6 (a) - (d) corresponds to the

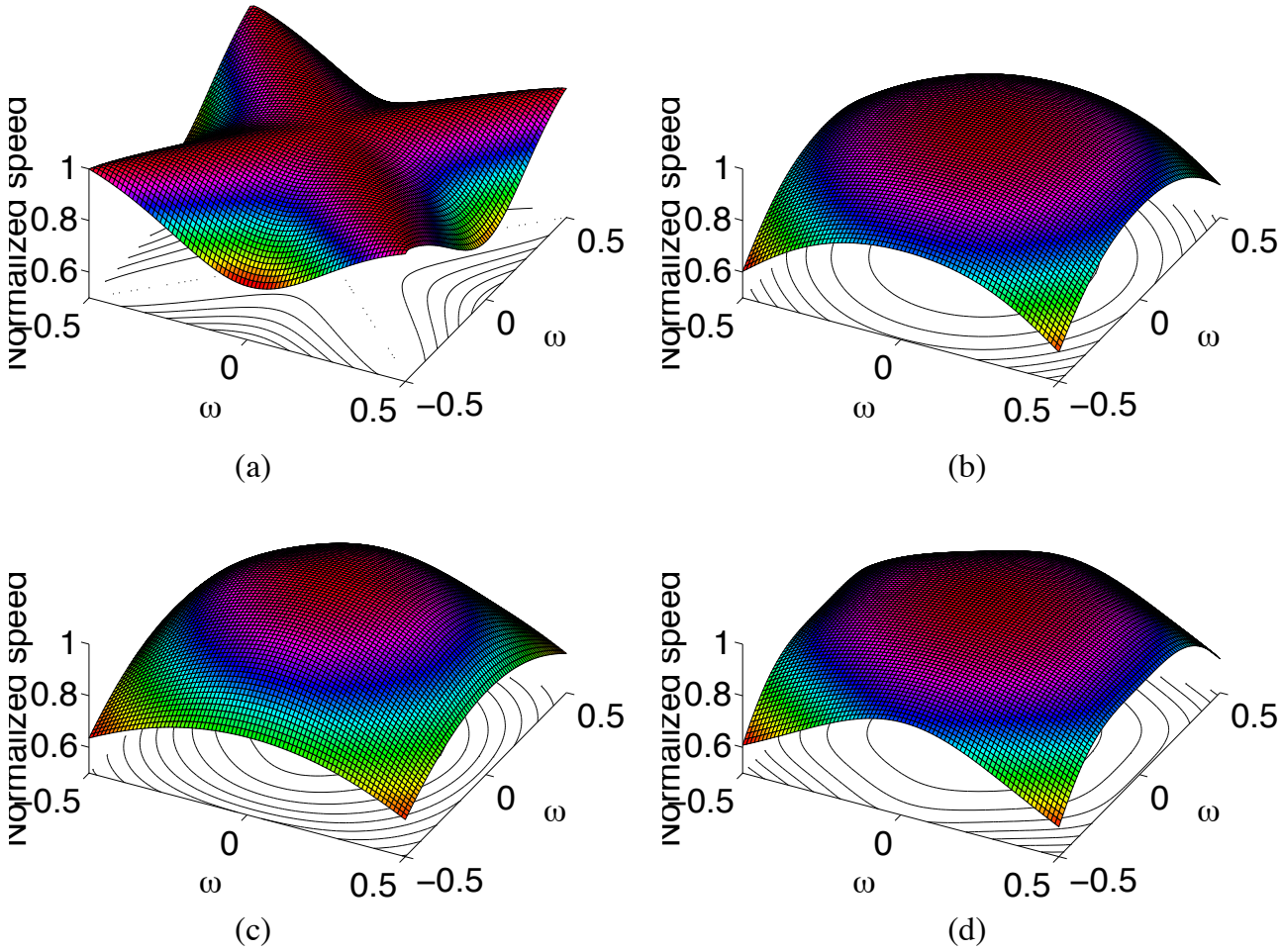

Figure 5.6: Dispersion in digital waveguides. The wave propagation speed is plotted as a function of spatial frequency and direction for a rectilinear mesh in (a), for a hypothetical 8-directional mesh in (b), for a deinterpolated 8-directional mesh in (c), and for a triangular tessellation in (d). The spatial frequency is the distance from the origin and the $\xi_1 T$- and $\xi_2 T$-axis of the horizontal plane correspond to x and y directions.

spatial frequency. The axis of the horizontal plane are the ξ_1- and ξ_2-axis which correspond to the x- and y-directions in the mesh, respectively. The contours of equal ratios are pictured on the bottom of each figure. The dependence of the propagation speed ratio on both the frequency and direction can be seen in (a). In other figures, the dependence on the direction can be observed to be less severe. It should be noted that the mesh of (b) is not realizable.

5.3 Single Delay Loop Models

First extensions to the Karplus-Strong algorithm presented in Section 2.3 were derived by Jaffe and Smith (1983). Even before that, Smith (1983) had developed a model for the violin that included a string model that is similar to the generic Karplus-Strong model in Figure 2.5 (b). Those works were the first to take a physical modeling interpretation of the Karplus-Strong model.

The digital waveguide presented in the previous chapter can be developed to an SDL model[1] in certain situations. In this chapter, the SDL model is derived for the guitar, as has been done by Karjalainen et al. (1998). In this context, only the case of a string with force signal output at the bridge will be considered. This corresponds to the construction of a classical acoustical guitar. The case of pickup output, which corresponds to electric guitars, is presented by Karjalainen et al. (1998). We start with a continuous-time waveguide model in the Laplace domain and develop a discrete-time model which can be identified as an SDL model.

In the discussion to follow the transfer functions of the model components are described in the Laplace transform domain. The Laplace transform is an efficient tool in linear continuous-time systems theory. Particularly, time-domain integration and derivative operations transform into division and multiplication by the Laplace variable s, respectively. The complex Laplace variable s may be changed with $j\omega$ (where j is the imaginary unit $\sqrt{-1}$, ω is the radian frequency $\omega = 2\pi f$, and f is the frequency in Hz), in order to derive the corresponding representation in the Fourier transform domain, i.e., the frequency domain. For a discrete-time implementation, the continuous-time system is finally approximated by a discrete-time system in the z-transform domain. For more information on Laplace, Fourier, and z-transforms, see a standard textbook on signal processing, such as (Oppenheim et al., 1983).

In the next subsection, a waveguide model for the acoustic guitar is presented. In the one after that, the digital waveguide representation is developed into an SDL model.

[1] In this document the models of a vibrating string consisting of a loop with single delay line are called single delay loop (SDL) models to distinguish them from both the non-physical KS algorithm and the bidirectional digital waveguide models.

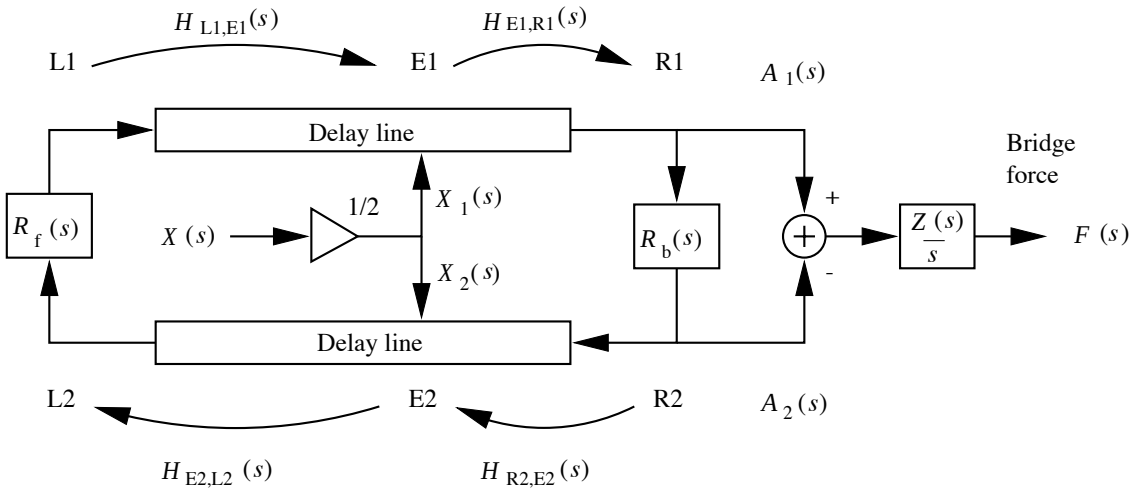

Figure 5.7: Dual delay-line waveguide model for a plucked string with a force output at the bridge (Karjalainen et al., 1998).

5.3.1 Waveguide Formulation of a Vibrating String

In Figure 5.7 a dual delay-line waveguide model for an ideally plucked acoustic guitar string with transversal bridge force as an output is presented. The delay lines and the reflection filters $\mathcal{R}_\mathrm{b}(s)$ and $\mathcal{R}_\mathrm{f}(s)$ form a loop in which the waveforms circulate. The two reflection filters simulate the reflection of the waveform at the termination points of the vibrating part of the string at the bridge and at the corresponding fret, respectively. The filters are phase-inversive, i.e., they have negative signs, and they also contain slight frequency-dependent damping. Let us assume for now that the delay lines correspond to the d'Alembert's solution of the wave equation for a stiff and lossy string. In this case they are dispersive and they also attenuate the signal continuously in a frequency-dependent manner.

Pluck excitation $\mathcal{X}(s)$ is divided into two parts $\mathcal{X}_1(s)$ and $\mathcal{X}_2(s)$, so that $\mathcal{X}_1(s) = \mathcal{X}_2(s) = \mathcal{X}(s)/2$. The excitation parts are fed into the waveguides at points E1 and E2. It has been shown by Smith (1992) that an ideal pluck of the string can be approximated by a unit impulse if acceleration waves are used. Thus, it is attractive to choose acceleration as the wave variable and, in this context, $\mathcal{A}_1(s)$ and $\mathcal{A}_2(s)$ correspond to the values of the right and the left traveling acceleration waves at positions R1 and R2, respectively.

The output signal of interest is the transverse force $\mathcal{F}(s)$ applied at the bridge by the vibrating string. It is obtained from the acceleration wave components $\mathcal{A}_1(s)$ and $\mathcal{A}_2(s)$ as

$$\mathcal{F}(s) = \mathcal{F}^+(s) + \mathcal{F}^-(s) = \mathcal{Z}(s)[\mathcal{V}^+(s) - \mathcal{V}^-(s)] = \mathcal{Z}(s)\frac{1}{s}[\mathcal{A}_1(s) - \mathcal{A}_2(s)], \quad (5.15)$$

i.e., the bridge force $\mathcal{F}(s)$ is the bridge impedance $\mathcal{Z}(s)$ times the difference of the string velocity components $\mathcal{V}^+(s)$ and $\mathcal{V}^-(s)$ at the bridge. In the last form of Equation 5.15 the velocity difference $\mathcal{V}^+(s) - \mathcal{V}^-(s)$ is expressed as integrated acceleration difference $\frac{1}{s}(\mathcal{A}_1(s) - \mathcal{A}_2(s))$.

Figure 5.7 also includes the transfer functions between the unobserved and unmodified parts of the waveguides; $\mathcal{H}_{\mathrm{A,B}}(s)$ refers to the transfer function from point A to point B. These transfer functions are elaborated in the following subsection where the bi-directional digital waveguide model is reformulated as an SDL model.

5.3.2 Single Delay Loop Formulation of the Acoustic Guitar

In the waveguide formulation pictured in Figure 5.7 there are four points, namely, E1, E2, R1 and R2, at which either a signal ($\mathcal{X}_1(s)$ and $\mathcal{X}_2(s)$) is fed to the waveguide or the wave variables ($\mathcal{A}_1(s)$ and $\mathcal{A}_2(s)$) are observed. It is immediately apparent that the formulation can be simplified by combining transfer function $\mathcal{R}_{\mathrm{f}}(s)$ and transfer functions $\mathcal{H}_{\mathrm{E2,L2}}(s)$ and $\mathcal{H}_{\mathrm{L1,E1}}(s)$ of the two parts of the lossy and dispersive waveguide to the left of the excitation point E1 and E2. However, it is more efficient to attempt to reduce the number of points in which the wave variables are processed or observed.

The explicit input to the lower delay line can be removed by deriving an equivalent single excitation at point E1 that corresponds to the net effect of the two excitation components at points E1 and E2. The equivalent single excitation at E1 can be expressed as [2]

$$\begin{aligned} \mathcal{X}_{\mathrm{E1,eq}}(s) &= \mathcal{X}_1(s) + \mathcal{H}_{\mathrm{E2,L2}}(s)\mathcal{R}_{\mathrm{f}}(s)\mathcal{H}_{\mathrm{L1,E1}}(s)\mathcal{X}_2(s) \\ &= \frac{1}{2}[1 + \mathcal{H}_{\mathrm{E2,E1}}(s)]\mathcal{X}(s) \\ &= \mathcal{H}_{\mathrm{E}}(s)\mathcal{X}(s), \end{aligned} \tag{5.16}$$

where $\mathcal{H}_{\mathrm{E2,E1}}(s)$ is the left-side transfer function from E2 to E1 consisting of the two parts of the lossy and dispersive delay lines $\mathcal{H}_{\mathrm{E2,L2}}(s)$ and $\mathcal{H}_{\mathrm{L1,E1}}(s)$, and reflection function $\mathcal{R}_{\mathrm{f}}(s)$. Thus, $\mathcal{H}_{\mathrm{E}}(s)$ is the equivalent excitation transfer function.

In a similar fashion, one of the explicit output points can be removed in order to obtain a structure with only single input and output positions. Since the guitar body is driven by the force applied by the vibrating string at the bridge, it is apparent that an acceleration-to-force transfer function is required. In Equation 5.15 output force $\mathcal{F}(s)$ is expressed in terms of acceleration waves $\mathcal{A}_1(s)$ and $\mathcal{A}_2(s)$. This is further elaborated as

$$\begin{aligned} \mathcal{F}(s) &= \mathcal{Z}(s)\frac{1}{s}[\mathcal{A}_1(s) - \mathcal{A}_2(s)] \\ &= \mathcal{Z}(s)\frac{1}{s}[\mathcal{A}_1(s) - \mathcal{R}_{\mathrm{b}}(s)\mathcal{A}_1(s)] \\ &= \mathcal{Z}(s)\frac{1}{s}[1 - \mathcal{R}_{\mathrm{b}}(s)]\mathcal{A}_1(s) \\ &= \mathcal{H}_{\mathrm{B}}(s)\mathcal{A}_1(s), \end{aligned} \tag{5.17}$$

where $\mathcal{H}_{\mathrm{B}}(s)$ is the acceleration-to-force transfer function at the bridge. Notice that it only depends on $\mathcal{A}_1(s)$, the wave variable of the upper delay line. In a similar fashion one can derive an expression for $\mathcal{F}(s)$ depending only on $\mathcal{A}_2(s)$.

[2] 'eq' in $\mathcal{X}_{\mathrm{E1,eq}}(s)$ stands for 'equivalent'.

To develop an expression for $\mathcal{A}_1(s)$ in terms of the equivalent input $\mathcal{X}_{\mathrm{E1,eq}}(s)$, we first write

$$\mathcal{A}_1(s) = \mathcal{H}_{\mathrm{E1,R1}}(s)\mathcal{X}_{\mathrm{E1,eq}}(s) + \mathcal{H}_{\mathrm{loop}}(s)\mathcal{A}_1(s), \tag{5.18}$$

where

$$\mathcal{H}_{\mathrm{loop}}(s) = \mathcal{R}_{\mathrm{b}}(s)\mathcal{H}_{\mathrm{R2,E2}}(s)\mathcal{H}_{\mathrm{E2,E1}}(s)\mathcal{H}_{\mathrm{E1,R1}}(s), \tag{5.19}$$

i.e., $\mathcal{H}_{\mathrm{loop}}(s)$ is the transfer function when the signal is circulated once around the loop. Thus, the sum terms of Eq. 5.18 correspond to the equivalent excitation signal $\mathcal{X}_{\mathrm{E1,eq}}(s)$ transfered to point R1 and signal $\mathcal{A}_1(s)$ transfered once along the loop. Solving Equation 5.18 for $\mathcal{A}_1(s)$, we obtain

$$\begin{aligned} \mathcal{A}_1(s) &= \mathcal{H}_{\mathrm{E1,R1}}(s)\frac{1}{1-\mathcal{H}_{\mathrm{loop}}(s)}\mathcal{X}_{\mathrm{E1,eq}}(s) \\ &= \mathcal{H}_{\mathrm{E1,R1}}(s)\mathcal{S}(s)\mathcal{X}_{\mathrm{E1,eq}}(s), \end{aligned} \tag{5.20}$$

where $\mathcal{S}(s)$ is the string transfer function that represents the recursion around the string loop.

Finally, the overall transfer function from excitation to bridge output is written as

$$\mathcal{H}_{\mathrm{E,B}}(s) = \frac{\mathcal{F}(s)}{\mathcal{X}(s)} = \frac{1}{2}[1+\mathcal{H}_{\mathrm{E2,E1}}(s)]\frac{\mathcal{H}_{\mathrm{E1,R1}}(s)}{1-\mathcal{H}_{\mathrm{loop}}(s)}\mathcal{Z}(s)\frac{1}{s}[1-\mathcal{R}_{\mathrm{b}}(s)], \tag{5.21}$$

or more compactly, based on the above notation

$$\mathcal{H}_{\mathrm{E,B}}(s) = \mathcal{H}_{\mathrm{E}}(s)\mathcal{H}_{\mathrm{E1,R1}}(s)\mathcal{S}(s)\mathcal{H}_{\mathrm{B}}(s), \tag{5.22}$$

which represents the cascaded contribution of each part in the physical string system.

At this point the continuous-time model of the acoustic guitar in the Laplace transform domain is approximated with a discrete-time model in the z-transform domain. This approximation is needed in order to make the model realizable in a discrete-time form. We rewrite Equation 5.22 in the z-transform domain

$$H_{\mathrm{E,B}}(z) = H_{\mathrm{E}}(z)H_{\mathrm{E1,R1}}(z)S(z)H_{\mathrm{B}}(z), \tag{5.23}$$

where

$$H_{\mathrm{E}}(z) = \frac{1}{2}[1+H_{\mathrm{E2,E1}}(z)] \tag{5.23a}$$

$$S(z) = \frac{1}{1-H_{\mathrm{loop}}(z)} \tag{5.23b}$$

$$H_{\mathrm{B}}(z) = Z(s)I(z)[1-R_{\mathrm{b}}(z)]. \tag{5.23c}$$

Filter $I(z)$ is a discrete-time approximation of the time-domain integration operation.

Equation 5.23 is interpreted by examining a block diagram in Figure 5.8. It shows qualitatively the delays and the discrete-time approximations of the cascaded filter

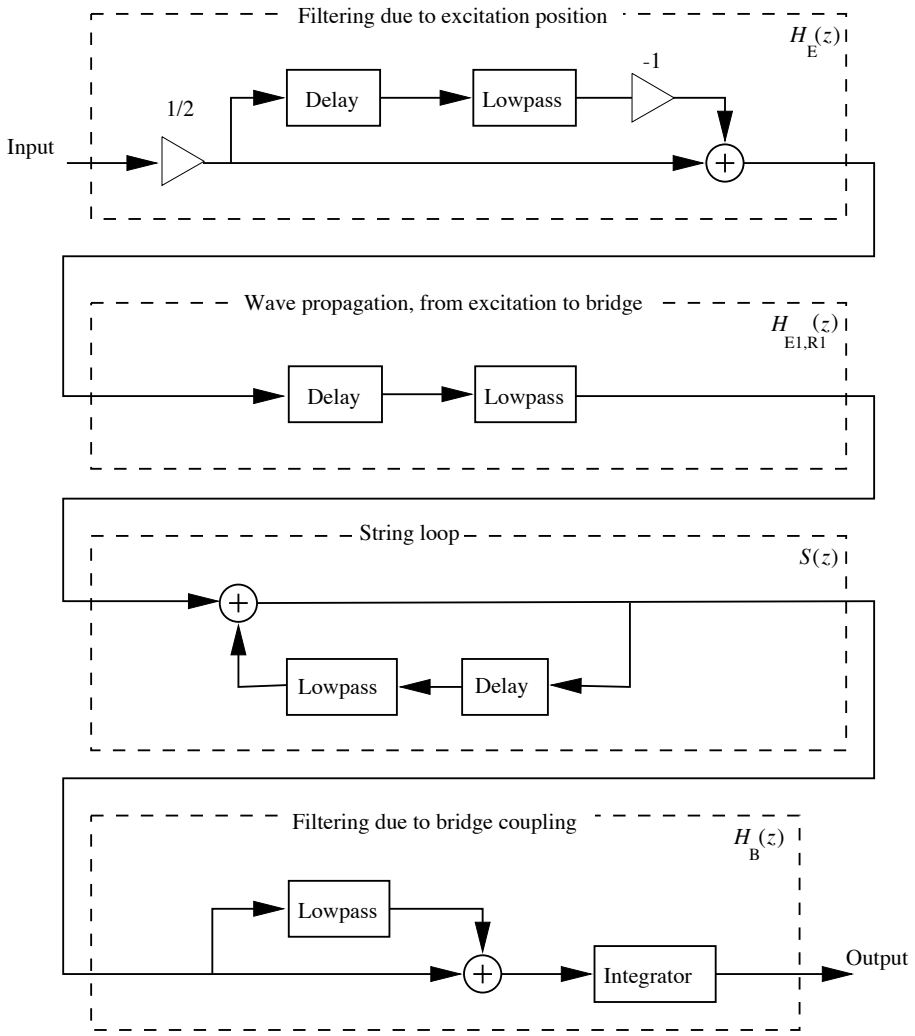

Figure 5.8: A block diagram of transfer function components as a model of the plucked string with force output at the bridge (Karjalainen et al., 1998).

components in Equation 5.21. The first block corresponding to $H_{\mathrm{E}}(z)$ simulates the comb filtering effect depending on the pluck position. Notice that the phase inversion of the reflection filter is explicated with the multiplication by -1. The second block corresponding to the transfer function from E1 to R1 in Figure 5.7 has a minor effect and is usually discarded in the final implementation of the model. This reduction is justified by noticing that the gain term $G(\omega)$ determining the attenuation of the traveling wave in one time step is extremely close to unity, and thus in the short time it takes for the wave to travel from E1 to R1 the attenuation is negligible. The third block in Figure 5.8 is the string loop and it simulates the vibration of the string. The delay in the loop corresponds in length to the sum of the two delay lines in Figure 5.7. the losses of a single round in the loop are consolidated in the lowpass filter. In the last block, the feedforward filter is typically discarded and only the integrator is implemented. In this case the lowpass filter corresponds to the opposite of reflection filter at the bridge and it is very close to unity. Thus, the sum of the filtered and the direct signal is approximated as being equal to 2.

Notice that the model of the acoustic guitar presented in Figure 5.8 is indeed a

single delay loop model, with the only loop in the third block. The presented model describes the vibration of a plucked string. It includes the effects of the plucking position and the output signal corresponds to the force applied by the string at the guitar body in a physically relevant manner. In the next section this model is extended to include models of the guitar body, the two vibration polarizations and sympathetic couplings between the strings.

5.4 Single Delay Loop Model with Commuted Body Response

The sound production mechanism of the guitar can be divided into three functional substructures, namely, the excitation, the vibration of the string, and the radiation from the guitar body. It is advantageous to retain this functional partition when developing a computational model of the acoustic guitar, as suggested by many studies presented in the literature (Smith, 1993; Karjalainen et al., 1993; Välimäki et al., 1996; Karjalainen and Smith, 1996; Tolonen and Välimäki, 1997; Välimäki and Tolonen, 1997a). In the previous section a detailed model for the vibration of a single string was described. In order to obtain a high-quality simulation of the acoustic guitar, the excitation and the body models have to be incorporated in the instrument model. The model should be sufficiently general to accommodate such effects as those produced by the two vibration polarizations and sympathetic couplings between the strings.

In the virtual instrument, the excitation model determines the amplitude of the sound, the plucking type, and the effect of the plucking point, while the body model gives an identity to the instrument, i.e., it determines what type of a guitar is being modeled. The body model includes the body resonances of the instrument and determines the directional properties of the radiation. The directional properties are not included in the model presented here, but they can be added by post-processing the synthesized signal (Huopaniemi et al., 1994; Karjalainen et al., 1995).

In this section, the principle of *commuted waveguide synthesis* (CWS) (Smith, 1993; Karjalainen et al., 1993) is first discussed for efficient realization of the excitation and body models. Second, a physical model that includes the aforementioned features is presented. In this context the synthetic acoustic guitar is only discussed generally without going into details of the DSP structures.

5.4.1 Commuted Model of Excitation and Body

The body of the acoustic guitar is a complex vibrating structure. Karjalainen et al. (1991) have reported that in order to fully model the response of the body, a digital all-pole filter of order 400 or more is required. However, this kind of implementation is impractical since it would be computationally far too expensive for real-time applications. Commuted waveguide synthesis (Smith, 1993; Karjalainen et al., 1993) can be applied to include the body response in the synthetic guitar signal

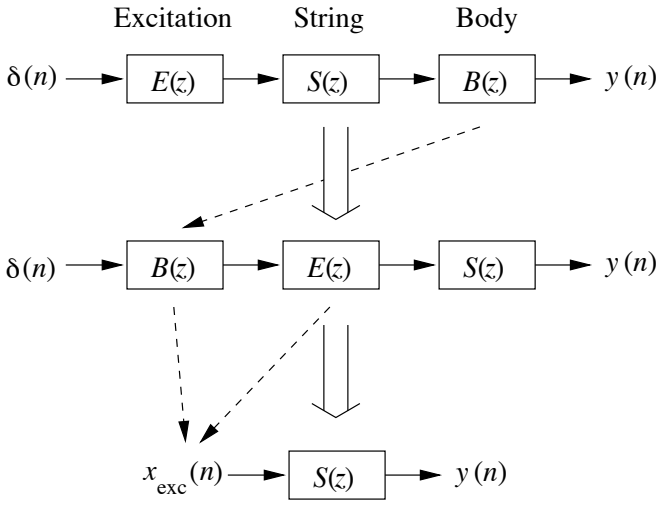

Figure 5.9: The principle of commuted waveguide synthesis. On the top, the instrument model is presented as three linear filters. In the middle, the body model $B(z)$ is commuted with the excitation and string models $E(z)$ and $S(z)$. On the bottom, the body and excitation models are convolved into a single response $x_{\text{exc}}(n)$ that is used to excite the virtual guitar.

in a computationally efficient manner. It is based on the theory of linear systems, and particularly, on the principle of commutation.

In CWS the instrument model is interpreted as pictured on the top of Figure 5.9, i.e., as the excitation, the vibrating string, and the radiating body. These parts are presented as linear filters with transfer functions $E(z)$, $S(z)$, and $B(z)$, respectively. Since the system is excited with an impulse $\delta(n)$, the cascaded configuration implies that the output signal $y(n)$ is obtained as a convolution of the impulse responses $e(n)$, $s(n)$, and $b(n)$ of the three filters and the unit impulse $\delta(n)$, i.e.,

$$y(n) = \delta(n) * e(n) * s(n) * b(n) = e(n) * s(n) * b(n), \tag{5.24}$$

where $*$ denotes the convolution operator defined as

$$h_1(n) * h_2(n) = \sum_{k=-\infty}^{\infty} h_1(k) h_2(n-k). \tag{5.25}$$

In the z-transform domain Equation 5.24 is expressed as

$$Y(z) = E(z)S(z)B(z). \tag{5.26}$$

Since we approximate the behavior of the instrument parts with linear filters, we can apply the principle of commutation and rewrite Equation 5.26 as

$$Y(z) = B(z)E(z)S(z), \tag{5.27}$$

as illustrated in the middle part of Figure 5.9. In practice, it is useful to convolve the impulse responses $b(n)$ and $e(n)$ of the body and excitation models into a single

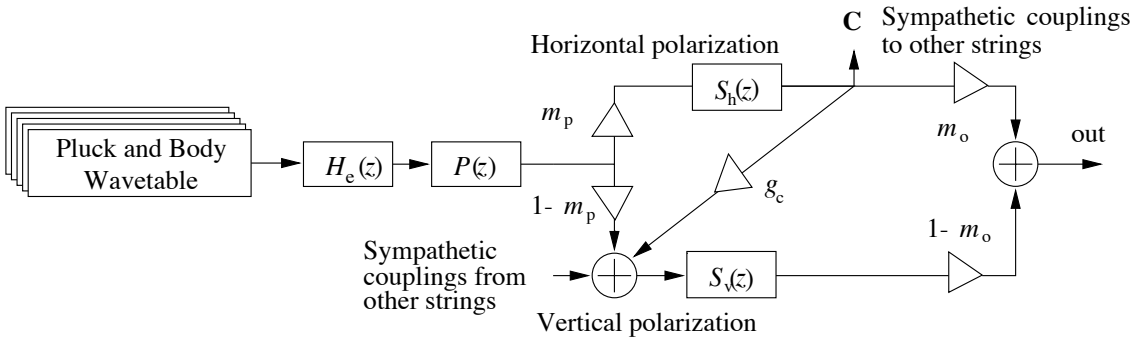

Figure 5.10: An extended string model with dual-polarization vibration and sympathetic coupling (Karjalainen et al., 1998).

impulse response denoted by $x_{\mathrm{exc}}(n)$ on the bottom of Figure 5.9. This signal is used to excite the string model and it can be precomputed and stored in a wavetable. Typically, several excitation signals are used for one instrument. The excitation signal is varied depending on the string and fret position as well as on the playing style. Computation of the excitation signal from a recorded guitar tone is presented in Section 5.4.3.

There are several other ways to incorporate a model of the instrument body in a realizable form. These are discussed by Karjalainen and Smith (1996) and they include methods of reducing the filter order using a conformal mapping to warp the frequency axis into a domain that better approximates the human auditory system, and of extracting the most prominent modes in the body response. These modes are reproduced to the synthetic signal by computationally cheap filters (Karjalainen and Smith, 1996; Tolonen, 1998).

5.4.2 General Plucked String Instrument Model

The model illustrated in Figure 5.10 exemplifies a general model for the acoustic guitar string employing the principle of commuted synthesis. A library of different pluck types and instrument bodies is stored in a wavetable on the left. The excitation signal is modified with a pluck shaping filter $H_{\mathrm{e}}(z)$ which brings about brightness and gain control, and a pluck position equalizer $P(z)$ which simulates the effect of the excitation position to the synthetic signal. The pluck position equalizer corresponds to the transfer function component presented on top of Figure 5.8.

After the excitation signal is fetched from a wavetable and filtered by transfer functions $H_{\mathrm{e}}(z)$ and $P(z)$, it is fed to the two string models $S_{\mathrm{h}}(z)$ and $S_{\mathrm{v}}(z)$ in a ratio determined by the gain parameter m_{p}. The string models simulate the effect of the two polarizations of the transversal vibratory motion and they are typically slightly mistuned in delay line lengths and decay rates to produce a natural-sounding synthetic tone. The output signal is a sum of the outputs of the two polarization models mixed in a ratio determined by m_{o}.

In the instrument model of Figure 5.10 sympathetic couplings between the strings

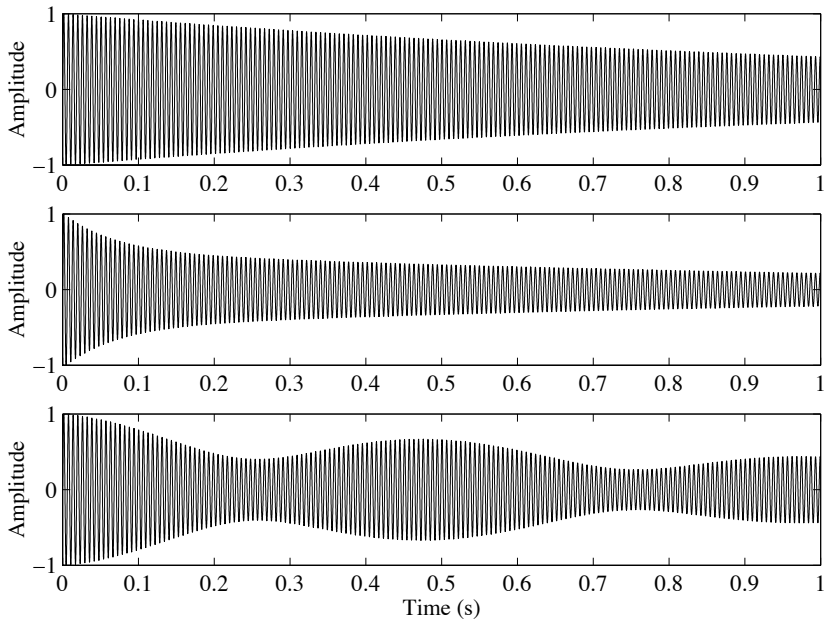

Figure 5.11: An example of the effect of mistuning the polarization models $S_{\mathrm{h}}(z)$ and $S_{\mathrm{v}}(z)$. Top: equal parameter values, middle: mistuned decay rates, and bottom: mistuned fundamental frequencies.

are implemented by feeding the output of the horizontal polarization to a connection matrix $\mathbf{C}$ which consists of the coupling coefficients. The matrix is expressed as

$$\mathbf{C} = \begin{bmatrix} g_{c1} & c_{12} & c_{13} & \cdots & c_{1N} \\ c_{21} & g_{c2} & c_{23} & & \\ c_{31} & c_{32} & g_{c3} & & \vdots \\ \vdots & & & \ddots & \\ c_{N1} & & \cdots & & g_{cN} \end{bmatrix} \tag{5.28}$$

where N is the number of dual-polarization strings, the coefficients g_{ck} (for $k = 1, 2, \ldots, N$) denote the gains of the output signal to be sent from the k^{th} horizontal string to its parallel vertical string, and coefficients c_{mk} are the gains of the k^{th} horizontal string output to be sent to the m^{th} vertical string. Notice that the gain terms g_{ck} implement a coupling between the two polarization in the k^{th} string and that the coefficient g_c is also presented explicitly in the figure. With this kind of structure, it is possible to obtain a simulation of both sympathetic coupling between string and coupling of the two polarizations within a string. The structure is inherently stable since there are no feedback paths in the model. Notice also that with parameters m_{p} and m_{o} it is possible to change the configuration of the virtual instrument. For instance, by setting $m_{\mathrm{p}} = 1$, the vertical polarization will act as a resonance string with the only input obtained from the horizontal polarization.

An example of the effect of mistuning the two polarization models is shown in Figure 5.11. On the top of the figure, the model parameters are equal and an exponential decay is resulted. In the middle, the fundamental frequencies of the models are equal but the loop filter parameters differ from each other and a two-

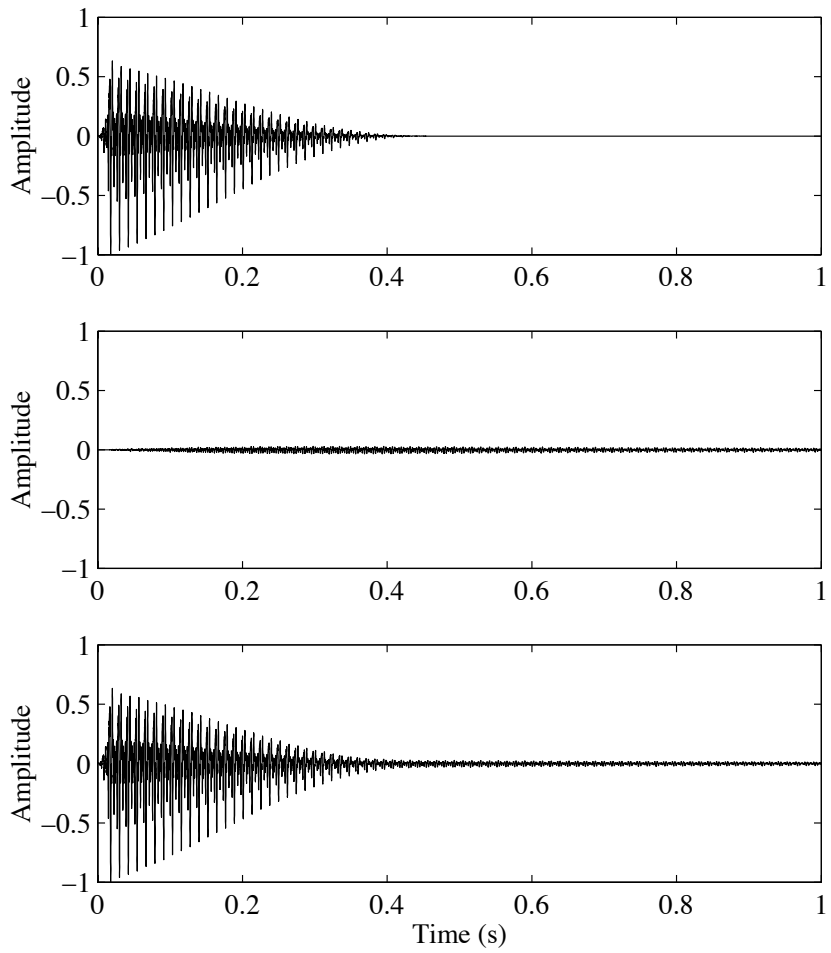

Figure 5.12: An example of sympathetic coupling. The output of a tone E_2 played on the 6^{th} string of the virtual guitar is plotted on the top. In the middle, the sum of the outputs of the other virtual strings vibrating due to the sympathetic coupling is illustrated. The output of the virtual instrument, i.e., the sum of all the string outputs, is presented on the bottom.

stage decay is produced. On the bottom figure, the loop filter parameters are equal while the frequencies are mistuned to obtain a beating effect. Another example is pictured in Figure 5.12 illustrating the sympathetic couplings between the strings. Tone E_2 is played on the 6^{th} string of the virtual instrument, and the vibration of the string is soon damped by the player. On the top part the output of the plucked string is depicted. In the middle, the summed output of the other strings vibrating sympathetically is illustrated. On the bottom the output of the virtual instrument is plotted. Notice that the other strings continue to vibrate after the primary vibration is damped.

5.4.3 Analysis of the Model Parameters

After the instrument model has been constructed both to closely simulate the physical behavior of a real instrument and to be efficiently realizable in real time, the model parameters have to be derived. It is natural to start by recording tones of a

real acoustic guitar. Since the recordings are treated as acoustic measurements of a sound production system, they have to be performed carefully. The side effects of the environment, such as noise and the response of the room, should be minimized.

An analysis scheme is proposed by Tolonen (1998). In this approach sinusoidal modeling is used to obtain the decaying partials of the guitar tone as separate additive signal components. It is shown that the sinusoidal modeling approach is well suited for this kind of parameter estimation problem.

6. Evaluation Scheme

The sound synthesis methods presented in this document have been developed to different types of synthesis problems. Thus it is not appropriate to compare these methods with each other since the evaluation criteria, no matter how carefully chosen, would favor some of the methods. The purpose of the evaluation is to give some guidelines on which methods are best suited for a given sound synthesis problem.

The methods presented in this document were divided into four groups, based on a taxonomy presented by Smith (1991), to better compare techniques that are closely related to each other. The groups are: abstract algorithms, sampling and processed recordings, spectral modeling synthesis, and physical modeling. This division is based on the premises of each sound synthesis method. Abstract algorithms create interesting sounds with methods that have little to do with sound production mechanisms in the real physical world. Sampling and processed recordings synthesis take existing sound events and either reproduce them directly or process them further to create new sounds. Spectral modeling synthesis uses information of the properties of the sound as it is perceived by the listener. Physical modeling attempts to simulate the sound production mechanism of a real instrument.

This taxonomy can also be interpreted as being based on tasks generated by the user of the synthesis system. For evaluation purposes it is helpful to identify these sound synthesis problems. The tasks for which methods are best suited are:

1. Abstract algorithms
    - creation of new arbitrary sounds
    - computationally efficient moderate-quality synthesis of existing musical instruments
2. Sampling and processed recordings synthesis
    - reproduction of recorded sounds
    - merging and morphing of recorded sounds
    - using short sound bursts or recordings to produce new sound events
    - applications demanding high-sound quality
3. Spectral models

- simulation and analysis of existing sounds
  - copy synthesis (audio coding)
  - study of sound phenomena
  - pitch-shifting, time-scale modification

4. Physical models

- simulation and analysis of physical instruments
  - copy synthesis
  - study of the instrument physics
  - creation of physically unrealizable instruments of existing instrument families
- applications requiring control of high fidelity

Typically a sound synthesis method can be divided into analysis and synthesis procedures. These techniques are evaluated separately as they usually have different requirements. In many cases the analysis can be done off-line and accuracy can be gained by the cost of computation time. The synthesis part has to be typically done in real time and flexible ways to control the synthesis process have to be available.

An excellent discussion on the evaluation of sound synthesis methods is given by Jaffe (1995). Ten criteria proposed by Jaffe are discussed in next section with some additions. These criteria are utilized to create the evaluation scheme used in this document in the last three sections of the chapter. In the next chapter the evaluation scheme is applied to the synthesis methods presented in this document. The results are collected and tabulated to ease the comparison of the methods.

The ten criteria address the usability of the parameters; the quality, diversity and physicality of sounds produced; and implementation issues. One more criterion is included in the evaluation scheme of this document. It considers the suitability for parallel implementation of the synthesis method. These criteria are rated poor, fair, or good for each synthesis method.

6.1 Usability of the Parameters

Four aspects of parameters are discussed: the intuitivity, physicality, and the behavior of the parameters as well as the perceptibility of parameter changes. Ratings used to judge the parameters are presented in Table 6.1.

By intuitivity it is meant that a control parameter maps to a musical attribute or quality of timbre in an intuitive manner. With intuitive parameters the user is easily able to learn how to control the synthetic instrument. A significant parameter change should be perceivable for the parameter to be meaningful. Such parameters are called *strong* in contrast to *weak* parameters which cause barely audible changes (Jaffe, 1995). The trend is that the more parameters a synthesis system has, the

weaker they are (Jaffe, 1995). However, too strong parameters are hard to control as a little change on the parameter value has a drastic change on produced sound, no matter how intuitive the parameter is.

Physical parameters provide the player of a synthetic instrument with the behavior of a real-world instrument. They correspond to quantities the player of a real instrument is familiar with, such as, string length, bow or hammer velocity, mouth pressure in a wind instrument, etc. The behavior of a parameter is closely related to the "linearity" of the parameters. A change in a parameter should produce a proportional change in the sound produced.

The criteria presented in this section are tabulated in Table 6.1 with ratings that are used in the evaluation.

	poor	fair	good
Intuitivity	*	**	***
Perceptibility	*	**	***
Physicality	*	**	***
Behavior	*	**	***

Table 6.1: Criteria for the parameters of synthesis methods with ratings used in the evaluation scheme.

6.2 Quality and Diversity of Produced Sounds

In this section, criteria for the properties of produced sound are being discussed. These include the robustness of the sound's identity, the generality of the method, and availability of analysis methods. Ratings used to evaluate these criteria are presented in Table 6.2.

The robustness of the sound is determined by how well the identity of the sound is retained when modifications to the parameters are presented. This is to say that, e.g., a model of a clarinet should sound like a clarinet when played with different dynamics and playing styles, or even if the player decides to experiment with the parameter values. A general sound synthesis method is capable of producing arbitrary sound events of high quality. Every existing sound synthesis method has its short comings in generality and, indeed, every method does not even attempt to work at all with arbitrary sounds. This criterion is still useful as one would hope to have as general methods as possible for several synthesis problems.

For many sound synthesis methods an analysis method exists to derive synthesis parameters from recordings of sound signals. This makes the use of the synthesis method easier as it provides for default parameters that can then be modified to play the synthetic instrument. The analysis part is essential for many of the methods to be useful at all. In theory, copy synthesis or otherwise optimal parameters can be derived for most synthesis methods using different kinds of optimization methods. This is not typically desired, but instead, the analysis part often uses the knowledge

of the synthesis system to obtain reliable parameters.

In many cases the analysis can be done off-line and typically it will only have to be performed once for each instrument to be modeled. Thus, accuracy can be given much more weight on the cost of computing time and in this context computing efficiency will be discarded as a criteria for analysis methods. In this document, the analysis procedures of each synthesis system will be judged according to their accuracy, generality, and demands for special devices or instruments.

	poor	fair	good
Robustness of identity	*	**	***
Generality	*	**	***
Analysis methods	*	**	***

Table 6.2: Criteria for the quality and diversity of synthesis methods with ratings used in the evaluation scheme.

6.3 Implemention Issues

The implementation of a synthesis method has several important criteria to meet. Efficiency of the techniques is judged, latency and the control-rate are estimated. Suitability for parallel implementation is also addressed. Rating used to evaluate the criteria concerning implementation issues is presented in Table 6.3

Efficiency is further divided into three parts: computational demands, memory usage, and the load caused by control of the method. In many cases, the memory requirements can be compensated by increasing computational cost (Jaffe, 1995). Computational cost is rated *good* if one or several instances of the method can easily run in real time in an inexpensive processor, *fair* if only one instance of the method can run in real-time in a modern desk-top computer like PC or a workstation, and *poor* if a real-time implementation is not possible without dedicated hardware or a powerful supercomputer.

The control stream of the method affects both the expressivity and the computational demands. Typically, more control is possible with dense control streams than with sparse control streams (Jaffe, 1995). Processing of the control stream can be a lot more cost-deficient as it can involve I/O with external devices or files. In this context the control stream is judged by examining the amount of control made possible by the density of the stream.

In real-time synthesis systems there will always be latency present, as the system needs to be causal to be realizable. Latency is a problem especially with methods that employ block calculations, such as DFT. Also with other computationally costly synthesis methods it is sometimes advantageous to run tight loops for tens or maybe hundreds of output samples to speed up the calculation. That can be a cause for latency problems as well. In the ratings, *poor* means that the system will have latency of tens or hundreds of milliseconds or more, *fair* means that if there is not practically any extra overhead caused by, e.g., operating system the latency will

not be perceivable, and *good* indicates that the method is tolerant to some altering overhead.

	poor	fair	good
Computational cost	*	**	***
Memory usage	*	**	***
Control stream	*	**	***
Latency	*	**	***
Parallel processing	*	**	***

Table 6.3: Criteria for the implementation issues of synthesis methods with ratings used in the evaluation scheme.

Suitability for parallel implementation can be an important factor in certain situations. In this context it is assumed that fast communication between parallel processes is available. The system will be rated *good* on suitability for parallel processing if it can easily be divided into several processes so that communication between processes happens approximately at the sampling rate level of the system. Rating *fair* is given if the system can be divided into two processes communicating at sampling rate level or if it is advantageous to distribute the computation at higher communication level. The method will be judged *poor* if there is little or no advantage to parallelize the processing.

The synthesis methods are rated in Table 7.1 against the criteria presented in this section.

7. Evaluation of Several Sound Synthesis Methods

In this chapter the sound synthesis methods presented in this document are evaluated using the criteria discussed in the previous chapter. Ratings are tabulated for each method and they are also collected in Table 7.1 to enable comparison of the methods. It should be noted that the intention is not to decide which synthesis method is the best in general for that would be impossible. Rather, the evaluation should give some guidelines upon which a proper method can be chosen for a given sound synthesis problem.

For some methods there are criteria that we feel we cannot evaluate and those criteria are not rated.

7.1 Evaluation of Abstract Algorithms

7.1.1 FM synthesis

The FM synthesis parameters are strong offenders in the criteria of intuitivity, physicality, and behavior, as modulation parameters do not correspond to musical parameters or parameters of musical instruments at all, and because the method is highly nonlinear. Thus it is rated *poor* in all these categories. Notice, however, that the modulation index parameter I is directly related to the bandwidth of the produced signal. The method has strong parameters, i.e., parameters changes are easily audible. The rating in Perceptibility is *good*.

FM synthesis does not behave well when it is used to mimic a real instrument with varying dynamics and playing styles. The parameters of the method have to be changed very carefully in order not to lose the identity of the instrument. The method is rated *poor* for robustness of identity. Generality of FM synthesis is *good*. Analysis methods for FM have been proposed but the methods do not apply well for general cases, thus the rating *poor* for analysis methods. The interested reader is referred to a work by Delprat (1997) for methods of extracting frequency modulation laws by signal analysis.

The efficient implementations of FM have made it a popular method. It is very

cheap to implement, uses little memory, and the control stream is sparse. Minimal latency makes the methods attractive for real-time synthesis purposes. The method is rated *good* for all these criteria. FM synthesis is computationally so cheap that distributing one FM instrument is not feasible. Naturally, several FM instruments can be divided to run in several processors.

7.1.2 Waveshaping Synthesis

Waveshaping parameters are more intuitive (*fair*) than FM parameters especially when Chebyshev polynomials are used as a shaping function. Scaling of a weighting gain of a single Chebyshev polynomial only changes the gain of one harmonic. Thus the parameters are neither very perceptible nor physical (*poor*). Depending on the parameterization, the parameters typically behave *fairly* well.

Waveshaping is *fairly* general in that arbitrary harmonic spectra are easy to produce. By adding amplitude modulation after the waveshaping synthesis inharmonic spectra can be produced. Noisy signals cannot be generated easily. Spectral analysis can easily be applied to obtain the amplitudes of each harmonic. This data can be directly used for gains to the Chebyshev polynomials. The rating for analysis methods is thus *good*.

Just as FM synthesis, waveshaping can be implemented very efficiently and distribution of one instance is not feasible. The method is rated *good* for computing, memory, and control stream efficiency as well as for latency.

7.1.3 Karplus-Strong Synthesis

The few parameters of the Karplus-Strong synthesis are very intuitive, the changes are easily audible, and are well-behaved. Thus the rating for all these criteria is *good*. In the basic form, the method only has a parameter for the pitch and one for determining the type of tone, e.g., string or percussion. The physicality is thus rated *fair*.

KS synthesis is robust in that it will sound like a plucked string or a drum even when the parameters are changed, thus *good* for robustness of identity. In generality the method fails *poorly*. Analysis techniques for KS synthesis are not available but for related sound synthesis methods they exist (see Section 5.4).

Just like the other abstract algorithms, the KS is very attractive to implement in real time. The ratings for implementation issues are the same as with FM synthesis and waveshaping.

7.2 Evaluation of Sampling and Processed Recordings

7.2.1 Sampling

In sampling synthesis a recording of a sound signal is played back with possible looping in the steady-state part. Sampling is controlled just by note on/off and gain parameters. We have decided not to give ratings of these trivial parameters in order not to disturb the evaluation of other synthesis methods.

Sampling is very general (*good*) in that any sound can be recorded and sampled. The "identity" of the sound is retained with different playing styles and conditions but at the cost of naturalness. Robustness of identity is rated *fair*. Analysis methods for determining the looping breakpoints are available and usually give *good* results with harmonic sounds.

Sampling is computationally very efficient (*good*) but it uses lot of memory (*poor*). Control stream is sparse (*good*) and latency time small (*good*). Distribution of one sampling instrument is not feasible unless, e.g., a server is utilized as a memory storage. The rating is *fair*.

7.2.2 Multiple Wavetable Synthesis

Multiple wavetable synthesis methods can be parameterized in various ways and the result of synthesis is highly dependent of the signals stored in wavetables. Thus we decided not to give ratings on parameters of the method or the robustness of sounds identity.

The method is general (*good*) and analysis methods for some implementations are available (*fair*).

The method is *fairly* easily implemented computationally but it uses a lot of memory (*poor*). Control stream is not very costly computationally (*good*) and latency times can be kept small (*good*). Just like with sampling, a separate wavetable server can reduce the memory requirements of a single instance of multiple wavetable synthesis provided that fast connections are available. Suitability for distributed parallel processing is rated *fair*.

7.2.3 Granular synthesis

Granular synthesis is a set of techniques that vary quite a lot from each other in parameterization and implementation. Here a general evaluation of the concept is attempted. In the most primitive form the parameters of granular synthesis control the grains directly. The number of grains is typically very large and more elaborate means to control them must be utilized. The parameters are thus rated *poor* in intuitivity, perceptibility, and physicality. The system is linear and the behavior of the parameters is *good*.

Analysis methods for the pitch synchronous granular synthesis exist and they are also efficient (*good*). As the asynchronous method does not attempt to model or reproduce recorded sound signals no analysis tools are necessary. Granular synthesis methods are general (*good*), and with the PSGS the robustness of identity is retained well (*good*).

The implementation of the method is *fairly* efficient, and also the memory requirements are (*fair*) as the grains are short and it is typically assumed that the signals can be composed of few basic grains. The low-level control stream can become very dense especially with AGS (*poor*). The method does not pose latency problems (*good*) and the suitability for parallel processing is rated *fair*.

7.3 Evaluation of Spectral Models

7.3.1 Basic Additive Synthesis

Parameters of the basic additive synthesis control directly the sinusoidal oscillators. Ways to reduce the control data are available and some of them will be discussed with other spectral modeling methods. In this context only the basic additive synthesis is discussed. The parameters are *fairly* intuitive in that frequencies and amplitudes are easy to comprehend. The behavior of parameters is *good* as the method is linear. Perceptivity and physicality of the parameters is *poor*.

Additive synthesis can in theory synthesize arbitrary sounds if an unlimited number of oscillators is available. This soon becomes impractical as noisy signals cannot be modeled efficiently and thus the generality is rated *fair*. Analysis methods (*good*) based on, e.g., STFT are readily available as additive synthesis is used as a synthesis part of some of the other spectral modeling methods. Robustness of identity is not evaluated as the control of a synthetic instrument would need more elaborate control techniques.

A single sinusoidal oscillator can be implemented efficiently but in additive synthesis typically a large number of them is required. Computational cost is rated *fair*. The control data requires a large memory (*poor*), and the control stream is very dense (*poor*). Latency time is small as the oscillators run in parallel (*good*). Parallel implementation can become feasible when the number of oscillators grows large. In distributed implementation the oscillators and corresponding controls have to be grouped. The suitability for distributed parallel processing is rated *fair*.

7.3.2 FFT-based Phase Vocoder

The parameters of the FFT-based phase vocoder are directly related with the STFT analysis, such as the FFT length, window length and type, and the hop size parameter. While these are not intuitive directly they can be comprehended in the case of, e.g., time-scale modifications or pitch shifts. Intuitivity is rated *fair*. Parameters like the hop size are relatively strong whereas the window type might not have

any significant effect except in some specific situations. Perceptibility is rated *fair*. Physicality of the parameters is *poor* and behavior of the parameters is *good* if the changes are taken into account in the analysis stage as well.

The method retains the identity of the modeled instrument well especially if time-varying time-scale modification is applied (Serra, 1997a) (*good*). Generality is *good* and it is heavily based on the analysis stage (*good*).

The implementation of the method can be done relatively efficiently (*fair*) by using the FFT. The memory requirements are *fair* and the control stream is dense (*poor*) as the phase vocoder uses a transform of an original signal. Latency time is large (*poor*) because of the block-based FFT computation. Suitability for parallel processing is *fair* as the synthesis stage is mainly composed of IFFT. It was decided that all FFT-based methods are rated fair on suitability for parallel processing since although the computation of an FFT can be efficiently parallelized, it is typically computed in a single process.

7.3.3 McAulay-Quatieri Algorithm

The McAulay-Quatieri algorithm is based on a sinusoidal representation of signals. It uses additive synthesis as its synthesis part and can thus be interpreted as an analysis and data reduction method for the simple additive synthesis.

The control parameters of the MQ algorithm consist of amplitude, frequency, and phase trajectories. These trajectories are interpolated to obtain the additive synthesis parameters. As with the phase vocoder the intuitivity is rated *fair*, the perceptibility *fair*, physicality *poor*, and behavior *good*.

The algorithm works with arbitrary signals if the number of sinusoidal oscillators is increased accordingly but is infeasible for noisy signals (*fair*). Analysis method is *good* and the sound identity is retained *fairly* well with modifications.

The implementation is *fairly* efficient. The control stream (*fair*) is reduced by the cost of interpolation of trajectories. The trajectories take less memory than the envelopes of the additive synthesis and the memory usage is rated *fair*. Latency of the synthesis part is better (*fair*) than in the phase vocoder as it is related to the hop size instead of the FFT size. Suitability for parallel processing is rated *fair* as with additive synthesis.

7.3.4 Source-Filter Synthesis

The parameters of source-filter modeling include the choice of excitation signal, fundamental frequency if it exists, and the coefficients of the time-varying filter. These do not seem very intuitive but when the filter is parameterized properly, formants can be controlled easily. Intuitivity is thus rated *fair*. Perceptibility also is rated *fair* as changes in excitation signal are easily audible. The audibility of filter parameters depends again on parameterization. When source-filter synthesis is applied to simulate the human sound production system, the parameters are *fairly*

physical as the formants correspond to the shape of the vocal tract. In time-varying filtering, transition effects caused by updating the filter parameters are problematic and can easily become audible as disturbing artifacts. This causes the behavior of parameters to be rated *poor*.

Source-filter synthesis is general (*good*) in that it can, in theory, produce any sound. For example, linear prediction offers an analysis method to obtain the filter coefficients and inverse-filtering can be utilized to obtain an excitation signal (*good*). Robustness of identity (*fair*) depends on the parameterization but it can be easily lost if the filter parameters are not updated carefully.

Excitation and filtering are *fairly* efficient to implement. The method does not require a great deal of memory (*fair*). The control stream depends on the modeled signal, for steady state signal it is very sparse but for speech the filter coefficients have to be updated every few milliseconds (*fair*). Latency time of source-filter synthesis is small (*good*). Parallel distribution does not seem to pose any great advantages as a large part of the computational cost comes from filter coefficient updates (*poor*).

7.3.5 Spectral Modeling Synthesis

Spectral modeling synthesis uses additive synthesis to produce the deterministic (harmonic) component and source-filter synthesis to produce the stochastic (noisy) component of the synthetic signal. The parameters consist of amplitude and frequency trajectories of the deterministic component and spectral envelopes of the stochastic part. These parameters are rated *fair* in intuitivity. For modification of the analyzed signal to be meaningful, higher level controls have to be utilized to reduce the control data. Perceptibility and physicality are thus rated *poor*. The behavior of the parameters is *good*.

Robustness of identity is rated *good* as the composition of the signal provides means to edit the deterministic and stochastic part separately. This allows for better control of attack and steady state parts as with, e.g., the phase vocoder. Spectral modeling synthesis is judged to be *good* in generality and analysis method.

Computational cost is reasonable (*fair*). The method requires more memory than the MQ algorithm but it is still rated *fair* in memory usage. The control stream is *fairly* sparse as the additive source-filter parameters are interpolated between STFT frames. Latency time is in the order of that of the MQ algorithm (*fair*). Additive and source-filter synthesis can be divided into separate parallel processors and thus the suitability for distributed parallel processing is *fair*.

7.3.6 Transient Modeling synthesis

Transient modeling synthesis is an extension to the spectral modeling synthesis in that it allows for further processing of the residual signal as separate noise and transient signals. It is fair to say that TMS is more general than SMS and that it also involves more computation. The two methods are close enough for the ratings to be identical.

7.3.7 FFT^{-1}

FFT^{-1} is an additive synthesis method in the frequency domain that is also capable of producing noisy signals. The parameters consist of frequencies and amplitudes of partials and of bandwidths and amplitudes of noisy components. They are rated *fair* in intuitivity. The parameters are *poorly* perceptible as in a complex signal the number of signal components can be very large. They are not physical (*poor*) but they are linear and behave well (*good*).

The method is general (*good*) as it can produce harmonic, inharmonic, and noisy signals. STFT provides a *good* analysis method. As with additive synthesis, the robustness of sound identity is not evaluated for FFT^{-1}.

The implementation of the method is efficient (*good*). The memory usage is *fair* but the control stream can become very dense (*poor*) when the number of signal components increases. FFT^{-1} is a block-based method and suffers from latency problems (*poor*). As the method uses IFFT to in the synthesis stage, it is rated *fair* for suitability for parallel processing.

7.3.8 Formant Wave-Function Synthesis

The parameters of FOF synthesis govern the fundamental frequency and the structure of the formants of synthesized sound signals. The parameters can be judged *fairly* intuitive and physical especially when the method is used for simulation of the human sound production system. Perceptibility is *good* as there are typically only a few formants present in speech or singing voice signals. The parameters are well-behaved (*good*).

The method is *fairly* general as it can produce high-quality harmonic sound signals of singing and musical instruments. Linear prediction provides for an analysis method (*good*), and the sounds produced retain their identity well (*good*) as is proven by sound examples (Bennett and Rodet, 1989).

The method can be implemented efficiently when different FOFs are stored in wavetables (*good*). This increases the amount of required memory (*fair*). The control rate is *fairly* sparse and the latency time is small (*good*). Parallel processing of a single FOF instrument does not seem feasible as the excitation signal is shared by all of the FOF generators (*poor*).

7.3.9 VOSIM

The parameters of the VOSIM model are not very well related to either the sound production mechanism being modeled or the sound itself. Thus physicality and intuitivity of the parameters are both rated *poor*. The perceptibility is rated *fair* as some of the parameters are strong and some weak. The behavior is also rated *fair* because, although the method is linear, the effect of each parameter to the sound produced may not be very well-behaved.

The method is *fairly* general as it has been used to model the human sound production mechanism and some musical instruments. An efficient analysis method was not found in the literature (*poor*). The parameterization suggests that the method is not robust with parameter modification (*poor*). VOSIM can be implemented efficiently and it only requires a small amount of memory (both rated *good*). The control stream is sparse (*good*) and latency time can be kept small (*good*). There is little advantage in parallel implementation of the system (*poor*).

7.4 Evaluation of Physical Models

7.4.1 Finite Difference Methods

The parameters of finite difference methods correspond directly to the physical parameters of the modeled sound production system. Thus they are very physical and intuitive, and can also be rated *good* for perceptibility and behavior as the vibratory motion is assumed linear.

Although the method can be applied in theory to arbitrary sound production systems, a new implementation is typically required as the instrument under study is changed. Thus, FD methods are rated *fair* in generality. A tuned model of an instrument behaves very much like the original and retains the identity well (*good*). Analysis methods are available but although the results can be very good, they often include specialized measurement instruments and require a great deal of time and effort (*fair*).

FD methods are computationally very inefficient (*poor*), and they also need a *fair* amount of memory. The control stream depends on the excitation but is very sparse with plucked or struck strings and with mallet instruments (*good*). The method does not pose a problem with latency times if sufficient computational capacity is available (*good*). The method is well suited (*good*) for distributed parallel processing as significant improvements can be achieved by dividing the system into several substructures running as different processes.

7.4.2 Modal Synthesis

Modal synthesis parameters consist of the modal data of the modeled structure and the excitation or driving data. The parameters are not very intuitive (*fair*), and a change in a single mode can be hard to perceive if the number of modes is large (*poor*). They correspond directly to physical structures and the physicality and the behavior are rated *good*.

Analysis methods for the system are available and they produce reliable and accurate results. However, they suffer from being very complicated and expensive (*fair*). The system is general as any vibrating object can be formulated as its modal data. Arbitrary sounds related to no physical objects are not easily produced by the mechanism. Generality is rated *fair*. The modeled structure retains its identity

very well (*good*) as it is typically controlled by the excitation signal.

The method is implemented as a set of parallel second-order resonators that are computationally efficient. The number of resonators can grow very large and thus the computational efficiency is rated *fair*. The modal data requires a large amount of memory (*poor*). The excitation signal defines the control stream for static structures and it can be rated sparse (*good*). The substructures can be efficiently distributed and processed in parallel (*good*).

7.4.3 CORDIS

A clear description of the CORDIS system parameters was not found in the literature. For this reason the parameters of CORDIS are not evaluated.

As the method uses a physics-based description of the system that vibrates, it retains the identity of sound well (*good*) with different meaningful excitation signals. No analysis system was described in the literature (*poor*). The generality of the system is *fair* as it can, in theory, model arbitrary vibrating objects. The method does not provide an easy way to create arbitrary sounds.

The method is judged to be computationally *fair* as although the basic elements can be computed efficiently, a large number of them is needed. This accounts also for rating *fair* for memory requirements. The control rate depends on the parameterization and is not evaluated here. The latency time of CORDIS is small (*good*). It appears that CORDIS may be well suited for parallel processing (*good*) (Rocchesso, 1998).

7.4.4 Digital Waveguide Synthesis

Digital waveguide synthesis parameters are intuitive and physical as they correspond well to physical structures of the instrument and the way it is being played (both rated *good*). The parameter changes are typically audible (*good*) and they behave well with linear models. As some of the waveguide models have nonlinear couplings, the behavior is rated *fair*.

The identity of the instrument is retained very well (*good*). The method can be used to simulate instruments with one-dimensional vibrating objects, such as string and wind instruments, and it is thus rated *fair* in generality. Automated analysis methods for linear models are efficient but they are not available for nonlinear models (*fair*).

A digital waveguide can be implemented very efficiently but typically the model also incorporates other structures that increase the computational requirements. Computational efficiency is rated *fair*. Digital waveguide models require little memory other than the excitation signal. A high-quality plucked string tone of several seconds can be produced with only several thousands words of memory (*good*). The control stream depends on the instrument being modeled and is here rated *fair*. The method does not pose latency problems and especially models with several vibrating

structures can be efficiently divided into substructures that are computed in parallel (both rated *good*).

7.4.5 Waveguide Meshes

The parameters of the waveguide meshes are *fairly* intuitive as they correspond to the excitation and the properties of the 2D or 3D vibrating system. Parameters are physical and perceptible and they behave well as the mesh is linear (rated *good* for all those criteria).

Analysis methods were not found in the literature (*poor*). The method is *fairly* general as it can be applied to simulation of 2D and 3D objects. The robustness of identity is not evaluated.

The method is computationally expensive and requires a large amount of memory (both rated *poor*). The control stream is *fairly* sparse as it consists only of the excitation information. The method itself does not pose latency problems (*good*), although real-time implementations of more complex structures cannot be achieved without expensive supercomputers. One of the main advantages of the model is that it can be divided into arbitrary substructures that are computed in parallel (*good*).

7.4.6 Commuted Waveguide Synthesis

Commuted digital waveguides have been used to produce high-quality synthesis of instruments that can be described as having linear or linearizable coupling of excitation to the vibrating structure. The parameters are very intuitive, perceptible, physical and well-behaved, and commuted waveguide synthesis is rated *good* for those criteria.

The method is very *good* in retaining the identity of the modeled instrument. For good synthesis results, parameters need to be derived by analysis of existing instruments. The analysis methods employ STFT and produce *good* results. The method is *fairly* general as a number of percussive, plucked, or struck string instruments can be modeled with commuted synthesis.

The implementation issues of commuted waveguide synthesis are very close to digital waveguide synthesis. The ratings are the same and they will be repeated for convenience. Computational efficiency and control stream are rated *fair*, and memory usage, latency, and suitability for parallel processing *good*.

7.5 Results of Evaluation

The evaluation results discussed in the previous sections are tabulated in Table 7.1. It can be observed that the abstract algorithms and sampling techniques are strongest int the implementation category. Spectral models are general, robust, and analysis methods are available. They are strongest in the sound category. Physical modeling employs very intuitive parameterization, and it is strongest in the parameter category.

	Parameters				Sound			Implementation				
	Int	Perc	Phys	Behav	Robust	Gen	Anal	Comp	Mem	Contr	Lat	Par
Abstract												
FM	*	***	*	*	*	***	*	***	***	***	***	–
Waveshaping	**	*	*	**	–	**	***	***	***	***	***	–
KS	***	***	**	***	***	*	*	***	***	***	***	–
Sampling												
Sampling	–	–	–	–	**	***	***	***	*	***	***	**
Multiple WT	–	–	–	–	–	***	**	**	*	***	***	**
Granular	*	*	*	***	***	***	***	**	**	*	***	**
Spectral												
Additive	**	*	*	***	–	**	***	**	*	*	***	**
Phase Vocoder	**	**	*	**	***	***	***	**	**	*	*	*
MQ	**	**	*	***	**	**	***	**	**	**	**	**
Source-filter	**	**	**	*	**	***	***	**	**	**	***	*
SMS	**	*	*	***	***	***	***	**	**	**	**	**
TMS	**	*	*	***	***	***	***	**	**	**	**	**
FFT−1	**	*	*	***	–	**	***	***	**	*	*	*
CHANT	**	***	**	***	***	**	***	***	**	**	***	*
VOSIM	*	**	*	**	*	**	*	***	***	***	***	*
Physical												
Modal	**	*	***	***	***	**	**	**	*	***	***	***
CORDIS	–	–	–	–	***	**	*	**	**	-	***	–
FD methods	***	***	***	***	***	**	**	*	**	***	***	***
Waveguide	***	***	***	***	***	**	**	**	***	**	***	***
WG Meshes	**	***	***	***	–	**	*	*	*	**	***	***
Commuted WG	***	***	***	***	***	**	***	**	***	**	***	***

Table 7.1: Tabulated evaluation of the sound synthesis methods presented in this document.

8. Summary and Conclusions

In this document, several modern sound synthesis methods have been discussed. The methods were divided into four groups according to a taxonomy proposed by Smith (1991). Representative examples in each group were chosen, and a describtion of those methods was given. The interested reader was referred to the literature for more information on the methods.

Three methods based on abstract algorithms were chosen: FM synthesis, waveshaping synthesis, and the Karplus-Strong algorithm. Also, three methods utilizing recordings of sounds were discussed. These are sampling, multiple wavetable synthesis, and granular synthesis.

In the spectral modeling category three traditional linear sound synthesis methods, namely, additive synthesis, the phase vocoder, and source-filter synthesis, were first discussed. Second, McAulay-Quatieri algorithm, Spectral Modeling Synthesis, Transient Modeling Synthesis and the inverse-FFT based additive synthesis method (FFT^{-1} synthesis) were described. Finally, two methods for modeling the human voice were shortly addressed. These methods are the CHANT and the VOSIM.

Three physical modeling methods that use numerical acoustics were investigated. First, models using finite difference methods were presented. Applications to string instruments as well as to mallet percussion instruments were presented. Second, modal synthesis was discussed. Third, CORDIS, a system of modeling vibrating objects by mass-spring networks, was described.

Continuing in the physical modeling category, digital waveguides were discussed. Waveguide meshes, which are 2-D and 3-D models, were also presented. Extensions and physical-modeling interpretation of the Karplus-Strong algorithm was discussed, and single delay loop (SDL) models were described. Finally, a case study of modeling the acoustic guitar using commuted waveguide synthesis was presented.

After the methods in the four categories were discussed, evaluation criteria based on those proposed by (Jaffe, 1995) were described. One additional criterion was added addressing the suitability of a method for parallel processing. Each method was evaluated with a discussion concerning each evaluation criterion. The criteria were rated with qualitative measure for each method. Finally, the ratings were tabulated in a comparable form. It was observed that abstract algorithms and sampling techniques are strongest in the implementation category. Spectral models are gen-

eral, robust, and analysis methods are available. They are strongest in the sound category. Physical modeling algorithms employ very intuitive parameterization, and are strongest in the parameter category.

Bibliography

Adrien, J. M. 1989. Dynamic modeling of vibrating structures for sound synthesis, modal synthesis, *Proceedings of the AES 7th International Conference*, Audio Engineering Society, Toronto, Canada, pp. 291–300.

Adrien, J.-M. 1991. The missing link: modal synthesis, *in:* G. D. Poli, A. Piccialli and C. Roads (eds), *Representations of Musical Signals*, The MIT Press, Cambridge, Massachusetts, USA, pp. 269–297.

Arfib, D. 1979. Digital synthesis of complex spectra by means of multiplication of nonlinear distorted sine waves, *Journal of the Audio Engineering Society* **27**(10): 757–768.

Bate, J. A. 1990. The effect of modulator phase on timbres in FM synthesis, *Computer Music Journal* **14**(3): 38–45.

Bennett, G. and Rodet, X. 1989. Synthesis of the singing voice, *in:* M. V. Mathews and J. R. Pierce (eds), *Current Directions in Computer Music Research*, The MIT Press, Cambridge, Massachusetts, chapter 4, pp. 19–44.

Borin, G. and Giovanni, D. P. 1996. A hysteretic hammer-string interaction model for physical model synthesis, *Proceedings of the Nordic Acouctical Meeting*, Helsinki, Finland, pp. 399–406.

Borin, G., De Poli, G. and Rocchesso, D. 1997a. Elimination of delay-free loops in discrete-time models of nonlinear acoustic systems, *Proceedings of the IEEE Workshop of Applications of Signal Processing to Audio and Acoustics*, New Paltz, New York.

Borin, G., De Poli, G. and Sarti, A. 1997b. Musical signal synthesis, *in:* C. Roads, S. T. Pope, A. Piccialli and G. De Poli (eds), *Musical Signal Processing*, Swets & Zeitlinger, Lisse, the Netherlands, chapter 1, pp. 5–30.

Bristow-Johnson, R. 1996. Wavetable synthesis 101, a fundamental perspective, *Proceedings of the 101st AES convention in Los Angeles, California.*

Cadoz, C., Luciani, A. and Florens, J. 1983. Responsive input devices and sound synthesis by simulation of instrumental mechanisms: the CORDIS system, *Computer Music Journal* **8**(3): 60–73.

Cavaliere, S. and Piccialli, A. 1997. Granular synthesis of musical signals, *in:* C. Roads, S. T. Pope, A. Piccialli and G. De Poli (eds), *Musical Signal Processing*, Swets & Zeitlinger, Lisse, the Netherlands, chapter 5, pp. 155–186.

Chaigne, A. 1991. Viscoelastic properties of nylon guitar strings, *Catgut Acoustical Society Journal* **1**(7): 21–17.

Chaigne, A. 1992. On the use of finite differences for musical synthesis. Application to plucked stringed instruments, *Journal d'Acoustique* **5**(2): 181–211.

Chaigne, A. and Askenfelt, A. 1994a. Numerical simulations of piano strings. I. A physical model for a struck string using finite difference methods, *Journal of the Acoustical Society of America* **95**(2): 1112–1118.

Chaigne, A. and Askenfelt, A. 1994b. Numerical simulations of piano strings. II. Comparisons with measurements and systematic exploration of some hammer-string parameters, *Journal of the Acoustical Society of America* **95**(3): 1631–1640.

Chaigne, A. and Doutaut, V. 1997. Numerical simulations of xylophones. I. Time-domain modeling of the vibrating bars, *Journal of the Acoustical Society of America* **101**(1): 539–557.

Chaigne, A., Askenfelt, A. and Jansson, E. V. 1990. Temporal synthesis of string instrument tones, *Quarterly Progress and Status Report*, number 4, Speech Transmission Laboratory, Royal Institute of Technology (KTH), Stockholm, Sweden, pp. 81–100.

Chowning, J. M. 1973. The synthesis of comples audio spectra by means of frequency modulation, *Journal of the Audio Engineering Society* **21**(7): 526–534. Reprinted in C. Roads and J. Strawn, eds. 1985. *Foundations of Computer Music.* Cambridge, Massachusetts: The MIT Press. pp. 6-29.

Cook, P. R. 1991. TBone: and interactive waveguide brass instrument synthesis workbench for the NeXT machine, *Proceedings of the International Computer Music Conference*, Montreal, Canada, pp. 297–299.

Cook, P. R. 1992. A meta-wind-instrument physical model, and a meta-controller for real time performance control, *Proceedings of the International Computer Music Conference*, International Computer Music Association, San Francisco, CA., pp. 273–276.

Cook, P. R. 1993. SPASM, a real-time vocal tract physical model controller; and singer, the companion software synthesis system, *Computer Music Journal* **17**(1): 30–44.

De Poli, G. 1983. A tutorial on digital sound synthesis techniques, *Computer Music Journal* **7**(2): 76–87. Also published in Roads C. (ed). 1989. *The Music Machine*, pp. 429–447. The MIT Press. Cambridge, Massachusetts, USA.

De Poli, G. and Piccialli, A. 1991. Pitch-synchronous granular synthesis, *in:* G. D. Poli, A. Piccialli and C. Roads (eds), *Representations of Musical Signals*, The MIT Press, Cambridge, Massachusetts, USA, pp. 391–412.

Delprat, N. 1997. Global frequency modulation laws extraction from the Gabor transform of a signal: a first study of the interacting components case, *IEEE Transactions on Speech and Audio Processing* **5**(1): 64–71.

Dietz, P. H. and Amir, N. 1995. Synthesis of trumpet tones by physical modeling, *Proceedings of the International Symposium on Musical Acoustics*, pp. 472–477.

Dolson, M. 1986. The phase vocoder: a tutorial, *Computer Music Journal* **10**(4): 14–27.

Dudley, H. 1939. The vocoder, *Bell Laboratories Record* **17**: 122–126.

Eckel, G., Iovino, F. and Caussé, R. 1995. Sound synthesis by physical modelling with Modalys, *Proceedings of the International Symposium on Musical Acoustics*, Dourdan, France, pp. 479–482.

Evangelista, G. 1993. Pitch-synchronous wavelet representation of speech and music signals, *IEEE Transactions on Signal Processing* **41**(12): 3312–3330.

Evangelista, G. 1994. Comb and multiplexed wavelet transforms and their applications to signal processing, *IEEE Transactions on Signal Processing* **42**(2): 292–303.

Evangelista, G. 1997. Wavelet representations of musical signals, *in:* C. Roads, S. T. Pope, A. Piccialli and G. De Poli (eds), *Musical Signal Processing*, Swets & Zeitlinger, Lisse, the Netherlands, chapter 4, pp. 127–153.

Fitz, K. and Haken, L. 1996. Sinusoidal modeling and manipulation using Lemur, *Computer Music Journal* **20**(4): 44–59.

Flanagan, J. L. and Golden, R. M. 1966. Phase Vocoder, *The Bell System Technical Journal* **45**: 1493–1509.

Fletcher, N. H. and Rossing, T. D. 1991. *The Physics of Musical Instruments*, Springer-Verlag, New York, USA, p. 620.

Florens, J.-L. and Cadoz, C. 1991. The physical model: modeling and simulating the instrumental universe, *in:* G. D. Poli, A. Piccialli and C. Roads (eds), *Representations of Musical Signals*, The MIT Press, Cambridge, Massachusetts, USA, pp. 227–268.

Fontana, F. and Rocchesso, D. 1995. A new formulation of the 2D-waveguide mesh fo percussion instruments, *Proceedings of the XI Colloquium on Musical Informatics*, Bologna, Italy, pp. 27–30.

Goodwin, M. and Gogol, A. 1995. Overlap-add synthesis of nonstationary sinusoids, *Proceedings of the International Computer Music Conference*, Banff, Canada, pp. 355–356.

Goodwin, M. and Rodet, X. 1994. Efficient Fourier synthesis of nonstationary sinusoids, *Proceedings of the International Computer Music Conference*, Aarhus, Denmark, pp. 333–334.

Goodwin, M. and Vetterli, M. 1996. Time-frequency signal models for music analysis, transformation, and synthesis, *Proceedings of the 3rd IEEE Symposium on Time-Frequency and Time Scale Analysis*, Paris, France.

Gordon, J. W. and Strawn, J. 1985. An introduction to the phase vocoder, *in:* J. Strawn (ed.), *Digital Audio Signal Processing: An Anthology*, William Kauffmann, Inc., chapter 5, pp. 221–270.

Harris, F. J. 1978. On the use of windows for harmonic analysis with the discrete Fourier transform, *Proceedings of the IEEE* **66**(1): 51–83.

Hiller, L. and Ruiz, P. 1971a. Synthesizing musical sounds by solving the wave equation for vibrating objects: part 1, *Journal of the Audio Engineering Society* **19**(6): 462–470.

Hiller, L. and Ruiz, P. 1971b. Synthesizing musical sounds by solving the wave equation for vibrating objects: part 2, *Journal of the Audio Engineering Society* **19**(7): 542–550.

Hirschman, S. E. 1991. Digital Waveguide Modeling and Simulation of Reed Woodwind Instruments, *Technical Report STAN-M-72*, Stanford University, Dept. of Music, Stanford, California.

Holm, F. 1992. Understanding FM implementations: a call for common standards, *Computer Music Journal* **16**(1): 34–42.

Horner, A., Beauchamp, J. and Haken, L. 1993. Methods for multiple wavetable synthesis of musical instrument tones, *Journal of the Audio Engineering Society* **41**(5): 336–356.

Huopaniemi, J., Karjalainen, M., Välimäki, V. and Huotilainen, T. 1994. Virtual instruments in virtual rooms—a real-time binaural room simulation environment for physical modeling of musical instruments, *Proceedings of the International Computer Music Conference*, Aarhus, Denmark, pp. 455–462.

Jaffe, D. A. 1995. Ten criteria for evaluating synthesis techniques, *Computer Music Journal* **19**(1): 76–87.

Jaffe, D. A. and Smith, J. O. 1983. Extensions of the Karplus-Strong plucked-string algorithm, *Computer Music Journal* **7**(2): 56–69. Also published in Roads C. (ed). 1989. *The Music Machine*, pp. 481–494. The MIT Press. Cambridge, Massachusetts, USA.

Kaegi, W. and Tempelaars, S. 1978. VOSIM—a new sound synthesis system, *Journal of the Audio Engineering Society* **26**(6): 418–426.

Karjalainen, M. and Laine, U. K. 1991. A model for real-time sound synthesis of guitar on a floating-point signal processor, *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 5, Toronto, Canada, pp. 3653–3656.

Karjalainen, M. and Smith, J. O. 1996. Body modeling techniques for string instrument synthesis, *Proceedings of the International Computer Music Conference*, Hong Kong, pp. 232–239.

Karjalainen, M., Huopaniemi, J. and Välimäki, V. 1995. Direction-dependent physical modeling of musical instruments, *Proceedings of the International Congress on Acoustics*, Vol. 3, Trondheim, Norway, pp. 451–454.

Karjalainen, M., Laine, U. K. and Välimäki, V. 1991. Aspects in modeling and real-time synthesis of the acoustic guitar, *Proceedings of the IEEE Workshop of Applications of Signal Processing to Audio and Acoustics*, New Paltz, New York, USA.

Karjalainen, M., Välimäki, V. and Jánosy, Z. 1993. Towards high-quality sound synthesis of the guitar and string instruments, *Proceedings of the International Computer Music Conference*, Tokyo, Japan, pp. 56–63.

Karjalainen, M., Välimäki, V. and Tolonen, T. 1998. Plucked string models—from Karplus-Strong algorithm to digital waveguides and beyond, Accepted for publication in *Computer Music Journal*.

Karplus, K. and Strong, A. 1983. Digital synthesis of plucked-string and drum timbres, *Computer Music Journal* **7**(2): 43–55. Also published in Roads C. (ed). 1989. *The Music Machine*. pp.467-479. The MIT Press. Cambridge, Massachusetts.

Kurz, M. and Feiten, B. 1996. Physical modelling of a stiff string by numerical integration, *Proceedings of the International Computer Music Conference*, Hong Kong, pp. 361–364.

Laakso, T. I., Välimäki, V., Karjalainen, M. and Laine, U. K. 1996. Splitting the unit delay—tools for fractional delay filter design, *IEEE Signal Processing Magazine* **13**(1): 30–60.

Lang, M. and Laakso, T. I. 1994. Simple and robust method for the design of allpass filters using least-squares phase error criterion, *IEEE Transactions on Circuits and Systems–II: Analog and Digital Signal Processingx* **41**(1): 40–48.

Laroche, J. and Dolson, M. 1997. About this phasiness business, *Proceedings of the International Computer Music Conference*, Thessaloniki, Greece, pp. 55–58.

Laroche, J. and Jot, J.-M. 1992. Analysis/synthesis of quasi-harmonic sound by use of the Karplus-Strong algorithm, *Proceedings of the 2nd French Congress on Acoustics*, Archachon, France.

Le Brun, M. 1979. Digital waveshaping synthesis, *Journal of the Audio Engineering Society* **27**(4): 250–266.

Makhoul, J. 1975. Linear prediction: a tutorial review, *Proceedings of the IEEE* **63**: 561–580.

McAulay, R. J. and Quatieri, T. F. 1986. Speech analysis/synthesis based on a sinusoidal representation, *IEEE Transactions on Acoustics, Speech, and Signal Processing* **34**(6): 744–754.

Moore, F. R. 1990. *Elements of Computer Music*, Prentice Hall, Englewood Cliffs, New Jersey.

Moorer, J. A. 1978. The use of the phase vocoder in computer music applications, *Journal of the Audio Engineering Society* **26**(1/2): 42–45.

Moorer, J. A. 1979. The use of linear prediction of speech in computer music applications, *Journal of the Audio Engineering Society* **27**(3): 134–140.

Moorer, J. A. 1985. Signal processing aspects of computer music: a survey, *in:* J. Strawn (ed.), *Digital Audio Signal Processing: An Anthology*, William Kauffmann, Inc., chapter 5, pp. 149–220.

Morrison, J. and Adrien, J. 1993. MOSAIC: a framework for modal synthesis, *Computer Music Journal* **17**(1): 45–56.

Morse, P. M. and Ingard, U. K. 1968. *Theoretical Acoustics*, Princeton University Press, Princeton, New Jersey, USA.

Msallam, R., Dequidt, S., Tassart, S. and Caussé, R. 1997. Physical model of the trombone including nonlinear propagation effects, *Proceedings of the Institute of Acoustics*, Vol. 19, pp. 245–250. Presented at the International Symposium on Musical Acoustics, Edinburgh, UK.

Nuttall, A. H. 1981. Some windows with very good sidelobe behavior, *IEEE Transactions on Acoustics, Speech, and Signal Processing* **29**(1): 84–91.

Oppenheim, A. V., Willsky, A. S. and Young, I. T. 1983. *Signals and Systems*, Prentice-Hall, New Jersey, USA, p. 796.

Paladin, A. and Rocchesso, D. 1992. A dispersive resonator in real-time on MARS workstation, *Proceedings of the International Computer Music Conference*, San Jose, California, USA, pp. 146–149.

Portnoff, M. R. 1976. Implementation of the digital phase vocoder using the fast Fourier transform, *IEEE Transactions on Acoustics, Speech, and Signal Processing* **24**(3): 243–248.

Rank, E. and Kubin, G. 1997. A waveguide model for slapbass synthesis, *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 1, Munich, Germany, pp. 443–446.

Risset, J.-C. 1985. Computer music experiments 1964- . . . , *Computer Music Journal* **9**(1): 67–74. Also published in Roads C. (ed). 1989. *The Music Machine.* pp. 67–74. The MIT Press. Cambridge, Massachusetts, USA.

Roads, C. 1991. Asynchronous granular synthesis, *in:* G. D. Poli, A. Piccialli and C. Roads (eds), *Representations of Musical Signals*, The MIT Press, Cambridge, Massachusetts, USA, pp. 143–185.

Roads, C. 1995. *The Computer Music Tutorial*, The MIT Press, Cambridge, Massachusetts, USA, p. 1234.

Rocchesso, D. 1998. Personal communication.

Rocchesso, D. and Scalcon, F. 1996. Accurate dispersion simulation for piano strings, *Proceedings of the Nordic Acouctical Meeting*, Helsinki, Finland, pp. 407–414.

Rocchesso, D. and Turra, F. 1993. A generalized excitation for real-time sound synthesis by physical models, *Proceedings of the Stockholm Music Acoustics Conference*, Stockholm, Sweden, pp. 584–588.

Rodet, X. 1980. Time-domain formant-wave-function synthesis, *Computer Music Journal* **8**(3): 9–14.

Rodet, X. and Depalle, P. 1992a. A new additive synthesis method using inverse Fourier transform and spectral envelopes, *in:* A. Strange (ed.), *Proceedings of the International Computer Music Conference*, pp. 410–411.

Rodet, X. and Depalle, P. 1992b. Spectral envelopes and inverse FFT synthesis, *Proceedings of the 93rd AES convention*, San Francisco, California.

Rodet, X., Potard, Y. and Barrière, J.-B. 1984. The CHANT project: from synthesis of the singing voice to synthesis in general, *Computer Music Journal* **8**(3): 15–31.

Savioja, L. and Välimäki, V. 1996. The bilinearly deinterpolated waveguide mesh, *Proceedings of the 1996 IEEE Nordic Signal Processing Symposium*, Espoo, Finland, pp. 443–446.

Savioja, L. and Välimäki, V. 1997. Improved discrete-time modeling of multi-dimensional wave propagation using the interpolated digital waveguide mesh, *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 1, Munich, Germany.

Serra, M.-H. 1997a. Introducing the phase vocoder, *in:* C. Roads, S. T. Pope, A. Piccialli and G. De Poli (eds), *Musical Signal Processing*, Swets & Zeitlinger, Lisse, the Netherlands, chapter 2, pp. 31–90.

Serra, X. 1989. *A System for Sound Analysis/Transformation/Synthesis Based on a Deterministic plus Stochastic Decomposition*, PhD thesis, Stanford University, California, USA, p. 151.

Serra, X. 1997b. Musical sound modeling with sinusoids plus noise, *in:* C. Roads, S. T. Pope, A. Piccialli and G. De Poli (eds), *Musical Signal Processing*, Swets & Zeitlinger, Lisse, the Netherlands, chapter 3, pp. 91–122.

Serra, X. and Smith, J. O. 1990. Spectral modeling synthesis: a sound analysis/synthesis system based on a deterministic plus stochastic decomposition, *Computer Music Journal* **14**(4): 12–24.

Smith, J. O. 1983. *Techniques for Digital Filter Design and System Identification with Application to the Violin*, PhD thesis, Stanford University, California, USA, p. 260.

Smith, J. O. 1986. Efficient simulation of the reed-bore and bow-string mechanisms, *Proceedings of the International Computer Music Conference*, The Hague, the Netherlands, pp. 275–280.

Smith, J. O. 1987. Music applications of digital waveguides, *Technical Report STAN-M-39*, CCRMA, Dept. of Music, Stanford University, California, USA, p. 181.

Smith, J. O. 1991. Viewpoints on the history of digital synthesis, *Proceedings of the International Computer Music Conference*, Montreal, Canada, pp. 1–10.

Smith, J. O. 1992. Physical modeling using digital waveguides, *Computer Music Journal* **16**(4): 74–91.

Smith, J. O. 1993. Efficient synthesis of stringed musical instruments, *Proceedings of the International Computer Music Conference*, Tokyo, Japan, pp. 64–71.

Smith, J. O. 1995. Introduction to digital waveguide modeling of musical instruments, Unpublished manuscript.

Smith, J. O. 1996. Physical modeling synthesis update, *Computer Music Journal* **20**(2): 44–56.

Smith, J. O. 1997. Acoustic modeling using digital waveguides, *in:* C. Roads, S. T. Pope, A. Piccialli and G. De Poli (eds), *Musical Signal Processing*, Swets & Zeitlinger, Lisse, the Netherlands, chapter 7, pp. 221–264.

Smith, J. O. and Serra, X. 1987. PARSHL: an analysis/synthesis program for non-harmonic sounds based on a sinusoidal representation, *Proceedings of the International Computer Music Conference*, Urbana-Champaign, Illinois, USA, pp. 290–297.

Smith, J. O. and Van Duyne, S. A. 1995. Commuted piano synthesis, *Proceedings of the International Computer Music Conference*, Banff, Canada, pp. 335–342.

Stilson, T. and Smith, J. 1996. Alias-free digital synthesis of classical analog waveforms, *Proceedings of the International Computer Music Conference*, Hong Kong, pp. 332–335.

Strawn, J. 1980. Approximation and syntactic analysis of amplitude and frequency functions for digital sound synthesis, *Computer Music Journal* **4**(3): 3–24.

Sullivan, C. S. 1990. Extending the Karplus-Strong algorithm to synthesize electric guitar timbres with distortion and feedback, *Computer Music Journal* **14**(3): 26–37.

Tolonen, T. 1998. *Model-based Analysis and Resynthesis of Acoustic Guitar Tones*, Master's thesis, Helsinki University of Technology, Espoo, Finland, p. 102. Report 46, Laboratory of Acoustics and Audio Signal Processing.

Tolonen, T. and Välimäki, V. 1997. Automated parameter extraction for plucked string synthesis, *Proceedings of the Institute of Acoustics*, Vol. 19, pp. 245–250. Presented at the International Symposium on Musical Acoustics, Edinburgh, UK.

Tomisawa, N. 1981. Tone production method for an electronic musical instrument, U.S. Patent 4,249,447.

Truax, B. 1977. Organizational techniques for c:m ratios in frequency modulation, *Computer Music Journal* **1**(4): 39–45. Reprinted in C. Roads and J. Strawn, eds. 1985. *Foundations of Computer Music*. Cambridge, Massachusetts: The MIT Press. pp. 68-82.

Van Duyne, S. A. and Smith, J. O. 1993a. The 2-D digital waveguide, *Proceedings of the IEEE Workshop of Applications of Signal Processing to Audio and Acoustics*, New Paltz, New York, USA.

Van Duyne, S. A. and Smith, J. O. 1993b. Physical modeling with the 2-D digital waveguide mesh, *Proceedings of the International Computer Music Conference*, pp. 40–47.

Van Duyne, S. A. and Smith, J. O. 1994. A simplified approach to modeling dispersion caused by stiffness in strings and plates, *Proceedings of the International Computer Music Conference*, Aarhus, Denmark, pp. 407–410.

Van Duyne, S. A. and Smith, J. O. 1995a. Developments for the commuted piano, *Proceedings of the International Computer Music Conference*, Banff, Canada, pp. 319–326.

Van Duyne, S. A. and Smith, J. O. 1995b. The tetrahedral digital waveguide mesh, *Proceedings of the IEEE Workshop of Applications of Signal Processing to Audio and Acoustics*, New Paltz, New York.

Van Duyne, S. A. and Smith, J. O. 1996. The 3D tetrahedral digital waveguide mesh with musical applications, *Proceedings of the International Computer Music Conference*, International Computer Music Association, Hong Kong, pp. 9–16.

Vergez, C. and Rodet, X. 1997. Comparison of real trumpet playing, latex model of lips and computer model, *Proceedings of the International Computer Music Conference*, Thessaloniki, Greece, pp. 180–187.

Verma, T. S., Levine, S. N. and Meng, T. H. Y. 1997. Transient modeling synthesis: a flexible analysis/synthesis tool for transient signals, *Proceedings of the International Computer Music Conference*, Thessaloniki, Greece, pp. 164–167.

Välimäki, V. 1995. *Discrete-Time Modeling of Acoustic Tubes Using Fractional Delay Filters*, PhD thesis, Helsinki University of Technology, Espoo, Finland, p. 193.

Välimäki, V. and Takala, T. 1996. Virtual musical instruments—natural sound using physical models, *Organised Sound* **1**(2): 75–86.

Välimäki, V. and Tolonen, T. 1997a. Development and calibration of a guitar synthesizer, *Presented at the 103rd Convention of the Audio Engineering Society, Preprint 4594*, New York, USA.

Välimäki, V. and Tolonen, T. 1997b. Multirate extensions for model-based synthesis of plucked string instruments, *Proceedings of the International Computer Music Conference*, Thessaloniki, Greece, pp. 244–247.

Välimäki, V., Huopaniemi, J., Karjalainen, M. and Jánosy, Z. 1996. Physical modeling of plucked string instruments with application to real-time sound synthesis, *Journal of the Audio Engineering Society* **44**(5): 331–353.

Välimäki, V., Karjalainen, M. and Laakso, T. I. 1993. Modeing of woodwind bores with finger holes, *Proceedings of the International Computer Music Conference*, pp. 32–39.

Välimäki, V., Karjalainen, M., Jánosy, Z. and Laine, U. K. 1992a. A real-time DSP implementation of a flute model, *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 2, San Francisco, California, pp. 249–252.

Välimäki, V., Laakso, T. I. and Mackenzie, J. 1995. Elimination of transients in time-varying allpass fractional delay filters with application to digital waveguide modeling, *Proceedings of the International Computer Music Conference*, Banff, Canada, pp. 303–306.

Välimäki, V., Laakso, T. I., Karjalainen, M. and Laine, U. K. 1992b. A new computational model for the clarinet, *in:* A. Strange (ed.), *Proceedings of the International Computer Music Conference*, International Computer Music Association, San Francisco, CA.