Smart Sound Generation for Mobile Phones

Diploma Thesis
at
Graz University of Music and Dramatic Arts

submitted by

Harald Rainer

Institute for Electronic Music and Acoustics (IEM)
Graz, University of Music and Dramatic Arts
A-8010 Graz, Austria

Graz, February 2003

Thesis Supervisor: o. Univ.-Prof. Mag. DI Dr. Robert Höldrich

Klangerzeugung für Mobiltelefone

Diplomarbeit
an der
Kunstuniversität Graz

vorgelegt von

Harald Rainer

Institut für Elektronische Musik und Akustik (IEM)
Kunstuniversität Graz
A-8010 Graz

Graz, im Februar 2003

Diese Diplomarbeit ist in englischer Sprache verfasst.

Begutachter: o. Univ. Prof. Mag. DI Dr. Robert Höldrich

Abstract

A new feature within mobile phones of the third generation is the "advanced audio functionality". Besides other things, the reproduction of polyphonic ringing tones is of great interest. Therefore, a synthesis unit, which is convenient for mobile phones in respect of memory usage and computational costs is needed.

The aim of this diploma thesis is the investigation of sound synthesis techniques, which are suited for this demands. Out of a wide range of different synthesis techniques, three have been chosen to synthesize the sound of a real piano. These are: the FM synthesis, the sampling synthesis and the digital waveguide synthesis.

These sound synthesis techniques have been implemented in Matlab™. Thus, the synthetically created piano sounds are compared to that of a real piano.

Kurzfassung

Eine Besonderheit von Mobiltelefonen der dritten Generation ist die „Erweiterte Audiofunktionalität". Dazu zählt unter anderem die Wiedergabe von mehrstimmigen Klingeltönen. Zur Erzeugung dieser benötigt man eine Syntheseeinheit, die hinsichtlich Speicherbedarf und Rechenleistung für den Einsatz in Mobiltelefonen geeignet ist.

Ziel dieser Diplomarbeit ist die Untersuchung von Klangsyntheseverfahren, die diesen Anforderungen genügen. Aus einer großen Anzahl an verschiedenen Syntheseverfahren wurden drei für die Synthese eines natürlichen Klavierklanges ausgewählt. Diese sind: die FM Synthese, die Sampling Synthese und die Waveguide Synthese.

Diese Klangsyntheseverfahren wurden in Matlab™ implementiert. Die dadurch erzielten synthetischen Klavierklänge werden mit jenen des realen Klaviers verglichen.

Acknowledgements

I would like to thank Prof. Robert Höldrich for his thesis advice and for supporting me and my project.

I also have to thank Renate for her patience during my hard times of writing this thesis.

Finally I would like to thank my parents Alois und Anneliese Rainer for their overwhelming love and support during all the years of my studies.

Harald Rainer
Graz, Austria, February 2003

Table of Contents

1 Introduction

The aim of this thesis is the investigation of three sound synthesis techniques, which can be implemented on a digital signal processor. Thereby, the provided amount of available memory is strictly limited and the computational performance is an average. As a result, the choice of the used sound synthesis techniques is based on these conditions.

The three sound synthesis techniques this thesis deals with are the FM synthesis, the sampling synthesis and the waveguide synthesis. The first technique is assigned to the group of abstract algorithms. The advantage of these algorithms is that only a few parameters have to be used to control their behavior. However, it is difficult to model the sound of real instruments due to the nonlinear characteristic of these algorithms. The second sound synthesis technique belongs to the group of processed recordings. Thereby, recordings of real sounds are used, which will be processed in either way to result in a desired output signal. The third technique is assigned to the group of physical models. The physics-based approach has several advantages compared to abstract algorithms or processed recordings, where the control parameters have no meaning to the musician. Physical modeling techniques concentrate on the underlying sound production mechanism and not on the modeling of the radiated sound. This is why the parameters of the models will have a more direct interpretation in the real world.

These sound synthesis techniques are investigated by means of the synthesis of the real piano sound. Thereby, some important properties of the real piano should be taken into account for the synthesis stage. These are:

- the inharmonic ratio of the partials caused by the stiffness of the vibrating string,
- the two-stage decay, which can be traced to the different polarizations and the coupling of the strings and
- the spectral behavior, which depends on the attack velocity of the depressed key as well as on time.

As the sound production mechanism and the consistence of the resultant sound of the real piano is quite complex, this is a good choice for analyzing the potential of the different synthesis techniques.

The synthesis of the real piano tone by means of FM synthesis is performed by the use of four sine oscillators, which are interconnected in a particular way. Their frequencies were chosen in a way to model the inharmonicity of the real piano. The time-varying spectral behavior of the real piano tone can be modeled by changing the modulation index for the individual oscillators in proportion to time. Therefore, envelope generators have to be introduced.

In the case of the sampling synthesis, the tone of the real piano is synthesized by using short pre-recorded samples of it. To reduce the demand on required memory, the technique of looping and pitch shifting is introduced. As a result, time-variant filtering of the output

signal has to be performed in order to model the spectral changes of the real piano versus time.

The modeling of the real piano tone with the aid of the digital waveguide synthesis is the last technique described in this thesis. Waveguide synthesis is one of the most widely used physics-based sound synthesis methods in use nowadays. The tasks for developing the structure of a physical model for an instrument are the following: first, the acoustical properties of the instrument have to be carefully investigated. Second, a decision has to be made about which features need to be simulated. The last step is finding efficient implementations for these features. In the digital waveguide synthesis, the physical behavior of a vibrating string can be modeled by a discretization of the wave equation of the ideal string. This leads to two parallel delay lines, which represent the traveling wave components of the medium interior. To model the behavior of the real vibrating string, a loop filter has to be integrated in the waveguide. This enables modeling the losses due to friction with the air and the finite mass of the string. Furthermore, the stiffness of the string has to be taken into account. Since the digital waveguide only models the string, a further model has to be introduced representing the hammer of the real piano. The resultant excitation signal drives the digital waveguide then. Furthermore, the resonating system of the real piano has to be taken into account.

As this thesis deals with the synthesis of the real piano, the acoustical properties of the instrument have to be discussed. This discussion can be found in chapter 2. In chapter 3, different sound synthesis techniques and their behavior with regard to sound quality, memory usage, and computational costs are shortly analyzed. Chapter 4 deals with the principles of the FM synthesis technique and proposes an algorithm for the synthesis of the piano tone. Chapter 5 is about the synthesis of the piano sound by means of the sampling synthesis technique. Thereby, the difficulties in finding convenient samples regarding to the length of the samples are discussed. The modeling of the real piano by means of the digital waveguide synthesis is described in chapter 6. Furthermore, two models for the piano hammers are introduced. As a last resort, the commuted piano synthesis is investigated for a computationally efficient implementation of a piano model. In chapter 7, the researched synthesis techniques are compared in regard to the memory usage and the computational costs. Finally, in chapter 8 results are summarized leading to a conclusion.

2 Acoustical properties of the piano

The piano belongs to the group of struck string instruments. The first piano was built as early as 1709 by Bartolomeo Cristofori, but it had to go through many changes to reach its modern form. In [1-3] the comparison of a new and the 1720 Cristofori piano can be found.

The general structure of the piano is the following: an iron frame is attached to the upper part of the wooden case and the strings are extended upon it in a direction nearly perpendicular to the keyboard. That end of the string, which is closer to the keyboard, is connected to the tuning pins on the pin block. After crossing the bridge, the other end is attached to the hitch pin rail of the frame. The bridge is a thin wooden bar transmitting the vibration of the string to the soundboard, which can be found under the frame.

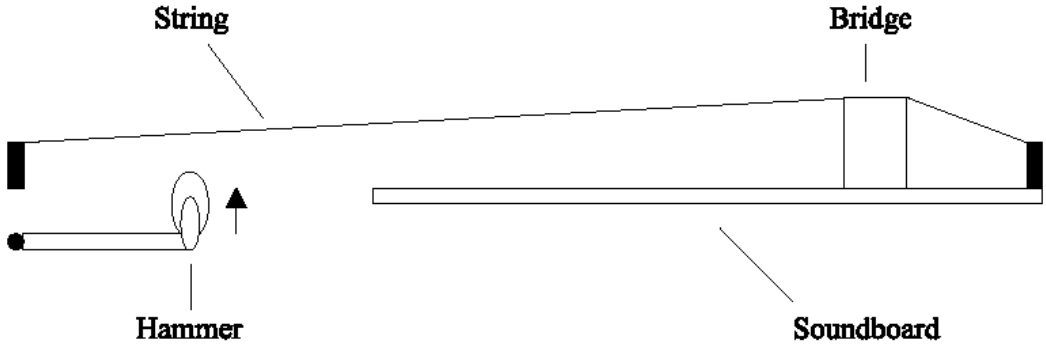

Figure 2.1: Schematic structure of the piano [4]

According to the above-mentioned parts the sound-production mechanism of the piano can be divided into the following steps:

1. First, the hammer strike, which transmits the kinetic energy taken in by the artist to the kinetic energy of the hammer. Therefore, the piano action belongs to this part as well. After bouncing to the string, it is transformed to vibrational energy.
2. The energy is stored by the string, whereas one part of that is dissipated due to internal losses; the other one gets to the soundboard through the bridge.
3. The soundboard converts the vibrational energy to acoustical energy, which results in the audible sound.

2.1 Piano action and hammer

2.1.1 The action

The action of the piano is an artwork of precision mechanics, and its operation is rather complicated. Allowing the fastest possible repetition of a single note results in a very complex mechanical structure. In this way, the repetition can be performed before the hammer reaches its rest position. Generally speaking, the action can be considered as a lever system with a ratio of 1:5 between the movement of the key and the hammer.

It is important to notice that in the moment of hammer-string contact the hammer rotates freely. When the key is pressed down slowly, the hammer stops 4-6 mm beneath the string. Under regular playing conditions, the hammer is moved towards the string by the transferred kinetic energy. By pressing a key the corresponding damper is lifted, which mutes the string by falling back after the key is released [5, 6].

By the examination of the timing of the action, [5] made some interesting observations. They concluded that the delay introduced by the action depends mainly on the dynamic level. In the piano-legato touch, this delay can be as high as 100 ms, while at the forte-staccato touch from the hit of the key to the sound of the note only 25 ms elapse. Since this difference is audible, the skilled pianist must compensate for this performance. A further characteristic trait of the piano, as well as of many other instruments is the alteration of timbre depending on a change of the dynamic level. Figure 2.2 shows three spectra due to playing pianissimo (ppp), mezzo forte (mf), and fortissimo (fff) on the G_3 (196 Hz) key. To facilitate a comparison, the spectra were normalized in order to ensure that they have the same total energy.

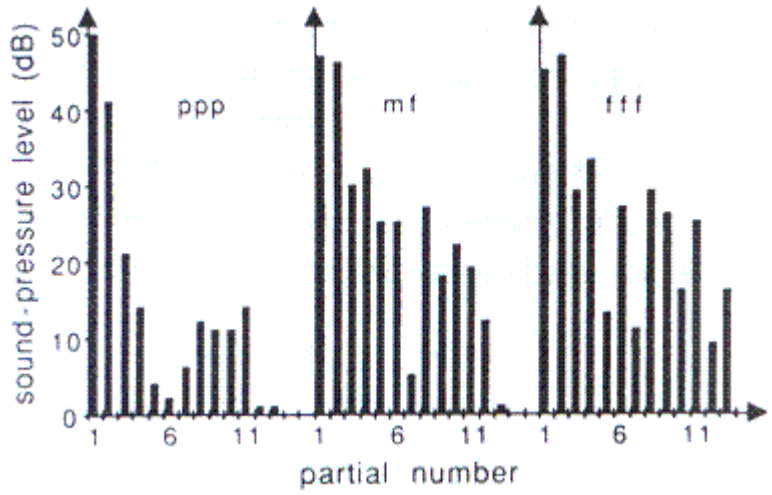

Figure 2.2: Relative sound-pressure level due to playing ppp, mf, and fff [7]

2.1.2 The hammer

Hammers have a great influence on the timbre of the piano, as they excite the strings. The hardwood cores of the hammers are covered by wool felt. The characteristics of the felt influence the resulting sound considerably. Thus, harder felt result in stronger partials, i.e., a brighter tone. On the contrary, softer hammers produce less partials and a softer tone. The timbre of the piano can be influenced by "voicing" (hammers can be made softer by needling, or hardened by a hardening agent). Voicing is the last step of piano production, giving a "personality" to the instrument [2].

The felt of the hammer is not homogeneous. Its hardness changes gradually from the outer part to the core. This is the main reason for the spectral differences at various dynamic levels.

Therefore, the felt can be considered as a nonlinear spring with a stiffness increasing with compression. An approximate power-law model can be found in, e.g., [7]:

$$F = K(\Delta y)^p \tag{2.1}$$

where Δy refers to the compression of the felt, K is the stiffness coefficient and p is the stiffness exponent. These values (K and p) can be determined by curve-fitting from measured data [8]. Some papers suggest a hysteretic model for the hammer felt (see, e.g., [9]).

The process of the hammer-string interaction is the following: the hammer, accelerated by the action, hits the string but it does not bounce back immediately, since its mass is not negligible compared to the string. The hammer is thrown back by the reflected pulses returning from the closer end of the string. This end is denoted as the agraffe. The force experienced by the string and the hammer is a sequence of shock waves. The shape and smoothness (and thus the frequency content) of the force curve is influenced by p, K and the initial velocity as well. Increasing K or the initial velocity has the same effect, they both enlarge the high frequency content of the excitation. The average duration of the hammer-string contact is determined by the ratio of the hammer mass and the mass of the string. The heavier the hammer is, the longer is the time of contact. For an efficient excitation of the string, the contact time is equal to the half period of the tone. In contemporary pianos, hammers of gradually changing mass are used in order to meet this condition.

The spectrum of the resultant sound also depends on the striking point. This results in a comb filtering effect, since those modes of the string, which have a node near to the striking position, cannot be excited effectively [2].

2.2 Piano strings

The strings of the piano are made of steel wire. High efficiency requires high tension (about 700 N for each string), and accordingly the strings are strained at the 30-60% of their specified tensile strength. In order to reach higher acoustic output, three strings for the same note are used (except for the lowest two octaves). These strings are not tuned in perfect unison, introducing beating and two-stage decay, two important characteristics of the piano sound. The length of the strings is not exactly in an inverse proportion to the fundamental frequency. Otherwise, the lowest strings would be too long yielding an unacceptable size of the piano case. To overcome this problem, the mass of the bass strings is increased. On the other hand, thicker strings result in higher inharmonicity, simply because the string begins to behave like a stiff bar. The solution is winding the bass strings with one or two layers of copper wire, which reduces the stiffness [1].

2.2.1 Beating and two-stage decay

Having more strings for exciting the same note increases the efficiency of energy transmission towards the bridge as well as it constitutes the characteristic piano sound. If we assume that the strings are not tuned to the same frequency but neglect the effect of coupling, the result will be beating between the partials. In reality, these strings are not independent since they are coupled to the bridge, new modal frequencies will occur. Coupling can be the reason for two-stage decay as well. This means that in the early part of the tone the sound decays much faster than in the latter part [10].

There is another explanation for the two-stage decay: the behavior of the two different polarizations. Since the vertical polarization is coupled more efficiently to the bridge than the horizontal polarization, the decay times differ significantly. The hammer excites the vertical polarization to a greater extent, and the energy transmission to the bridge is more effective in this direction as well. As a result, at the beginning of the tone the sound produced by the vertical polarization will be the dominant one. As the vibration of the vertical polarization decays faster, the latter part of the tone will be determined by the horizontal polarization [10].

2.2.2 Inharmonicity

As already mentioned, the stiffness of the strings results in a slightly inharmonic sound. (2.2) shows the wave equation of the stiff string[1].

$$\mu \frac{\partial^2 y}{\partial t^2} = T \frac{\partial^2 y}{\partial x^2} - QS\kappa^2 \frac{\partial^4 y}{\partial x^4} \tag{2.2}$$

where x is the position along the string, y is the transversal string displacement, t is the time, T refers to tension and μ to mass density. Q stands for Young's modulus, S is the cross-section area of the string and κ is the radius of gyration. Because of the forth-power term, dispersion will appear, and waves with higher frequency will travel faster on the string, i.e., the wave velocity will not be constant any more. As a result of dispersion, higher modes will go back and move forward on the string within a shorter time, so their frequency will be the same nearby, like that of the ideal string.

It is an important question whether the inharmonicity is a desired factor or just an unavoidable feature. [1] concludes that inharmonicity is an important factor of piano sound, but there should be as little as possible of it.

Inharmonicity also affects piano tuning. Since tuning is based on beating between intervals, stretching of partials will also cause stretching of fundamental frequencies. As a result, the lowest notes of the piano are 30 cent below, while the highest are 30 cent above the tempered values.

[1] For the wave equation of the ideal string see chapter 6 of this diploma thesis

2.3 The bridge and the soundboard

The soundboard of modern pianos is generally made of assembling strips of solid softwood (such as Sitka or red spruce). These 5-15 cm wide strips are glued together. Since the stiffness of the wood is higher along the grain, the cross-grain stiffness is increased by adding ribs to the soundboard. This increases the travel velocity in this direction. Cheaper Pianos are generally built of laminated wood. The comparison of soundboards of different materials can be found in [3].

The vibration of the strings is transmitted to the soundboard through the bridge. The bridge functions as an impedance transformer, presenting higher impedance to the string than that if the strings were directly connected to the soundboard. In the latter case, decay times would be too short. By carefully designing the soundboard and the bridge, the loudness and the decay time of the partials can be set.

The impedance curve of the soundboard exhibits a high modal density. Many studies have been made on the low frequency behavior of the soundboard. These resonances are similar to the simple motion of a plate and can easily be observed by the Chladni method [3]. For the higher frequency region of the soundboard the average of the impedance is constantly up to about 3-7 kHz and after that it starts to decrease with frequency. This decline exists because of the ribs.

2.4 Pedals

Pianos have either two or three pedals. The most important one is the sustain pedal on the right which lifts the dampers of all the strings. This sustains the struck keys on the one hand and changes the character of the timbre on the other hand, since all the other strings can vibrate freely in sympathetic mode.

The left pedal is the “una corda” pedal, which shifts the entire action sideways. Consequently, the hammer strikes only two strings out of three in the treble, or one out of two in the midrange. This causes only about 1 dB reduction of the sound pressure level, but changes the timbre. One of the reasons for this is that the third string gains energy from the vibration of the other two. The initial conditions of the coupled vibration are changed, resulting in a slower decay [10]. Another reason for the spectral change may be the fact that when the same hammer hits a smaller number of strings, it appears to be heavier with respect to the single strings, causing longer contact times and a softer timbre.

The middle pedal is the “sustenuto” pedal, which sustains only those notes that have been hit before depressing the pedal.

3 Overview of synthesis methods

Digital sound synthesis methods are numerical algorithms that aim at producing musically interesting sounds. The following approach for the evaluation of digital sound synthesis methods is based on a classification by [11]. He divides the different methods into four groups:

- abstract algorithms,
- processed recordings,
- spectral models and
- physical models.

An overview of representative synthesis methods of each category follows. For more detailed information see [12-15].

3.1 Abstract algorithms

The advantages of abstract algorithms are their simplicity and the small number of control parameters. However, because of the nonlinear behavior, the analysis procedures are complicated. Thus, the simulation of the sound of numerous real instruments is almost impossible. They are rather useful to create inconvenient sonorities.

3.1.1 FM synthesis

FM (frequency modulation) synthesis is a fundamental digital sound synthesis technique. The fundamental approach of this synthesis method is the distortion of the frequency of an oscillator in accordance with the amplitude of a modulating signal [12]. So it's main advantage is, that even a two-oscillator system can produce a rich spectrum. For further information to FM synthesis, see chapter 4 of this thesis.

3.1.2 Waveshaping synthesis

In waveshaping synthesis, a nonlinear shaping function is used to modify the input signal. The spectrum produced by a waveshaping instrument changes with the amplitude of the sound. This corresponds with the characteristics of the spectra of acoustic instruments. In the most fundamental form, waveshaping is implemented as a mapping of a sinusoidal input signal with a nonlinear distortion function. With the aid of Chebyshev polynomials, the matching of a desired steady-state spectrum can be realized. Therefore, the input signal has to be a cosine wave with the amplitude of one. For full independence between the timbre and the output amplitude, some form of amplitude normalization is used [12, 14].

3.1.3 Karplus-Strong Algorithm

Karplus and Strong [16] developed a method for an amazingly high-quality synthesis of plucked strings and drum sounds. The Karplus-Strong (KS) algorithm is an extension of the simple wavetable synthesis technique. In contrast to wavetable synthesis, where the sound signal is periodically read from computer memory, in the KS algorithm, the wavetable is modified each time a sample is being read. In this way, the content of the wavetable will evolve with time.

However, it was found out soon, that the Karplus-Strong algorithm is a special case of the technique now called digital waveguide modeling [17].

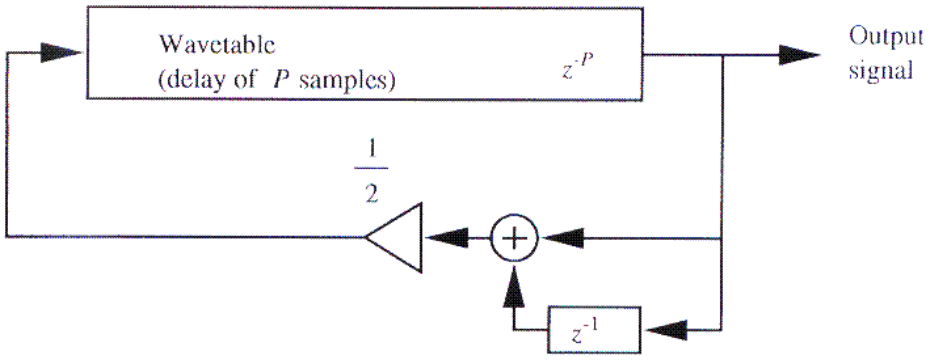

Figure 3.1: Karplus-Strong model for plucked strings [15]

3.2 Processed recordings

The methods described below are based on recording and processing of real sounds. First experiments were done at 1920's by composers like Hindemith, Milhaud, and Toch, who experimented with variable speed phonographs in concert [14].

In the following chapters, three methods utilizing recordings are discussed shortly. These are sampling, multiple wavetable synthesis and granular synthesis.

3.2.1 Sampling synthesis

In sampling synthesis, recordings of relatively short sounds are played back. To change or to vary the characteristic of the sounds, digital sampling instruments use techniques like pitch shifting, looping, and time stretching. The synthesis itself is very efficient to implement, however, the required memory storage is huge. For a detailed description of the sampling syntheses approach, see chapter 5 of this thesis.

3.2.2 Multiple wavetable synthesis

The terms "wavetable synthesis" and "sampling synthesis" are often used synonymously, mainly in industrial lingo, because of their analogical fundamental approaches. The most widely used methods are wavetable cross fading and wavetable stacking [14].

In wavetable cross fading two or more wavetables are used to generate time-varying timbres. Each output of the individual wavetables are multiplied with an amplitude envelope and summed together. The way of generating the output signal is controlled by an oscillator that cross-fades between the various wavetables. So it is possible to synthesize a sound starting with an attack recorded from an acoustic instrument, like a struck of a string, and then cross-fade the sound into a synthetic generated waveform [14, 15].

Wavetable stacking is similar to additive synthesis, were sine waves are summed together to generate a certain sound. In wavetable stacking, not sine waves but complicated waveforms such as sampled sounds are summed together to generate rich hybrid textures. Surveying commercial synthesizers shows, that usually four to eight wavetables are used in wavetable stacking [14].

In [18] methods for matching the time-varying spectra of a harmonic wavetable-stacked tone to an original are presented. The overall objective is to find wavetable spectra and associated amplitude envelopes, which together provide a close fit on an original time-varying spectrum.

3.2.3 Granular synthesis

In granular synthesis, the sound signal is produced by adding sound grains in the time domain, where one sound grain can have a duration ranging from one millisecond to more than a hundred milliseconds. The waveform of the grain can be a windowed sinusoid or a sampled signal. The algorithms can be divided into asynchronous and pitch synchronous methods.

Asynchronous granular synthesis was developed by Roads [14]. This method 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 [14]. 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. This method is rather used to generate synthetic waveforms. It is not suited to reproduce the behavior of acoustic instruments.

In pitch synchronous granular synthesis, grains are derived from the short-time Fourier transform. From these analysis grains, impulse responses corresponding to prominent content in the frequency domain representation have derived. In the resynthesis stage a pulse train is used to drive a set of FIR filters. The output signal results from the excitation of the pulse train on the weighted sum of the impulse responses of all the filters.

3.3 Spectral models

Spectral sound synthesis methods are based on modeling the behavior of sound waves in the frequency domain, which is more similar to the human sound perception. Some of them also consider psychoacoustic criteria.

In this chapter, three methods, namely, additive synthesis, source-filter synthesis and spectral modeling synthesis are shortly discussed.

3.3.1 Additive synthesis

Additive synthesis is one of the oldest synthesis techniques. The concept is to generate a composite waveform by summing sinusoidal components with different frequency and amplitude envelopes. Sometimes additive synthesis is also called “Fourier synthesis”, since the theoretical framework is the Fourier transform [13].

In additive synthesis, the control parameters for each sinusoidal oscillator are the amplitude, the frequency and in some cases the phase. For instance, these parameters can be obtained by a bank of narrow bandpass filters, where every filter is tuned to a specific center frequency.

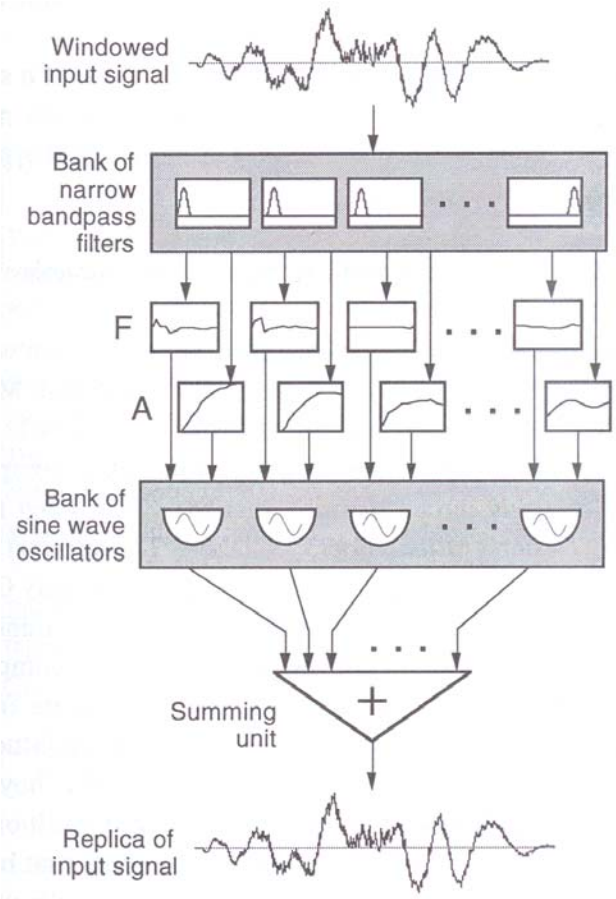

Figure 3.2: Additive analysis and synthesis stage after [14]

For an efficient digital implementation of an additive synthesis algorithm, the filter bank can be replaced by the short-time Fourier transform. Anyway, the main drawbacks of the additive synthesis are the enormous amount of data involved and the demand for a large number of oscillators [15].

3.3.2 Source-Filter Synthesis

In Source-Filter Synthesis the desired output signal is obtained by filtering a spectrally rich source sound. Sometimes the method is also called subtractive synthesis. Originally developed for speech synthesis this technique can be applied to simulate the sound of traditional instruments [14].

Figure 3.3 depicts a block diagram of the method.

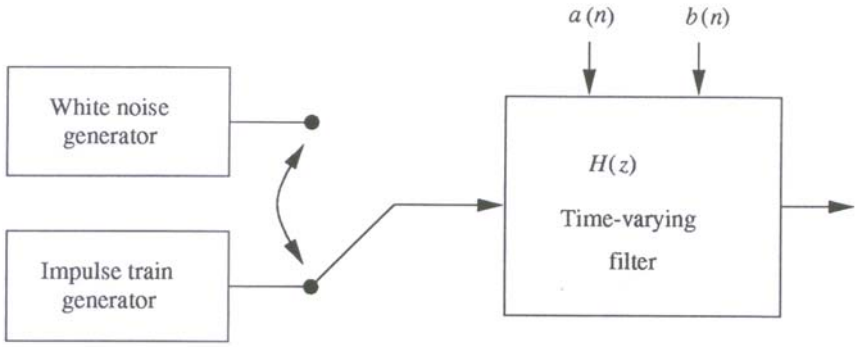

Figure 3.3: Block diagram of the source-filter synthesis method after [15]

Here, a harmonically rich excitation signal is filtered to get the desired output signal. The filter acts as a resonating system with time-varying coefficients *a(n)* and *b(n)*, where the computation of the coefficients can be accomplished by linear predictive analysis. Since many traditional instruments have a stationary or a slowly time-varying resonating system, source-filter synthesis can be used to model these instruments.

3.3.3 Spectral modeling synthesis

The spectral modeling synthesis (SMS) technique is based on the assumption that the input signal can be represented as the sum of a deterministic and a stochastic component. The deterministic part of the signal can be obtained by using a magnitude-only analysis or by using the MQ algorithm [19]. This component is a data-reduced version of the analysis that models the prominent frequencies in the spectrum. Subtracting the deterministic signal from the original one produces a residual signal, which corresponds to the stochastic component. The residual signal can be modeled by the time-domain convolution of white noise with a time-varying frequency-shaping filter [14, 15].

3.4 Physical Models

The aim of physical modeling is to describe the acoustic and mechanical behavior of an instrument. Based on the fact that not the sound of the instrument but the instrument itself is modeled, it is possible to create sounds of instruments that would otherwise be impossible to built [14]. Theoretically, the physical modeling approach could deliver the most realistic synthetic instrument sounds. Their drawback is the loss of generality and the high computational costs.

3.4.1 Modal Synthesis

Modal synthesis starts from the premise that a sound-producing object can be represented as a set of vibrating substructures. Typical substructures include violin bridges, violin bodies, acoustic tubes, bells, drumheads and so on [14]. By an excitation of these substructures with the corresponding physical parameters (force, air flow etc.), a set of natural "modes of vibration" is obtained. These modes are specific to a particular structure. As the sound-producing mechanisms are separated into substructures, it is possible to add or subtract such substructures to create time-varying synthesis effects. Furthermore, this method also permits an interpolation from one timbre to another by combining substructures in an unnatural manner [14].

3.4.2 Finite difference method

The principle of this method is the mathematical description of the vibratory motion in the object under study. The received wave equations are then solved for a finite number of points along the object, thus obtaining a difference equation. [20] were the first to take the approach of solving the differential equations of a vibrating string for the purpose of sound synthesis. Since that pioneer work, developments have been made in modeling the excitation, e.g., the interaction of the hammer and the piano strings, see ([8, 21]) for references.

For real-time sound synthesis, the finite difference model is not very attractive as it can only be applied to a simple structure.

3.4.3 Digital Waveguides

Digital waveguide models are physical models for certain classes of musical instruments, which are made up of delay lines and digital filters. In principle, they can be viewed as a particular class of finite difference schemes for numerical physical modeling.

For modeling instruments with two- or three-dimensional vibrators, the digital waveguide can be expanded to a waveguide mesh. To be able to formulate a waveguide mesh, a

junction of waveguides needs to be developed. Applications of waveguide meshes for instance are modeling soundboards, cymbals, membranes and gongs [15].

A detailed description to digital waveguide models will be given in chapter 6 of this thesis.

3.5 The synthesis methods by comparison

So far, only the basic principles of several synthesis methods have been described. Since the aim of this diploma thesis is the investigation of synthesis techniques by means of their computational efficiency, their memory requirements, and their reachable sound quality, now the aforementioned synthesis methods should be compared regarding to these properties. For this purpose, the evaluation of sound synthesis methods made by [15] is adopted as follows:

	Sound quality	Computational cost	Memory usage
Abstract			
FM	*	***	***
Waveshaping	*	***	***
Karplus-Strong	**	***	***
Sampling			
Sampling	***	***	*
Multiple WT	**	**	*
Granular	***	**	**
Spectral			
Additive	**	**	*
Source-filter	**	**	**
SMS	***	**	**
Physical			
Modal	***	**	*
Finite Difference	***	*	**
Waveguide	***	**	***

Table 3.1: Tabulated evaluation of the sound synthesis methods; *…poor, **…fair, ***…good

4 FM Synthesis

4.1 Theory

In 1973, John Chowning at Stanford University released the technical bases of frequency modulation for sound synthesis [22]. With the use of this sound synthesis method the possibility of changing the spectral behavior of a sound with very great simplicity was born.

4.1.1 The basic technique of FM

In simple FM synthesis, the frequency of a sinusoidal oscillator (carrier) is modulated by another sinusoidal oscillator (modulator) to generate a complex waveform. The spectral characteristics depend on the modulation index and the parameters of the two sine waves. The FM signal *x(t)* is given by (4.1).

$$x(t) = A\sin\left[2\pi f_c t + I\sin\left(2\pi f_m t\right)\right] \tag{4.1}$$

where A is the amplitude, f_c is the carrier frequency, f_m is the modulation frequency, and I is the modulation index. Figure 4.1 diagrams a simple FM instrument which meets (4.1).

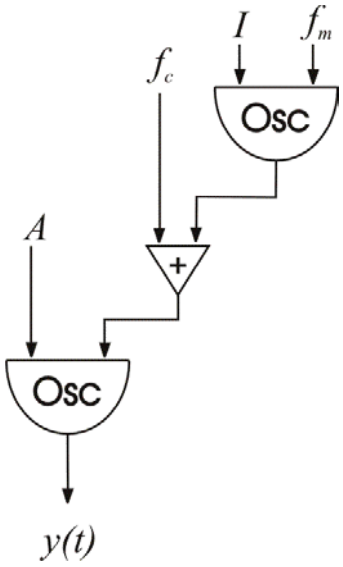

Figure 4.1: A simple FM instrument

4.1.2 The Spectrum of Simple FM

The number of occurring side frequencies is related to the modulation index I in such a way that an increasing I yields a wider distribution of the power among the sidebands as well as an increasing number of sidebands with significant amplitudes.

The total bandwidth is approximately equal to twice the sum of the peak frequency deviation d and the modulation frequency [22]. In formal terms:

$$bandwidth \approx 2(d + f_m) = 2(I + 1)f_m \tag{4.2}$$

4.1.2.1 Bessel Functions

Bessel functions of first kind and *nth* order, $J_n(I)$, where the argument to the function is the modulation index I are used to determine the amplitude of the carrier and the sideband components.

Equation (4.1) can also be written as

$$\begin{aligned} x(t) &= \frac{A}{2j} \cdot e^{j\omega_c t} e^{jI\sin(\omega_m t)} - \frac{A}{2j} \cdot e^{-j\omega_c t} e^{-jI\sin(\omega_m t)} \\ &= \frac{A}{2j} \cdot e^{j\omega_c t} \cdot y(t) - \frac{A}{2j} \cdot e^{-j\omega_c t} \cdot y^{-1}(t) \end{aligned} \tag{4.3}$$

$$\text{with:} \quad y(t) = e^{jI\sin(\omega_m t)} \tag{4.4}$$

The next step is to expand $y(t)$ into a fourier series representation.

$$y(t) = \sum_{n=-\infty}^{\infty} c_n \cdot e^{jn\omega_m t} \tag{4.5}$$

The coefficients are:

$$c_n = \frac{1}{T} \int_{-T/2}^{T/2} y(t) \cdot e^{-jn\omega_m t} dt = \frac{1}{T} \int_{-T/2}^{T/2} e^{j[I\sin(\omega_m t) - n\omega_m t]} dt \tag{4.6}$$

With the help of the following substitution

$$\varphi = \omega_m t = \frac{2\pi}{T} t, \quad d\varphi = \omega_m dt = \frac{2\pi}{T} dt, \quad \frac{1}{T} \int_{-T/2}^{T/2} dt = \frac{1}{2\pi} \int_{-\pi}^{\pi} d\varphi \tag{4.7}$$

the coefficients alter from (4.6) to (4.8).

$$c_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} e^{j(I\sin\varphi - n\varphi)} d\varphi = J_n(I) \tag{4.8}$$

Now by inserting (4.8) into (4.5) and the result into (4.4) this yields a trigonometry series representation of $x(t)$.

$$x(t) = A \sum_{n=-\infty}^{\infty} J_n(I) \sin\left((\omega_c + n\omega_m)t\right) \tag{4.9}$$

An inspection of (4.9) reveals that the frequency-domain representation of the signal *x(t)* consists of a peak at f_c and additional peaks at frequencies

$$f_n = f_c \pm nf_m, \quad n = 1, 2...$$

Note that (4.9) allows the partials to be determined analytically.

4.1.2.2 Harmonic and Inharmonic Spectra

The ratio of the carrier frequency to the modulating frequency is the decisive factor for the kind of spectra obtained by frequency modulation, thus

$$\frac{f_c}{f_m} = \frac{N_1}{N_2} \tag{4.10}$$

If N_1 and N_2 are integers, the result is a harmonic spectra and the fundamental frequency of the modulated wave will be

$$f_0 = \frac{f_c}{N_1} = \frac{f_m}{N_2} \tag{4.11}$$

Inharmonic spectra will result if (4.10) is a ratio of irrational numbers. In that case, the reflected side frequencies will fall between the positive components and the fundamental frequency. Especially if the modulation index increases this is very hard to be detected by the listeners.

4.2 Generation of a sinusoidal oscillator

[23] splits up algorithms to produce harmonic sequences into three categories:

- Direct calculation of sin(Θn)
- Recursive algorithms
- Wavetable algorithms

4.2.1 Direct calculation of sin(Θn)

With the following series expansion values of the trigonometric function sin(Θn) are calculated numerically:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + -.... = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n+1}}{(2n+1)!} \qquad \text{for } |x| < \infty \tag{4.12}$$

The calculation of the above mentioned series expansion is aborted after the n^{th} element, which yields an error. Due to the fact that the sine is a periodic function, the argument can be limited to the area of *0* to *2π*.

Therefore

$$\sin(\Theta n) = \sin(x) \qquad \text{with: } x = (\Theta n) \bmod (2\pi) \tag{4.13}$$

Now it is possible to reduce the error of calculation by considering the half and fourth cycle symmetry of the sine functions. This means that the function has to be calculated only for argument values between *0* and π respectively *0* and π*/2*.

Half-wave symmetry

$$\sin(\Theta n) = \begin{cases} \sin(x) \text{.....for } 0 \le x < \pi \\ -\sin(x-\pi) \text{....for } \pi \le x < 2\pi \end{cases} \tag{4.14}$$

$$\text{with: } x = (\Theta n) \bmod (2\pi)$$

Quarter-wave symmetry

$$\sin(\Theta n) = \begin{cases} \sin(x) \text{.....for } 0 \le x < \pi/2 \\ \sin(\pi - x) \text{.....for } \pi/2 \le x < \pi \\ -\sin(x-\pi) \text{.....for } \pi \le x < 3\pi/2 \\ -\sin(2\pi - x) \text{....for } 3\pi/2 \le x < 2\pi \end{cases} \tag{4.15}$$

$$\text{with: } x = (\Theta n) \bmod (2\pi)$$

4.2.2 Recursive algorithms

With the aid of recursive algorithms a further method to generate sine waves is given. Thereby the instant values of the sine sequence are calculated by preceded values and state variables of the system. In the following, two methods [23] of recursive algorithms are shortly mentioned.

4.2.2.1 The Coupled Form

The coupled first-order form can be expressed as a pair of equations:

$$\begin{aligned} x[n] &= \cos(\Theta)\, x[n-1] + \sin(\Theta)\, y[n-1] \\ y[n] &= -\sin(\Theta)\, x[n-1] + \cos(\Theta)\, y[n-1] \end{aligned} \tag{4.16}$$

or in matrix form as

$$\begin{bmatrix} x[n] \\ y[n] \end{bmatrix} = \begin{bmatrix} \cos(\Theta) & \sin(\Theta) \\ -\sin(\Theta) & \cos(\Theta) \end{bmatrix} \begin{bmatrix} x[n-1] \\ y[n-1] \end{bmatrix} = \begin{bmatrix} \cos(\Theta) & \sin(\Theta) \\ -\sin(\Theta) & \cos(\Theta) \end{bmatrix}^n \begin{bmatrix} x[0] \\ y[0] \end{bmatrix} \tag{4.17}$$

This is equivalent to

$$\begin{bmatrix} x[n] \\ y[n] \end{bmatrix} = \begin{bmatrix} \cos(n\Theta) & \sin(n\Theta) \\ -\sin(n\Theta) & \cos(n\Theta) \end{bmatrix} \begin{bmatrix} x[0] \\ y[0] \end{bmatrix} \tag{4.18}$$

With the initial condition of

$$\begin{bmatrix} x[0] \\ y[0] \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{4.19}$$

it can be seen that $x[n]$ is a cosine-shaped and $y[n]$ is a sine-shaped sequence with frequency Θ. Since finite word-length machines have the property to be numerical instable, the determinant of the matrix is not necessarily equal to one, which results in waveforms that either decay rapidly or grow very fast in amplitude.

4.2.2.2 The Modified Coupled Form

Using a set of coefficients, which is slightly different to that used in the coupled form yields a stable behavior of the system.

$$\begin{aligned} x[n+1] &= x[n] - \varepsilon y[n] \\ y[n+1] &= \varepsilon x[n+1] + y[n]' \end{aligned} \tag{4.20}$$

which in matrix form becomes

$$\begin{bmatrix} x[n+1] \\ y[n+1] \end{bmatrix} = \begin{bmatrix} 1 & -\varepsilon \\ \varepsilon & 1-\varepsilon^2 \end{bmatrix}^n \begin{bmatrix} x[0] \\ y[0] \end{bmatrix}. \tag{4.21}$$

In any case here the determinant of the matrix is one, independent from the inaccurate representation of ε. The interrelationship between the frequency Θ and the coefficient ε is as follows:

$$\sin\left(\frac{\Theta}{2}\right) = \frac{\varepsilon}{2} \quad \text{and} \quad \cos\left(\frac{\Theta}{2}\right) = \sqrt{1-\varepsilon^2/4}\,. \tag{4.22}$$

4.2.3 Wavetable algorithms

A further method to generate sine waves is the so-called table lookup method. Thereby, a waveform memory contains samples of one period of the sine wave to be generated. The

increment I is repeatedly added to the phase register at each clock pulse and the contents of this register are used to address the waveform memory whereby the method of forming the table address is important for determining the quality of the digital output signal. [24] elucidated three prospects to gain the function values:

- Truncation method
- Rounding method
- Interpolation method

The output frequency f_1 is given by:

$$f_1 = \frac{If_s}{L} \tag{4.23}$$

where I is the increment, f_s is the sampling frequency, and L is the table length. Figure 4.2 illustrates the principle of the table lookup method.

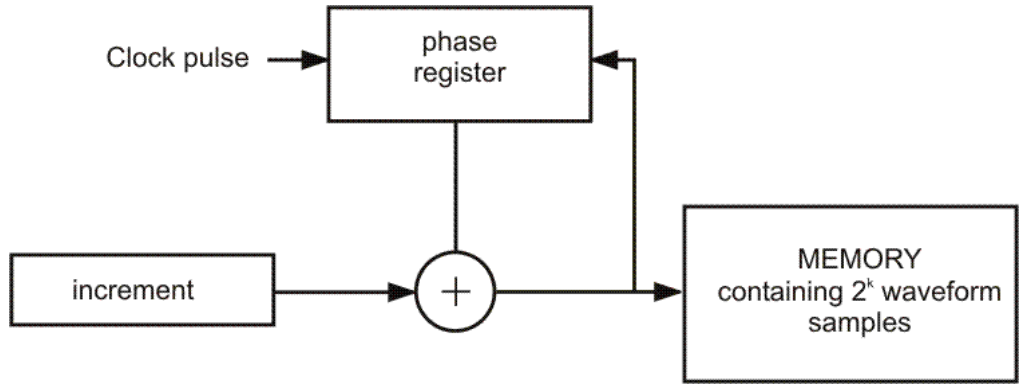

Figure 4.2: Block diagram of the hardware required for the look-up oscillator.

4.2.3.1 Truncation of the phase value

By using this method the argument x with 0 P $x < 2\pi$ is converted into the table address S with 0 P $S < L$-1 by truncation. The arising error is noticeable as noise and is referred to as "phase jitter". The signal-to-noise ratio is determined by the resolution of the function values in k bit and the memory length L.

$$SNR_{truncation} = 10\log\left\{\frac{\sum_{n=1}^{L}\left\{\left(Q_{k-1}\left[\sin\left(\frac{2\pi}{L}(n-0.5)\right)\right]\frac{2\pi}{L}\right)^2\right\}}{\sum_{n=1}^{L}\left\{\int_{x=\frac{2\pi}{L}(n-0.5)}^{\frac{2\pi}{L}(n+0.5)}\left(Q_{k-1}\left[\sin\left(\frac{2\pi}{L}(n-0.5)\right)\right]-\sin\left(x+\frac{2\pi}{L}(n-1)\right)\right)^2 dx\right\}}\right\} \quad (4.24)$$

$Q_{k-1}[..]$ ...quantisation to k bit (sign bit included)

$\underline{SNR_{truncation}}$ dB		Memory Word length in k bit						
		8	10	12	14	16	20	24
Memory Length L (one full period)	256	35.9	36.9	36.9	36.9	36.9	36.9	36.9
	512	40.3	42.7	42.9	42.9	42.9	42.9	42.9
	1024	42.5	48.1	48.9	48.9	48.9	48.9	48.9
	2048	43.2	52.3	54.6	54.8	54.8	54.8	54.8
	4096	43.6	54.5	59.9	60.6	60.7	60.7	60.7
	8192	43.6	55.5	64.0	66.1	66.3	66.3	66.3

Table 4.1: Signal-to-Error Noise Ratios for the truncating oscillator

It can be seen from Table 4.1 that after a certain value of the memory word length no improvement is gained by adding more bits of precision.

4.2.3.2 Rounding of the phase value

Rounding of the phase value to m bit demands an addition of bit $m - 1$ in place m. The formula to calculate the signal-to-noise ratio is as follows:

$$SNR_{rounding} = 10\log\left\{\frac{\sum_{n=1}^{L}\left\{\left(Q_{k-1}\left[\sin\left(\frac{2\pi}{L}(n-0.5)\right)\right]\frac{2\pi}{L}\right)^2\right\}}{\sum_{n=1}^{L}\left\{\int_{x=\frac{2\pi}{L}(n-1)}^{\frac{2\pi}{L}n}\left(Q_{k-1}\left[\sin\left(\frac{2\pi}{L}(n-0.5)\right)\right]-\sin\left(x+\frac{2\pi}{L}(n-1)\right)\right)^2 dx\right\}}\right\} \quad (4.25)$$

$\underline{SNR_{rounding}}$ dB		Memory Word length in k bit						
		8	10	12	14	16	20	24
Memory Length L (one full period)	256	39.9	42.8	43.0	43.0	43.0	43.0	43.0
	512	42.6	48.2	49.0	49.0	49.0	49.0	49.0
	1024	43.4	52.4	54.8	55.0	55.0	55.0	55.0
	2048	43.5	54.8	60.3	61.0	61.0	61.0	61.0
	4096	43.6	55.4	64.4	66.8	67.0	67.0	67.0
	8192	43.6	55.8	66.8	72.2	72.9	72.9	72.9

Table 4.2: Signal-to-Error Noise Ratio for the rounding oscillator

As already mentioned in the truncation method, no improvement is gained by adding more bits of precision after a certain value of the memory word length.

4.2.3.3 Interpolation of the phase value

For equidistant sampling points, as in the case of the stored sine sequence, the interpolation formula by Gregory-Newton [25] is most suitable.

$$I_n(x) = y_0 + \frac{\Delta^1 y_0 (x - x_0)}{1!h} + \frac{\Delta^2 y_0 (x - x_0)(x - x_1)}{2!h^2} + \ldots + \frac{\Delta^n y_0 (x - x_0)(x - x_1) \cdot \ldots \cdot (x - x_{n-1})}{n!h^n} \tag{4.26}$$

with

$h = x_{k+1} - x_k$ ...increment

$\Delta^0 y_k := y_k$

$\Delta^i y_k := \Delta^{i-1} y_{k+1} - \Delta^{i-1} y_k$ ...difference of n-th order

By interpolation of the sine sequence, the argument x can be represented as the sum of integer part *int* and fractional part *frac*. This leads to the following results:

a) For linear interpolation (*n* = 1):

$$\begin{aligned} I_1(x) &= I_1(int + frac)\Big|_{int}^{int+1} \\ &= Q_{k-1}\left[\sin(int)\right] + \left(Q_{k-1}\left[\sin(int+1)\right] - Q_{k-1}\left[\sin(int)\right]\right) frac \end{aligned} \tag{4.27}$$

where $Q_{k-1}\left[\sin(int)\right]$ is the integer part of the function value quantized by *k* bit. The resulting signal-to-noise ratio is given by equation (4.28):

$$SNR_{lin.interp.} = 10\log\left\{\frac{\sum_{n=1}^{L}\left\{\int_{x=n}^{n+1}\left(Q_{k-1}\left[I_1(x)\Big|_n^{n+1}\right]\right)^2 dx\right\}}{\sum_{n=1}^{L}\left\{\int_{x=n}^{n+1}\left(Q_{k-1}\left[I_1(x)\Big|_n^{n+1}\right] - \sin\left(\frac{2\pi}{L}(x+n)\right)\right)^2 dx\right\}}\right\} \tag{4.28}$$

$Q_{k-1}\left[I_s(x)\Big|_a^b\right]$ is the value of x interpolated between *a* and *b* using a s-th order interpolation function. Subsequently this value is quantized by *k* bit. The results are listed below:

$\underline{SNR_{lin.\ interp.}}$ dB		Memory Word length in *k* bit						
		8	10	12	14	16	20	24
Memory Length *L* (one full period)	64	37.5	49.0	56.0	59.8	60.8	61.1	61.1
	128	38.3	50.2	60.7	68.5	71.9	73.1	73.1
	256	38.9	50.6	61.9	72.7	80.5	84.9	85.2
	512	39.3	50.8	62.7	74.4	85.5	95.9	97.2
	1024	40.7	50.9	63.1	74.7	86.4	104.5	108.9
	2048	42.0	51.5	63.1	75.1	87.2	109.3	120.0

Table 4.3: Signal-to-Error Noise Ratio for the interpolating oscillator using linear interpolation

Increasing the memory word-length for the interpolating oscillator influences the Signal-to-Error Noise Ratios to an increasing degree compared to the truncating and the rounding oscillator.

b) For quadratic interpolation (n = 2):

For $frac < 0.5$:

$$I_2(x) = I_2(int + frac)\Big|_{int}^{int+0.5} = Q_{k-1}\left[\sin(int)\right] + \\ + \left(-\frac{1}{2}Q_{k-1}\left[\sin(int-1)\right] + \frac{1}{2}Q_{k-1}\left[\sin(int+1)\right]\right) frac + \\ + \left(\frac{1}{2}Q_{k-1}\left[\sin(int-1)\right] - Q_{k-1}\left[\sin(int)\right] + \frac{1}{2}Q_{k-1}\left[\sin(int+1)\right]\right) frac^2 \tag{4.29}$$

For $frac \geq 0.5$:

$$I_2(x) = I_2(int + frac)\Big|_{int+0.5}^{int+1} = Q_{k-1}\left[\sin(int)\right] + \\ + \left(-\frac{3}{2}Q_{k-1}\left[\sin(int)\right] + 2Q_{k-1}\left[\sin(int+1)\right] - \frac{1}{2}Q_{k-1}\left[\sin(int+2)\right]\right) frac + \\ + \left(\frac{1}{2}Q_{k-1}\left[\sin(int)\right] - Q_{k-1}\left[\sin(int+1)\right] + \frac{1}{2}Q_{k-1}\left[\sin(int+2)\right]\right) frac^2 \tag{4.30}$$

The signal-to-noise ratio is defined as:

$$SNR_{quad.interp.} = 10\log\left\{\frac{\sum_{n=1}^{L}\left\{\int_{x=n}^{n+1}\left(Q_{k-1}\left[I_2(x)\Big|_n^{n+1}\right]\right)^2 dx\right\}}{\sum_{n=1}^{L}\left\{\int_{x=n}^{n+1}\left(Q_{k-1}\left[I_2(x)\Big|_n^{n+1}\right] - \sin\left(\frac{2\pi}{L}(x+n)\right)\right)^2 dx\right\}}\right\} \tag{4.31}$$

$\underline{SNR_{quad.\ interp.}}$ DB		Memory Word length in k bit						
		8	10	12	14	16	20	24
Memory Length L (one full period)	4	16.6	16.8	16.8	16.8	16.8	16.8	16.8
	8	32.5	34.2	34.3	34.3	34.3	34.3	34.3
	16	38.3	48.1	51.7	52.3	52.3	52.3	52.3
	32	38.5	49.9	62.8	69.4	71.2	71.3	71.3
	64	38.8	50.9	61.7	74.1	83.0	85.5	85.5
	128	38.9	50.5	62.4	74.1	82.2	83.8	83.8

Table 4.4: Signal-to-Error Noise Ratio for the interpolating oscillator using quadratic interpolation

c) For cubic interpolation (n = 3):

$$
\begin{aligned}
I_3(x) = I_3(int + frac)\Big|_{int}^{int+1} &= Q_{k-1}\left[\sin(int)\right] + \\
&+\left(-\frac{1}{3}Q_{k-1}\left[\sin(int-1)\right] - \frac{1}{2}Q_{k-1}\left[\sin(int)\right] + Q_{k-1}\left[\sin(int+1)\right] - \frac{1}{6}Q_{k-1}\left[\sin(int+2)\right]\right) frac + \\
&+\left(\frac{1}{2}Q_{k-1}\left[\sin(int-1)\right] - Q_{k-1}\left[\sin(int)\right] + \frac{1}{2}Q_{k-1}\left[\sin(int+1)\right]\right) frac^2 + \\
&+\left(-\frac{1}{6}Q_{k-1}\left[\sin(int-1)\right] + \frac{1}{2}Q_{k-1}\left[\sin(int)\right] - \frac{1}{2}Q_{k-1}\left[\sin(int+1)\right] + \frac{1}{6}Q_{k-1}\left[\sin(int+2)\right]\right) frac^3
\end{aligned}
\tag{4.32}
$$

The signal-to-noise ratio is given by (4.33)

$$
SNR_{cub.interp.} = 10\log\left\{\frac{\sum_{n=1}^{L}\left\{\int_{x=n}^{n+1}\left(Q_{k-1}\left[I_3(x)\Big|_n^{n+1}\right]\right)^2 dx\right\}}{\sum_{n=1}^{L}\left\{\int_{x=n}^{n+1}\left(Q_{k-1}\left[I_3(x)\Big|_n^{n+1}\right] - \sin\left(\frac{2\pi}{L}(x+n)\right)\right)^2 dx\right\}}\right\} \tag{4.33}
$$

$\underline{SNR_{cub.\ interp.}}$ DB		Memory Word length in *k* bit						
		8	10	12	14	16	20	24
Memory Length *L* (one full period)	4	20.0	20.8	20.9	20.9	20.9	20.9	20.9
	8	35.2	41.7	43.6	44.0	44.2	44.3	44.3
	16	38.6	49.6	58.4	65.4	76.2	68.0	68.0
	32	38.6	49.9	63.4	73.4	83.5	89.6	90.4
	64	37.9	50.9	61.7	74.4	86.0	97.8	98.3
	128	39.0	50.5	62.5	74.7	87.1	100.8	101.4

Table 4.5: Signal-to-Error Noise Ratio for the interpolating oscillator using cubic interpolation

4.2.3.4 Consideration of quarter-wave symmetry

As already mentioned in chapter 4.2.1, consideration of quarter-wave symmetry reduces the memory length *L* of the wavetable to a fourth of the original length. Furthermore, the resolution of the function values can be reduced by one bit due to the fact, that the sign bit is no longer required.

This means that in case of linear interpolation the memory length *L* can be chosen to be 64 in order to get a SNR of about 85 dB by a used memory word length of 24 bits. These values should be sufficient for the target application.

4.2.3.5 Frequency resolution

To avoid aliasing, the increment I have to be smaller than the half memory length L. If the word length of the increment with included sign bit is l bit, than only 2^l increment values can be represented. The achievable frequency resolution $f_{\cap}$ is thereby determined by

$$f_{\Delta} = \frac{f_s / 2}{2^l} \ , \tag{4.34}$$

where f_s is the sampling frequency in Hertz.

4.3 The operators

Beneath Simple FM where one oscillator modulates one carrier oscillator, there are many other possibilities how to combine several oscillators in order to obtain a complex FM sound. Since we are talking about synthesis in the digital domain, it is sufficient to know that an operator is equivalent to an oscillator in an analog synthesizer. Instead of changing voltage values, an operator performs essentially the same task by producing a series of changing numbers whose pattern is always that of a sine wave [26].

4.3.1 The interconnection of the Operators

The aim is to interconnect a limited number of operators in such a way that the resulting output signal is similar to the tone of a real piano[1]. For the target application four operators are used to create one piano tone. For the purpose of a 16-note polyphony, the number of operators increase to 64, determining the boundary of hardware requirements.

The input parameters for each operator are the frequency f_k and the amplitude factor a_k where k specifies the current operator.

$$f_k = (ratio_k f_0 + offset_k) \qquad k = 1...4 \ . \tag{4.35}$$

In this equation, $ratio_k$ is the frequency ratio of the individual operators, f_0 is the fundamental frequency, and $offset_k$ is the frequency offset of the individual operators. Thus, it is possible to slightly detune a specified operator.

In the further considerations for the FM algorithm, index k determines a certain operator and its input parameters.

[1] The choice of interconnection of the operators and all used parameters are similar to the piano algorithm implemented in the Yamaha DX7.

4.3.2 Accurate determination of the modulation index

To ensure a simple control of the algorithm it is important to reduce the number of the used parameters. Due to the fact that an operator may act as a carrier in one algorithm and as a modulator in another one an interrelationship for the modulation index has to be found. So it is possible to reduce the number of required parameters up to only one per operator. The modulation index as a function of output values is shown in Figure 4.3.

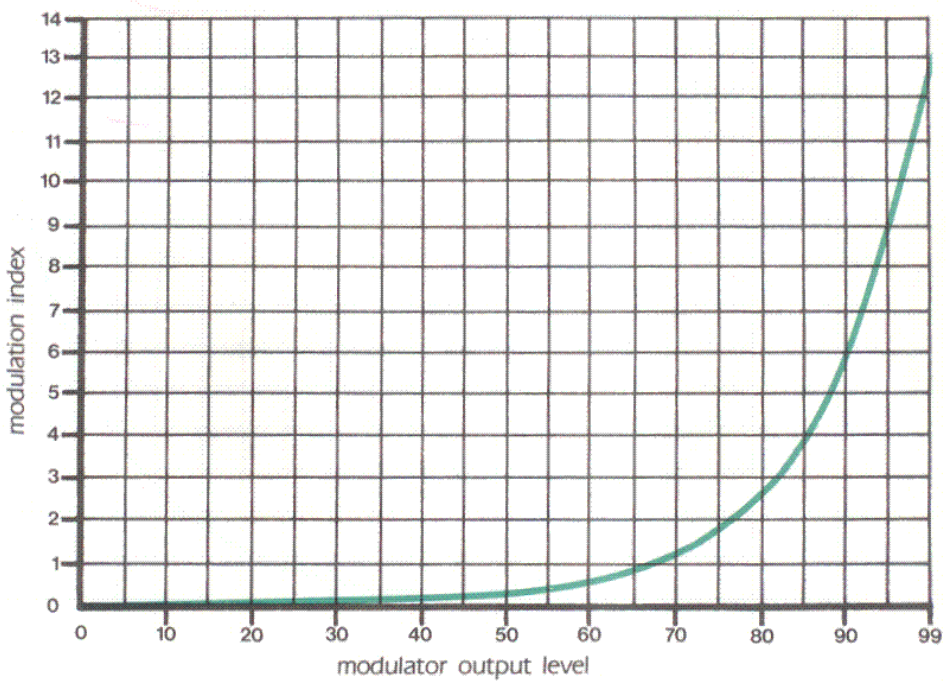

Figure 4.3: Modulation index as a function of output values [26]

4.4 Key scaling

To simulate the spectral behavior of a real piano in subject to the pressed key number, the modulation index I has to be changed according to the pitch of the tone to be generated. In that case the amplitude input of the operators alter from $a_k = I_k$ to

$$a_k(m) = I_k + kscale_k(m) \quad k = 1...4; m = 0...127, \tag{4.36}$$

where $kscale_k(m)$ represents the scaling factor for note number m. The note numbers satisfy the specifications of the MIDI Standard and the resultant fundamental frequency can be found by the following equation [12].

$$f_0 = 440 \cdot 2^{(note-69)/12} \tag{4.37}$$

In Figure 4.4 a key scaling as a function of note number is shown for the individual operators.

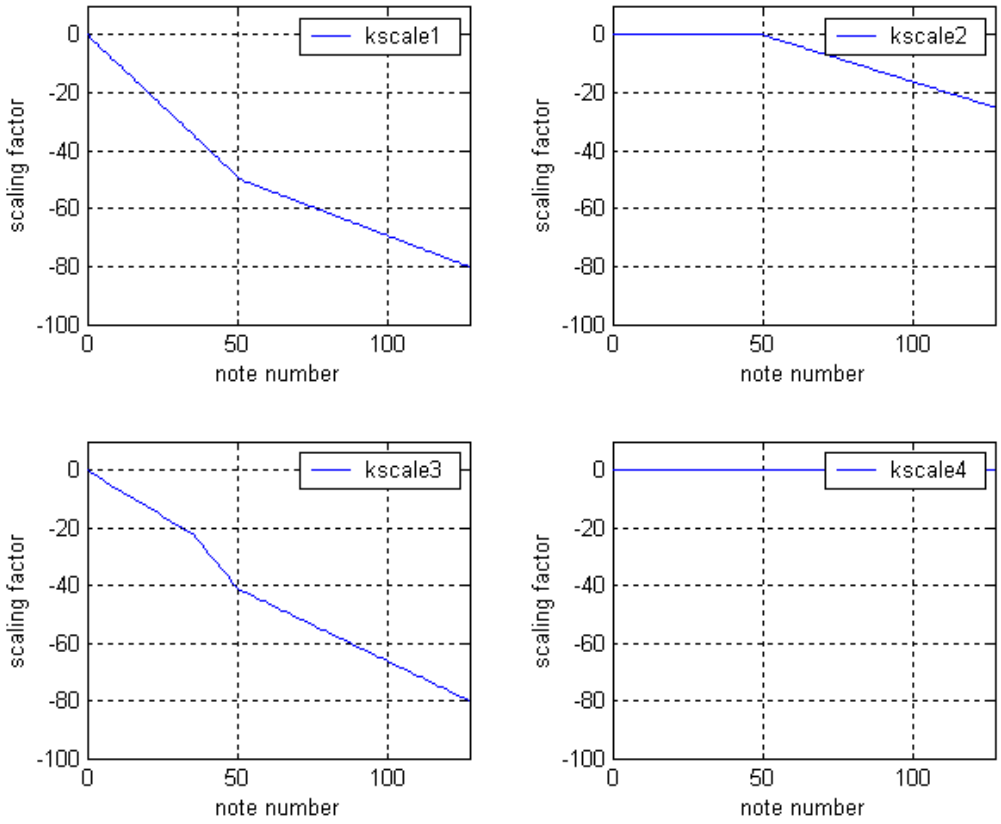

Figure 4.4: Key scaling parameter for the specific operators

For the modulation index of the three modulators, it can be seen from Figure 4.4 that it decreases with expanding note number. Here, the modulation index I was adapted to the lowest note number. In that case, the rate of partials is the greatest.

4.5 The Envelopes

Envelopes are used to model the time variant amplitude devolution of real instruments. The output of the envelope generator controls the amplitude of the operator so that the instrument produces a fixed waveform enclosed in the envelope. For the piano, the shape of the envelope curve decays exponentially, which can be realized by the following relation:

$$e(n) = r \cdot e(n-1) \qquad with\ r < 1 \tag{4.38}$$

4.5.1 Release part of the envelope

As the string of a real piano is being damped after the release of the stroke key, the envelope generator has to model this behavior. This can be done by modifying the factor *r* in (4.38) to get a fast decaying curve for the release part. In the MIDI Standard, the duration of a tone is represented by the MIDI messages "note on" and "note off".

Thus, the envelope generator has to be controlled by these messages to produce the correct shape. This results in

$$e_k(n) = \begin{cases} r_{1k} \cdot e_k(n-1) \text{.....Note On} \\ r_{2k} \cdot e_k(n-1) \text{.....Note Off} \end{cases}, \tag{4.39}$$

where r_1 and r_2 have to be less then 1. The initial condition for $e_k(n)$ is given by:

$$e_k(n)\Big|_{n=0} = 1. \tag{4.40}$$

Figure 4.5 depicts the envelope curves for each operator, where the duration of the synthesized piano tone is two seconds plus the little overhead of the release phase[1]. An informal listening test has shown that the use of these parameter settings yields the best synthesis of the piano tone.

4.5.2 Envelope scaling

Considering a real piano shows that the decay rate is a function of the note number. Therefore the higher the note number is, the faster the tone decays. Thus, a further control parameter for the envelope generators has to be introduced to simulate that property. Now (4.39) can be expanded by introducing the envelope-scaling factor $w(m)$.

This leads to the following equation:

$$e_k(n) = \begin{cases} w_k(m) \cdot r_{1k} \cdot e_k(n-1) \text{.....Note On} \\ r_{2k} \cdot e_k(n-1) \text{.....Note Off} \end{cases} \qquad m = 0...127 \tag{4.41}$$

It can be seen from (4.41) that only the part for the Note On event is influenced by the scaling factor $w_k(m)$. The final decay remains constant over the whole range of note numbers.

[1] To reduce the computational costs, the maximum length of a piano tone is limited to four seconds.

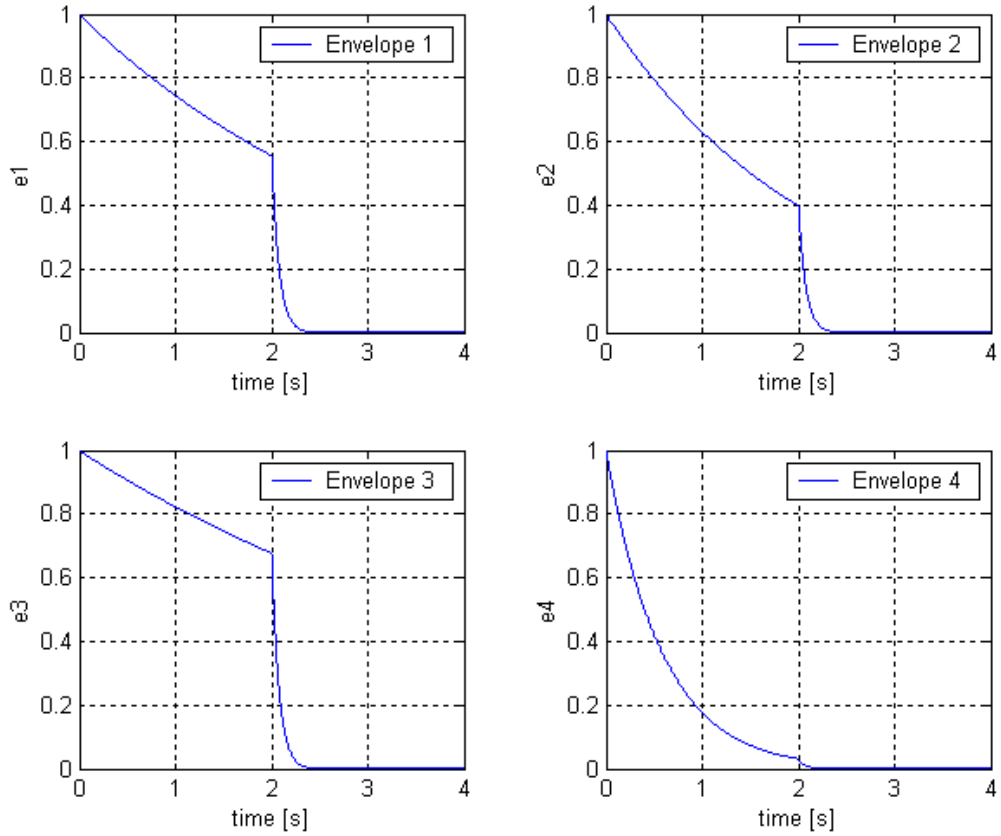

Figure 4.5: Envelope shape as a function of time for each operator of the FM instrument

4.5.3 Velocity sensitivity

Almost all digital synthesizers have the feature to adjust the velocity sensitivity in a way to narrow the dynamic level of the rendered sound down. This can be realized by the following equation.

$$s_k = \frac{100 - l_k}{200} + \left(\frac{l_k}{12{,}8 \cdot 10^3} \right) v \tag{4.42}$$

$$\text{with } \; l = 0...100; \; v = 0...127$$

In (4.42), s is the new initial condition for the envelope generators, l is the sensitivity value, and v is the velocity value. It can be seen that an increasing l yields a widening of the dynamic range of s. Thus, (4.40) changes to:

$$e_k(n)\Big|_{n=0} = s_k \tag{4.43}$$

4.6 The entire algorithm

Now all parameters for the determination of the entire algorithm of a piano FM instrument are given. Table 4.6 gives an overview of the used parameter.

Operators	Ratio	Output Level	Offset [Hz]
Op1	5,00240	92	0,00
Op2	1,00030	38	0,26
Op3	11,0070	40	0,00
Op4	1,00020	78	0,13

Table 4.7: Parameters for the individual operators

Figure 4.6 depicts the interconnection of the operators. Furthermore, the envelope generators as well as their input parameters are shown in more detail.

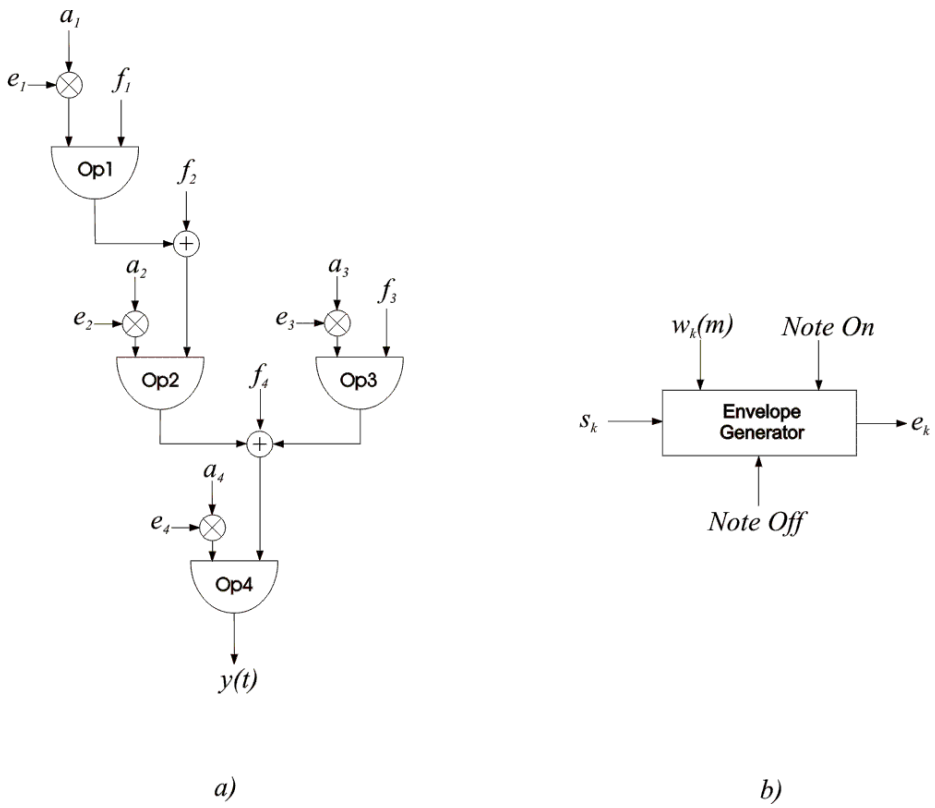

Figure 4.6: a) Interconnection of 4 operators; b) Envelope generator with its input parameters

The associated equation is as follows:

$$y(t) = a_4 e_4 \sin\left\{2\pi f_4 t + a_3 e_3 \sin(2\pi f_3 t) + a_2 e_2 \sin\left[2\pi f_2 t + a_1 e_1 \sin(2\pi f_1 t)\right]\right\} \qquad (4.44)$$

5 Sampling synthesis

In this chapter, the synthesis of real piano sounds by means of sampling will be discussed. Sampling synthesis itself can be implemented efficiently. Thus, all the commercially available digital pianos employ this technique. However, the required amount of memory storage is huge. To overcome this problem, techniques like looping and pitch shifting are employed. Furthermore, techniques to modify the timbre of the synthesized tones have to be used.

5.1 Looping

The difficulty at looping a sampled tone is to find the correct loop points. Normally, looping is applied to the steady state part of the tone. After the attack phase, a number of instrument families own the property of remaining relatively constant in respect to their amplitude and pitch. Thus, locating the starting and ending loop points is straightforward. In the case of synthesizing the sound of a piano, after a short attack phase the tone decays exponentially. Therefore, for this class of instruments it is rather difficult to define loop points. Though it is possible to perform looping near the end of the original sample. However, this again increases the amount of required memory. As a result, the sample has to be looped shortly after the attack part.

5.1.1 Equalization of the magnitude

Since the tone of the piano decays exponentially, the starting point and the end point of the loop differ in their amplitude values. To overcome this problem, the devolution of the amplitude has to be equalized. This can be performed with the aid of the Hilbert transform. For a real input sequence x it offers a complex signal y. The amplitude of y represents the instantaneous amplitude (amplitude envelope) of x. However, as there are still too many ripples in the devolution of the amplitude envelope, a filtering by a moving average filter has to be performed. By now, the looped region of the sample has to be weighted by the inverse of the filtered version of y.

All these operations can be conducted quite easily with the aid of Matlab™ and thus, they do not have to be implemented in the target application.

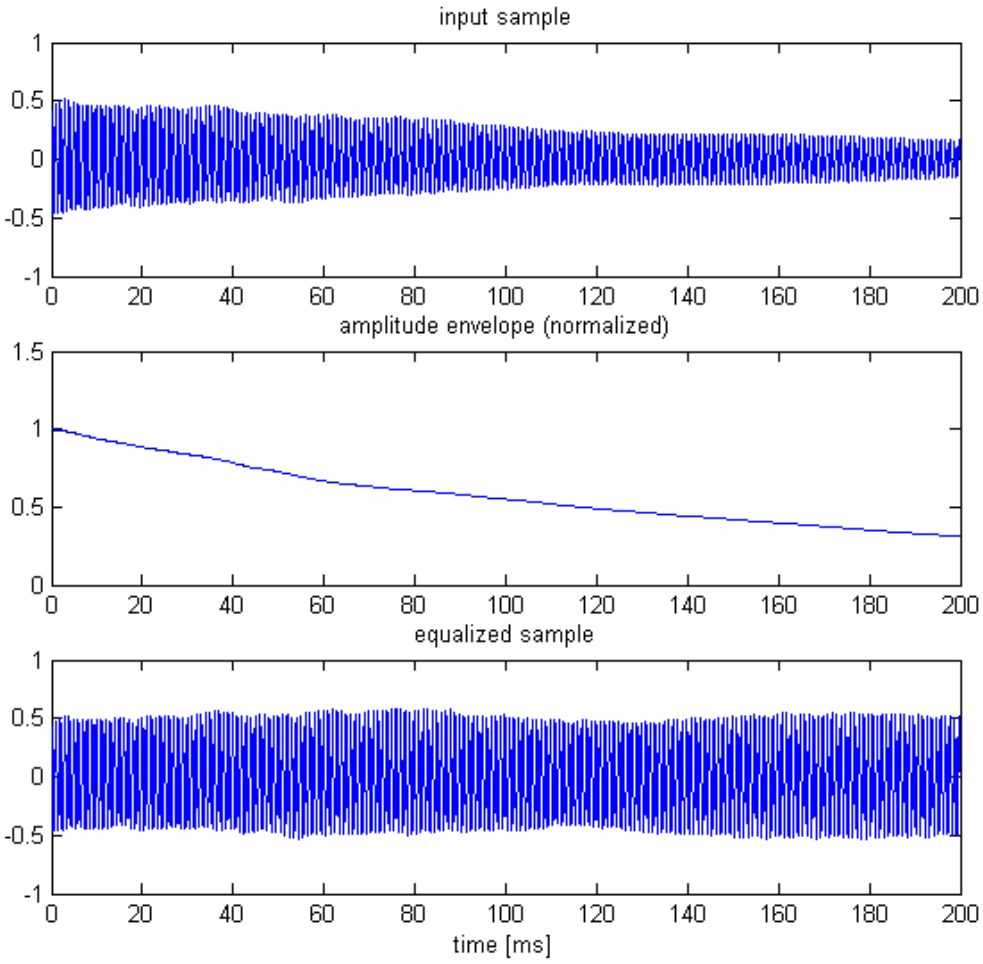

Figure 5.1: Equalization of the amplitude values of the input sample

5.1.2 Length of the looped region

Usually, the length of the loop is taken to be a multiple of the fundamental frequency of the recorded instrument tone. If the loop is too short, the synthesized tone sounds a little "sterile". Otherwise, the length of the loop may not be too long, since the magnitudes of the spectral components of the piano tone vary with time. Thus, the looping points are perceived by the listener as beats, even if the magnitude of the looped region has been equalized.

In general, the beginning as well as the end points of a loop are spliced together at a common sampling point. In most instances, this results in a click or pop at the splicing point, unless the beginning and ending points are well matched in regard to the amplitude. These artifacts may be reduced by crossfading the loop regions at their boundaries. In the case that none of the above mentioned techniques produce a smooth loop, the method of bi-directional looping can be brought into account.

Thereby the direction of playing back the loop is alternated. The loop, which is played back in forward direction, can be layered on top of the loop, which is played back in inverse direction. Thus, some discontinuities may be masked [14].

5.2 Pitch shifting

To further reduce the amount of required memory, one can take pitch shifting into account. Thus, only a few tones, e.g., every third or fourth semitone of the real piano have to be sampled. The pitch of these samples is shifted up and down by a specific value to cover the whole range of piano tones.

In most samplers, pitch shifting is performed by a simple time-domain technique. This is essentially the same pitch variation technique as used in wavetable-lookup synthesis described in 4.2.3. A side effect of this method is the alteration of the sounds duration, depending on the key that has been depressed.

However, pitch shifting may be done in the frequency domain as well, with the main advantage of preserving the original duration of the sound. The "phase vocoder" theory (e.g., [27, 28]) gives a lot of theoretical framework to these techniques. A profound consideration lies beyond the scope of this thesis.

5.2.1 The accuracy of the frequency shift

Due to the fact that humans auditory perception is capable to differ a change in pitch of about one percent of a semitone, the accuracy of the pitch shifting method has to be as follows:

$$\left.\begin{aligned} f_{out} &= \varphi f_{wav} \\ \Delta f_{out} &= \Delta\varphi f_{wav} \end{aligned}\right\} \frac{\Delta f_{out}}{f_{out}} = \frac{\Delta\varphi}{\varphi} \tag{5.1}$$

$$\Delta\varphi = 2^{-N} \tag{5.2}$$

Solving (5.1) for N yields the following equation:

$$N > -\log_2\left[\varphi \cdot (2^{1/1200} - 1)\right], \tag{5.3}$$

where

f_{wav}… frequency of the signal in the wavetable
f_{out}… desired frequency
φ… phase increment
$\Delta\varphi$… smallest value of the phase increment
N… number of bits for the fractional part

By the limitation of the pitch shift to the amount of one octave, at least a number of 12 bits should be used for the fractional part of the phase increment.

5.3 Modification of the timbre

In a real piano, the timbre of a tone changes with time. Furthermore, it varies concerning the velocity of the piano hammer. Hence, these properties should be taken into account by the synthesis of piano tones. This problem can partly be solved by recording several piano tones for different velocity values and by storing them in wavetables. However, the main drawback of this method is the increase of required memory.

A different approach to model the changes of the timbre can be given by means of time-varying filtering of the output signal. Therefore, the samples, which are used for the synthesis, have to be recorded at a maximum level of velocity. In this case, it is ensured that most of the spectral components of the real piano tone are present. For the filtering of the output signal, a first-order high frequency shelving filter is used. The transfer function of this filter is given by:

$$H(z) = 1 + \frac{H_0}{2}\left[1 - A(z)\right] \tag{5.4}$$

with the first order allpass

$$A(z) = \frac{z^{-1} + a}{1 + az^{-1}}. \tag{5.5}$$

The block diagram in Figure 5.2 shows the first order low/high-frequency shelving filter, which leads to the following difference equation:

$$y_1(n) = a_{B/C}x(n) + x(n-1) - a_{B/C}y_1(n-1) \tag{5.6}$$

$$y(n) = \frac{H_0}{2}\left[x(n) - y_1(n)\right] + x(n). \tag{5.7}$$

The gain G in dB can be adjusted by the parameter

$$H_0 = V_0 - 1, \qquad \text{with} \quad V_0 = 10^{G/20}. \tag{5.8}$$

It is used to simulate the timbral behavior of the piano for different velocity levels. This means that after adjusting the cutoff frequency of the shelving filter, G has to be decreased if the key-velocity increases.

The cutoff parameter a_C can be calculated as

$$a_C = \frac{V_0 \tan(\pi f_c / f_s) - 1}{V_0 \tan(\pi f_c / f_s) + 1}. \tag{5.9}$$

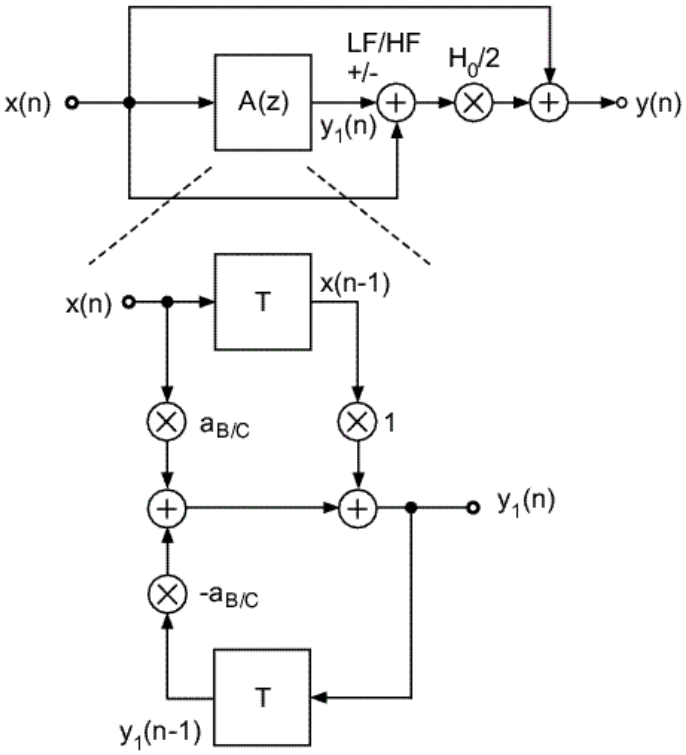

Figure 5.2: First-order low / high-frequency shelving filter

The time-varying timbral behavior can be modeled by a variation of *G* as well, but with the slight difference that *G* is continuously decremented over the duration of the tone.

5.4 The envelope

Since the envelope generator for the sampling synthesis is approximately identically to that of the FM synthesis described in 4.5, only that extra edge will be pointed out in this thesis.

The samples used in the sampling synthesis algorithm are recordings of real piano tones and therefore, their amplitudes already decay exponentially.

Thus, the envelope generator has to be bypassed until the starting loop point is reached. Otherwise, the amplitude of the generated output signal would decay too rapidly.

5.5 The entire algorithm

Figure 5.3 depicts the entire structure of the sampling instrument.

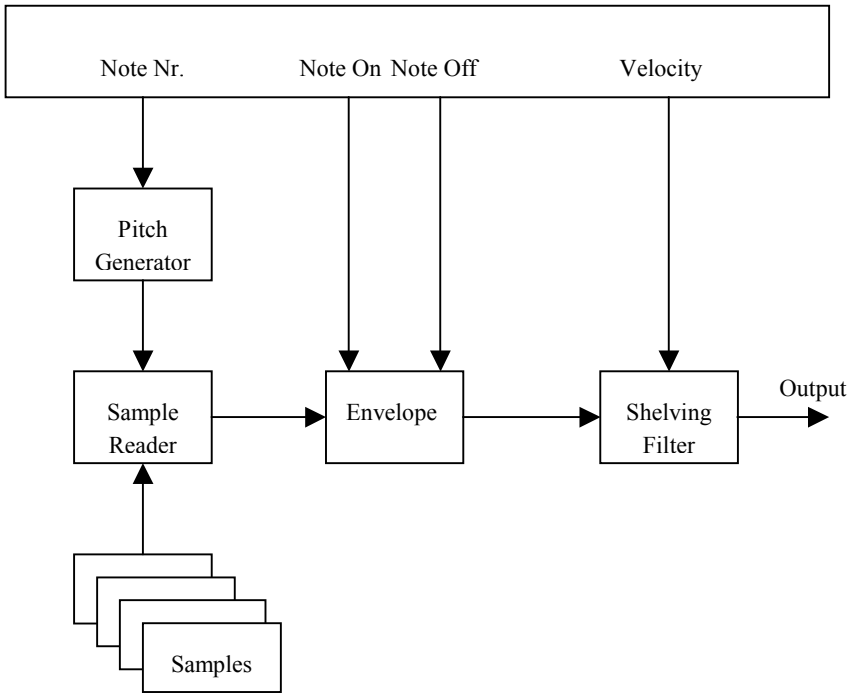

Figure 5.3: Structure of sampling algorithm

6 Digital waveguide synthesis

In this chapter, the theory of digital waveguides will be described in terms of their mathematical background. After haven given an overview of the ideal string, the effect of losses and dispersion will be discussed. Furthermore, the piano hammer and the soundboard of the piano are taken into account.

6.1 Waveguide model of the string

Modeling strings by means of digital waveguides was introduced by [17]. It can be performed by solving the wave equation in a general way to obtain "traveling waves" in the interior medium.

6.1.1 Modeling the ideal string

By establishing the wave equation for the ideal string, some simplifications have to be made:

- The length of the string is assumed infinite.
- The string tension and the linear mass density of the string have to be homogeneous
- The displacement of the string has to be small in relation to the length of the string

Furthermore, only one transversal polarization of the string is taken into account. The outcome of this is the wave equation for the ideal (lossless, linear, flexible) vibrating string which can be expressed as

$$K\frac{\partial^2 y}{\partial x^2} = \varepsilon\frac{\partial^2 y}{\partial t^2}, \tag{6.1}$$

where K is the string tension, ε the linear mass density, and y the displacement of the string [17]. (6.1) is applicable to any perfectly elastic medium with a one- dimensional vibratory motion.

The wave equation can be solved by a function of the form

$$y(x,t) = y_r(x-ct) + y_l(x+ct) \tag{6.2}$$

where

$$c = \sqrt{\frac{K}{\varepsilon}}. \tag{6.3}$$

This is d'Alembert's solution to the wave equation. It represents the superposition of two traveling wave components; one moving to the right, denoted as $y_r(x\text{-}ct)$, the other one to the left, denoted as $y_l(x+ct)$. Thereby, y_r and y_l are arbitrary twice-differentiable functions. For a digital representation of the traveling wave components, a discretization of these functions has to be performed.

This can be done by first changing the variables

$$x \to x_m = mX$$
$$t \to t_n = nT$$

where T is the temporal sampling interval, and X is the corresponding increment in space. The new variables are related by

$$c = \frac{X}{T}. \tag{6.4}$$

If the sampling is done in a way that the traveling waves move one spatial sampling interval during one time-instant, it will lead to the digital waveguide model of the ideal string [17]:

$$y(t_n, x_m) = y^+(n-m) + y^-(n+m) \tag{6.5}$$

In this notation, the "+" superscript denotes the traveling wave component going to right direction and the "-" superscript the component moving leftwards.

As a result, the digital implementation of the waveguide can be performed by the use of two parallel delay lines. The upper delay line represents the wave components traveling rightwards, whereas the lower one is defined to represent the left moving traveling wave component. These delay lines are pictured in Figure 6.1.

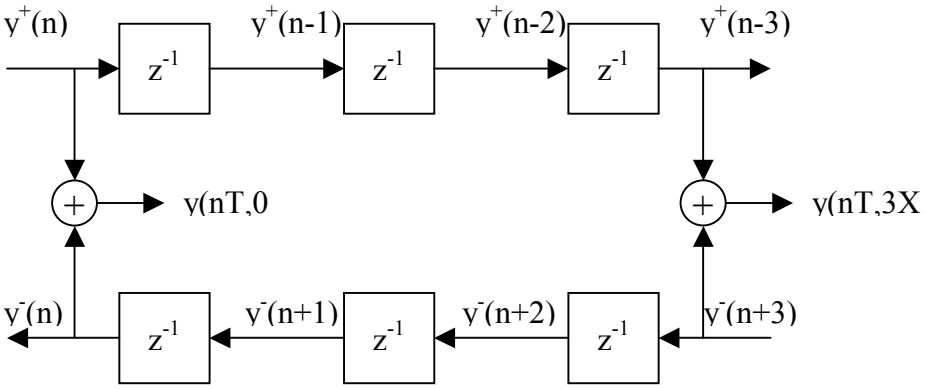

Figure 6.1: Digital simulation of the ideal, lossless waveguide after [17]

The output signal of the delay line is obtained by simply adding the delay line variables at position m along the delay-line pair.

Because of the linear behavior of the digital waveguide, other variables can be used instead of the transversal displacement as well. These variables can be derived by integration or differentiation of y. However, the most important variables are the transversal velocity v as well as the force f, since they are proportional to each other. The digitized, traveling force-wave components are defined by

$$\begin{aligned} f^+(n) &= Rv^+(n) \\ f^-(n) &= -Rv^-(n) \end{aligned}. \tag{6.6}$$

Thereby, R represents the wave impedance (also called the characteristic impedance), denoted by

$$R = \sqrt{K\varepsilon} \ . \tag{6.7}$$

6.1.2 Frequency-dependent losses

In real vibrating objects, losses that increase with frequency are always present. The largest losses in stringed instruments occur at the bridge, especially at frequencies, which couple to body resonances. Furthermore, losses due to air drag increase monotonically with frequency [17].

These losses have to be added to the digital waveguide model in terms of several frequency-dependent gain factors, which are inserted between every unit delay.

Because of the fact, that the digital representation is linear and time-invariant and considering that the medium parameters are constant as well, the gain factors can be commuted for every unobserved portion of the delay line. Figure 6.2 depicts the modified digital waveguide.

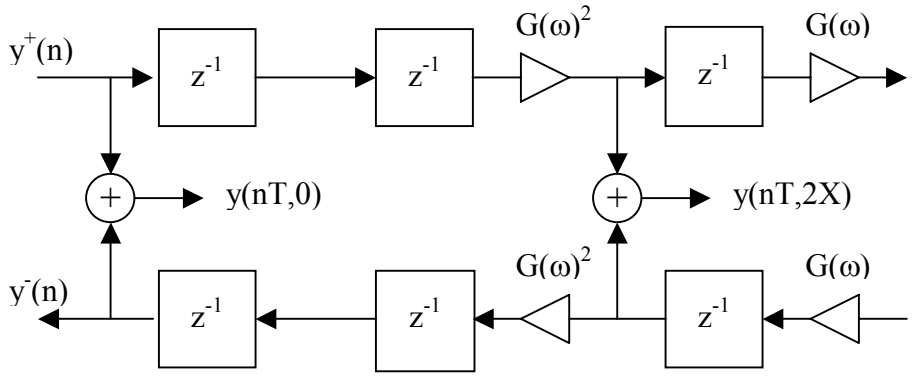

Figure 6.2: Linear digital waveguide with commuted losses

For a computationally efficient implementation, the frequency-dependent losses have to be approximated by a digital filter. Since implementing a frequency-dependent delay is undesired, the filter has to be a linear-phased FIR filter [29].

6.1.3 Waveguide with dispersion

To further approximate the behavior of the real string, its stiffness has to be taken into account. Stiffness introduces a restoring force to the fourth derivative of the string displacement. Therefore the wave propagation speed is no longer constant, but it depends on the frequency. This effect is present in every physical string and is called dispersion. For the altered wave equation, see 2.2.2 of this thesis.

Dispersion in vibrating strings results in an inharmonic overtone series, as already mentioned in. Dispersion can be modeled by the use of allpass filters, which have a non-uniform delay versus frequency.

If the string parameters are known, then the frequency of the k^{th} stretched partial can be computed as

$$f_k = kf_0\sqrt{1+Bk^2}\ , \tag{6.8}$$

where the value of the inharmonicity coefficient B depends on string parameters [4]. A design method for group-delay filters is given in [30]. Scalcon and Rocchesso [31] studied the dependence of the bandwidth of perceived inharmonicity on the fundamental frequency by performing listening tests with decaying piano tones.

6.1.4 The termination of the string

The simplest case is the rigid termination of the string, corresponding to infinite terminating impedance. In that case, the string cannot move at all at the termination point. Thus, the traveling wave components are totally reflected. As a result, considering the case of the lossless string, the generated tone would never decay.

For a non-ideal termination of the vibrating string, where the string is terminated by a finite impedance, the absolute value of the reflection coefficient will be somewhat lower than 1. In this case, the traveling wave components in the delay lines are multiplied by a constant value every time they pass the termination. As a result, they decay exponentially.

6.1.5 The non-ideal string

The model for a non-ideal string can be given by taking into account all the previously mentioned losses and the dispersion. As a fact of linearity and time-invariance, all these losses can be lumped to one point. The model of the non-ideal string is depicted in Figure 6.3.

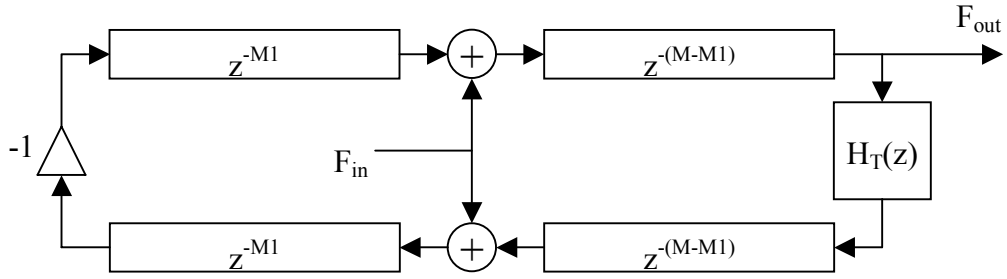

Figure 6.3: Model of the non-ideal string; In this case, the input signal is a force signal

The filter in this figure contains the losses as well as the dispersion of the string on the one hand and the terminations on the other.

6.1.6 Coupled strings

Coupling between strings leads to beating and two-stage decay. These effects are caused by two or three slightly detuned strings, which are sounding together when a single piano key is depressed (except for the lowest octave of the piano). Furthermore, the bridge and the action of the soundboard also introduce coupling effects.

The simplest way to simulate beating and two-stage decay is to use two digital waveguides in parallel. When their pitches are slightly different, beating will appear in the sound. Otherwise, when their decay rates differ, the two-stage decay will appear.

Another approach, taken by [32], couples two strings to the same termination and lumps all the losses to the bridge impedance. This comes from the assumption that all the losses come from the bridge, which is a rough approximation. By using one coupling filter H_b for calculating the velocity of the bridge, the reflected velocity waves of the string can be computed by subtracting the incoming velocity waves from the bridge velocity. Thus,

$$\begin{aligned} v_1^-(t) &= v_b(t) - v_1^+(t) \\ v_2^-(t) &= v_b(t) - v_2^+(t) \end{aligned} \tag{6.9}$$

The velocity of the bridge in the Laplace domain is given by

$$V_b(s) = H_b(s)\left[R_1 V_1^+(s) + R_2 V_2^+(s) \right] \tag{6.10}$$

R_1 and R_2 are the wave impedances of the strings, and

$$H_b(s) = \frac{2}{R_b(s) + R_1 + R_2} \tag{6.11}$$

Figure 6.4 shows the structure of the coupling system.

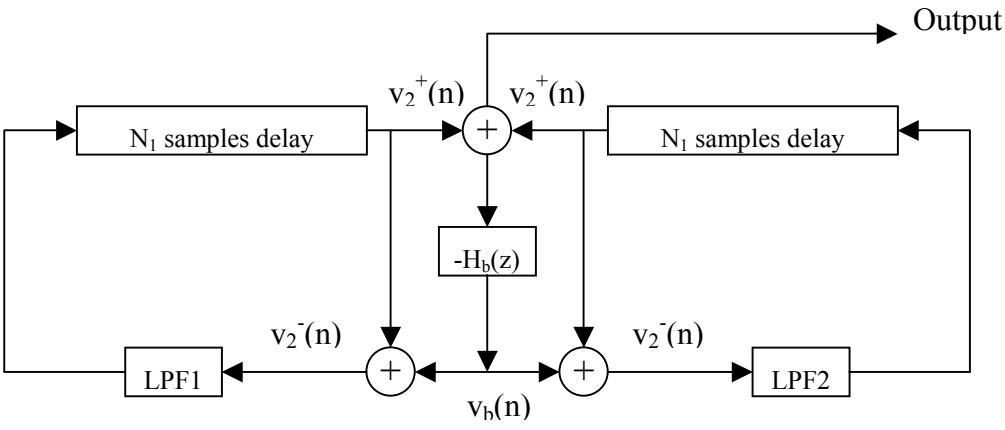

Figure 6.4: Linear coupling of two equal-impedance strings after [32]

6.2 Modeling the hammer

In this subchapter, two approaches of modeling the piano hammer will be described. The first model is the so-called “wave digital hammer” introduced by [33]. The second one is a linear model for the piano hammer introduced by [34].

A description of further models is given in (e.g., [5-7]).

6.2.1 The wave digital hammer

For the physical model, the hammer is considered as a mass connected to a nonlinear spring where the spring represents the felt portion of the hammer [5]. The nonlinear behavior can be traced back to the nonlinear compression characteristic of the felt.

In deriving a digital representation of the hammer model, first a wave decomposition of the linear mass and spring system is performed.

The mass and spring system

First, a definition of the equations of motion for the mass and spring system, depicted in Figure 6.5 has to be found.

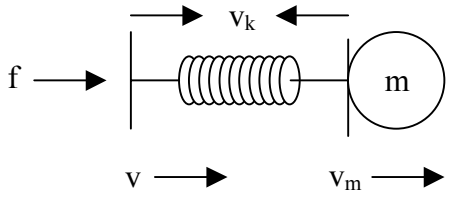

Figure 6.5: mass and spring force diagram

The system is driven at the spring end by an external force. For the mass, an initial velocity at time zero is assumed. The driving point force f is equally applied to the spring and to the mass.

$$f = f_k = f_m \tag{6.12}$$

For the velocities the following relation can be applied:

$$v = v_k + v_m \tag{6.13}$$

The force equation for the ideal spring is:

$$f_k(t) = kx_k(t) \quad \Rightarrow \quad \frac{df_k(t)}{dt} = kv_k(t)\,. \tag{6.14}$$

Thereby, $f_k(t)$ is the force applied on the spring, $x_k(t)$ is the compression distance of the spring, $v_k(t)$ is the velocity of compression, and k is the stiffness constant.

By assuming that there is no initial force on the spring, the Laplace transform of $f_k(t)$ is

$$F_k(s) = (k/s)V_k(s) \quad \Rightarrow \quad V_k(s) = (s/k)F_k(s) \tag{6.15}$$

The force equation for the mass is

$$f_m(t) = m\frac{dv_m(t)}{dt} - mv_0\delta(t). \tag{6.16}$$

Hence, the Laplace transform of the force on the mass is

$$F_m(s) = msV_m(s) - mv_0 \quad \Rightarrow \quad V_m(s) = \frac{1}{ms}F_m(s) + \frac{v_0}{s} \tag{6.17}$$

In (6.16), the term $mv_0\delta(t)$ represents an initial impulse, which sets the mass in motion at velocity v_0. Combining (6.15) and (6.17) leads to the driving point admittance relation for the mass and spring system.

$$V(s) = \left(\frac{s^2 + k/m}{ks}\right)F(s) + \frac{v_0}{s} \tag{6.18}$$

Traveling wave decomposition of the lumped system

Since it is possible to define wave variables for lumped impedances [35], the driving point force can be considered as the superposition of a force wave traveling into the impedance, which is denoted as f_{in}, and a force wave traveling out of the impedance, denoted as f_{out}. Similarly, the driving point velocity can be defined by $v = v_{in} + v_{out}$.

Furthermore, there is a wave impedance relation between the force and the velocity waves traveling in the same direction. This relation can be defined by the reference impedance R_h.

$$F_{in} = R_h V_{in} \qquad F_{out} = -R_h V_{out} \tag{6.19}$$

By changing the variables from F and V to V_{in} and V_{out}, (6.18) changes to

$$V_{out} = \frac{s^2 - (k/R_h)s + k/m}{s^2 + (k/R_h)s + k/m} V_{in} + \frac{v_0 k/R_h}{s^2 + (k/R_h)s + k/m} \tag{6.20}$$

For a digital representation the Bilinear transform

$$s \to \alpha \frac{1 - z^{-1}}{1 + z^{-1}} \tag{6.21}$$

is applied to (6.20) which yields

$$V_{out}(z) = z^{-1} H(z) V_{in}(z) + G(z) v_0 \tag{6.22}$$

where,

$$H(z) = \frac{a_0 + z^{-1}}{1 + a_0 z^{-1}}, \tag{6.23}$$

and

$$G(z) = \left(\frac{1}{2\alpha T} \right) \frac{1 + 2z^{-1} + z^{-2}}{1 + a_0 z^{-1}}. \tag{6.24}$$

Scattering junction connection

For connecting the mass and spring system to a string, a lossless 3-port scattering junction is introduced. At the junction point, the velocity of both sides of the string and the driving point velocity of the spring has to be equal to the junction velocity. Furthermore, the sum of all the forces at the junction point has to be zero, because it is assumed to be massless. These conditions lead to the following lossless scattering equations:

$$\begin{aligned} v_j &= \frac{2(R_0 v_1^{+} + R_0 v_2^{+} + R_h v_h^{+})}{2R_0 + R_h} \\ v_1^{-} &= v_j - v_1^{+} \\ v_2^{-} &= v_j - v_2^{+} \\ v_h^{-} &= v_j - v_h^{+} \end{aligned} \tag{6.25}$$

Thereby, v_j is the velocity of the junction and R_0 is the characteristic wave impedance. These equations say that, as a wave is entering the junction, some portion of the wave is reflected and travel back where it comes from, while the rest of the wave component is distributed among the other outgoing waves. Figure 6.6 illustrates the attachment of the mass and spring system to a waveguide.

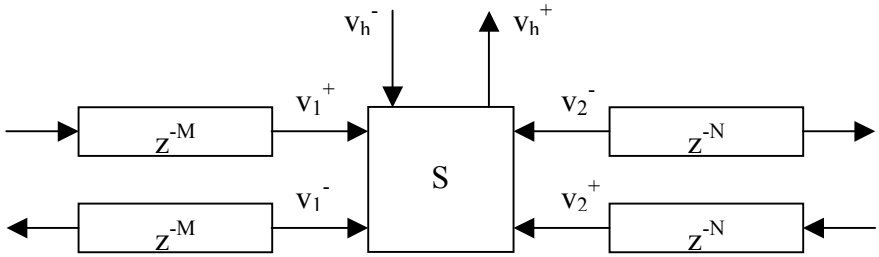

Figure 6.6: Attaching the wave digital hammer

The nonlinear behavior of the felt

Until now, the spring constant was considered linear. By modeling the nonlinear behavior of the hammer felt, the spring constant has to depend on the compression distance x_k of the felt, which can be defined as

$$x_k = (R_h / k)(v_h^- - v_h^+) = \frac{1-a_0}{2\alpha}\left(v_h^- - v_h^+\right). \tag{6.26}$$

Now the lossless wave digital hammer system is complete and is depicted in Figure 6.7.

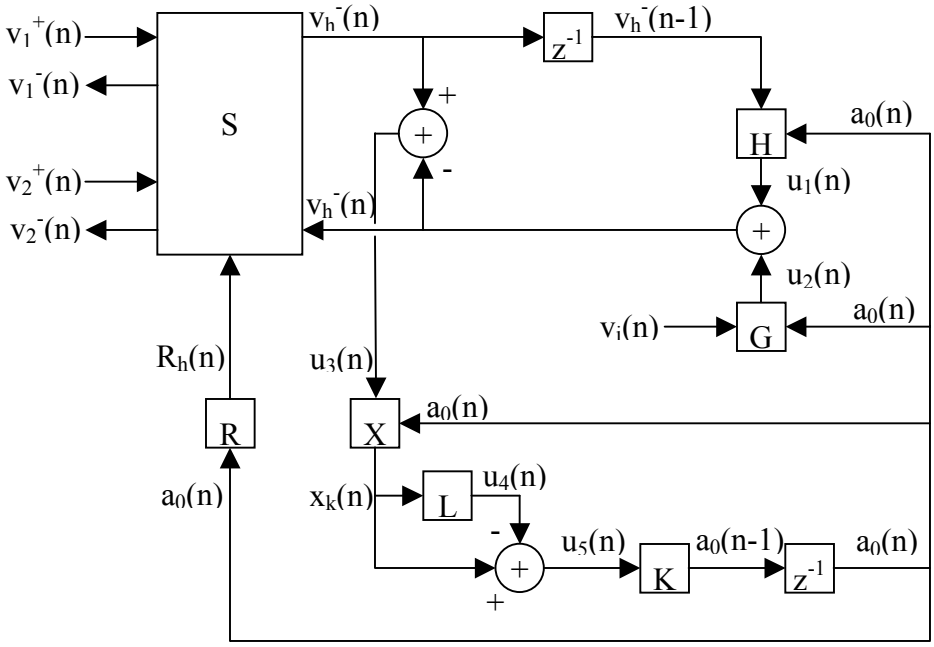

Figure 6.7: The wave digital hammer after [33]

It is obvious that the upper half of the model corresponds to the linear mass and spring system depicted in [33]. The lower half of the digital hammer system acts as the nonlinear and lossy part.

- *X* computes the actual felt compression and thereby indexes the look-up table *K*, which contains the stiffness coefficients.
- *R* computes the effective wave impedance for the hammer model.
- *L* is used for modeling the hysteretic loss in the felt by offsetting the pointer in table *K* corresponding to the velocity of the felt compression.

6.2.2 Linear model for the piano hammer

In the case of the hammer-string interaction, waves from the agraffe return and interact with the hammer before it leaves the string. Reflected waves from the bridge end take a much longer time to reach the striking position of the string. Thus, only at the very highest notes an interaction between the hammer and the waves coming from the bridge end is possible. Therefore, a one-sided termination at the agraffe end is used to formulate the string impedance.

Impedance of the terminated ideal string

The following equation represents the impedance of the terminated ideal string.

$$R_s = \frac{F_s}{V} = \frac{2R_0}{1-e^{-sT}}, \tag{6.27}$$

where T is the time the force impulse takes to travel to the agraffe and back to striking position, and R_0 is the characteristic impedance. V and F_s are the Laplace transforms of the velocity and the force at the driving point.

Impedance of the ideal linear hammer

The hammer is a mass and spring system, where the spring represents the felt portion of the hammer. The impedance relation for the hammer is:

$$F_H = R_H\left(V - \frac{v_0}{s}\right) \qquad \text{where} \quad R_H = \frac{ks}{s^2 + k/m}. \tag{6.28}$$

The initial striking velocity is represented by v_0/s .

During the hammer-string interaction the velocity of the string is equal to the velocity of the spring end of the hammer model. The force on the string is equal and opposite to the force on the spring.

$$F_S = -F_H = -R_H\left(V - \frac{v_0}{s}\right) = -R_H\left(\frac{F_S}{R_S} - \frac{v_0}{s}\right) \tag{6.29}$$

Therefore, F_S is given by

$$F_S = \left(\frac{R_H R_S}{R_H + R_S}\right)\frac{v_0}{s} = \left(R_H \left\| R_S\right.\right)\frac{v_0}{s}. \tag{6.30}$$

Now H_∞ can be defined as the transfer function from the initial striking velocity step to the force experienced by the string and, equivalently, by the hammer felt.

$$H_\infty = R_H \left\| R_S = \left(\frac{ks}{s^2 + k/m}\right)\right\| 2R_0 = \frac{ks}{s^2 + \frac{k}{2R_0}s + \frac{k}{m}} \tag{6.31}$$

(6.31) represents a damped second order system.

For the case of the terminated string, (6.31) alters to

$$H_T = R_H \left\| R_S = \left(\frac{ks}{s^2 + k/m}\right)\right\| \left(\frac{2R_0}{1 - e^{-sT}}\right) = \frac{ks}{s^2 + \frac{k}{2R_0}\left(1 - e^{-sT}\right)s + \frac{k}{m}} \tag{6.32}$$

With (6.31) and (6.32) it is possible to formulate a recursive relationship between H_∞ and H_T.

$$H_T = \frac{H_\infty}{1 - e^{-sT}H_\infty} \tag{6.33}$$

Hence, it can be seen that the exact nature of the hammer impedance R_H is not present in the above equation.

Because of the fact that H_∞ can be treated as a differentiated lowpass filter, the step-driven hammer system may be transformed to an impulse-driven system.

$$H_\infty = sL_s = s\frac{k}{s^2 + \frac{k}{2R_0}s + \frac{k}{m}} \tag{6.34}$$

Figure 6.8 depicts the resultant impulse-driven hammer filter.

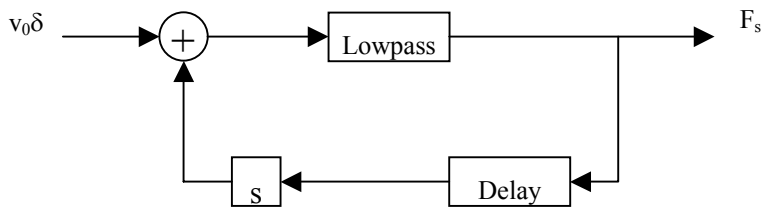

Figure 6.8: Impulse-driven recursive hammer filter after [34]

For a digital implementation, the model has to be transformed to the z-domain by the bilinear transformation.

The filter coefficients depend on k, R_0 and m. Thus, a recalculation has to be performed every time a parameter changes. R_0 and m changes whenever a new key is depressed; k changes with impact velocity.

6.3 Modeling the soundboard

Large vibrating objects, such as the soundboard and the enclosure of a piano, have many resonant modes in the range of the human hearing. Thus, high order filters have to be used to model this resonating system. The computationally most efficient method, concerning the modeling of the soundboard and the enclosure of the piano, is the commuted piano model introduced by [34, 36]. Thereby, the resonating system is commuted with the digital waveguide and the hammer model. Thus, the soundboard of the piano does not have to be implemented as a high order digital filter, but as a wavetable (containing the impulse response data of the soundboard), whose content is simply "played" into the digital waveguide.

[34] introduced a further method for the modeling of the soundboard. Since the idealized soundboard should have a smooth spectral response, the sampled impulse response of the soundboard can be replaced by an exponentially decaying white noise. The different decay rates of the individual partials can thereby be modeled by the use of a time-varying lowpass filter.

The resonant effect of the sustain pedal can be modeled by filtering white noise too. Beating down the "sustain pedal" leads to many resonating partials, which couple to the soundboard. Thus, filtered white noise with a slow decay rate can be used to model this behavior.

6.4 The simulation

In this subchapter, the simulation of a simplified acoustic model of a piano will be discussed. The simulation is based on the commuted piano model [36].

Since the physical modeling approach tries to simulate the structure of the instrument and not the sound itself, the piano model consists of the same parts as the real piano. The structure of the model is shown in Figure 6.9. The first step of the sound production mechanism is the excitation, which is the hammer in the case of the piano. The resultant signal propagates to the string, which determines the fundamental frequency of the tone. The periodic output signal is filtered through the radiator, covering the effect of the soundboard.

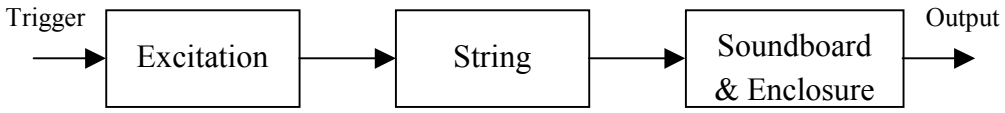

Figure 6.9: Schematic diagram of a stringed musical instrument

6.4.1 The excitation signal

An implementation of the wave digital hammer discussed in 6.2.1 would be to complex for the target application. Furthermore, since it is a nonlinear model, it does not match with the specifications of the commuted synthesis technique. This is because commutativity of system elements is only possible for linear and time-invariant elements. Thus, another approach for generating the excitation signal has to be chosen.

Because of the fact, that the hammer-string interaction basically consists of a few discrete events per hammer strike (the string has to be initially at rest), it can be approximated as one or a few discrete impulses, which are filtered to produce a smooth shape. [36].

In this simulation, these multiple interaction impulses are generated by the use of Hanning windows, where the width and the magnitude of the windows depend on the key number and the hammer velocity of the depressed key. In Figure 6.10, this method is depicted. The amount of the delay is given by the time the traveling wave components need to travel from the excitation point to the agraffe and back.

For the simulation, the used Hanning windows are limited to the amount of three. A superposition of these windows will result in the hammer force signal. The individual Hanning windows are controlled by the parameters A_k (controls the amplitude) and N_k (controls the width of the Hanning winodow).

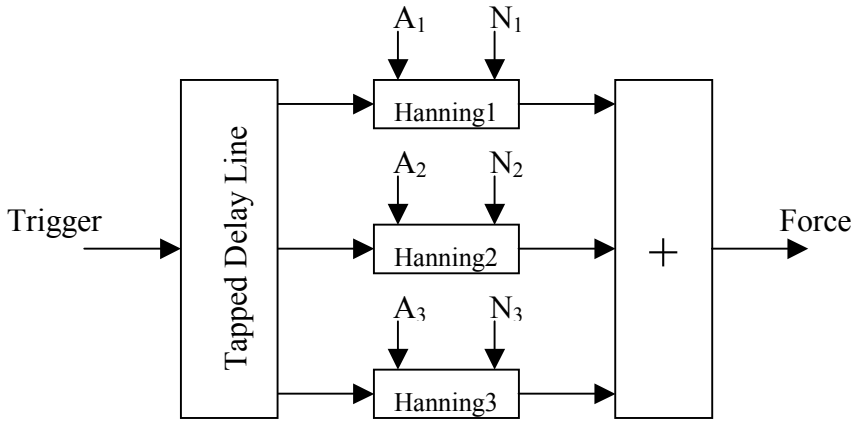

Figure 6.10: Creation of three hammer-string interaction force pulses

Good results for the notes C2 (65,4 Hz), C4 (262 Hz), and C7 (2093 Hz) as an example can be reached with the following settings. Thereby, the results of the simulated hammer forces given in [21] were approximated.

Note	Velocity 1	Velocity 2	Velocity 3	N	Delay [ms]
C2	1	0,3	0,5	60	1
C4	0,8	1	0,45	33	0,45
C7	1	1	1	20	0,01

Table 6.1: Settings for the excitation model

In this table, the *Velocity* values are the scaling factors for the amplitude of the individual Hanning windows. *N* is the number of points used for the individual Hanning windows.

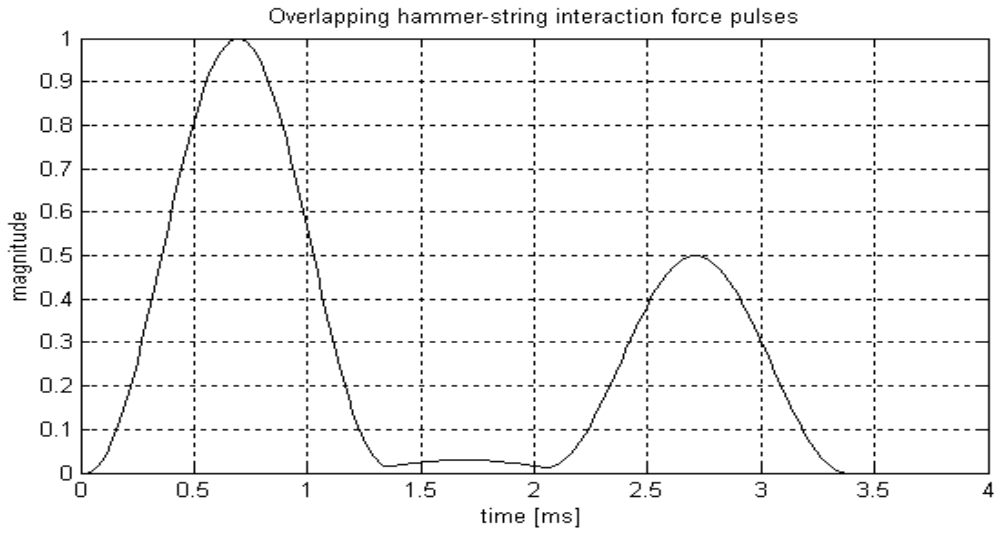

Figure 6.11: Hammer-string interaction force pulses for the note C2

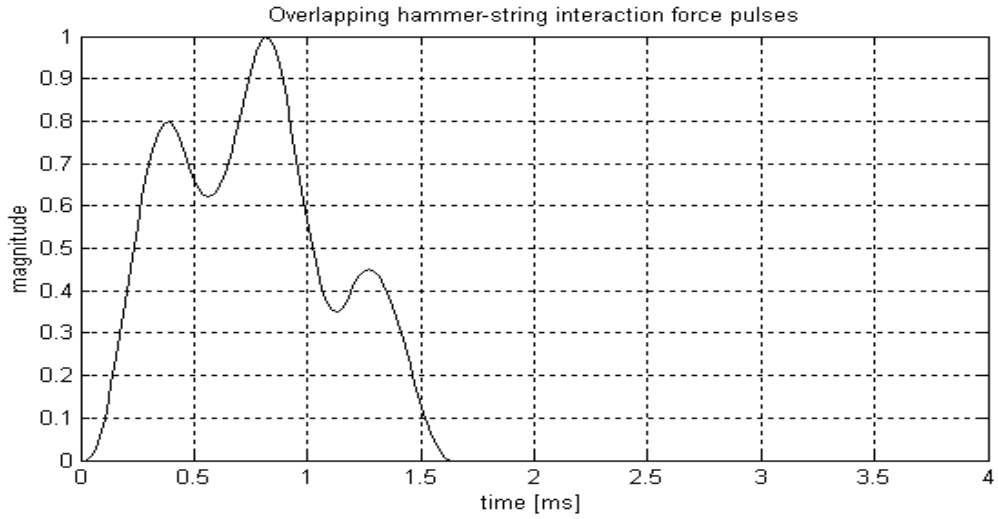

Figure 6.12: Hammer-string interaction force pulses for the note C4

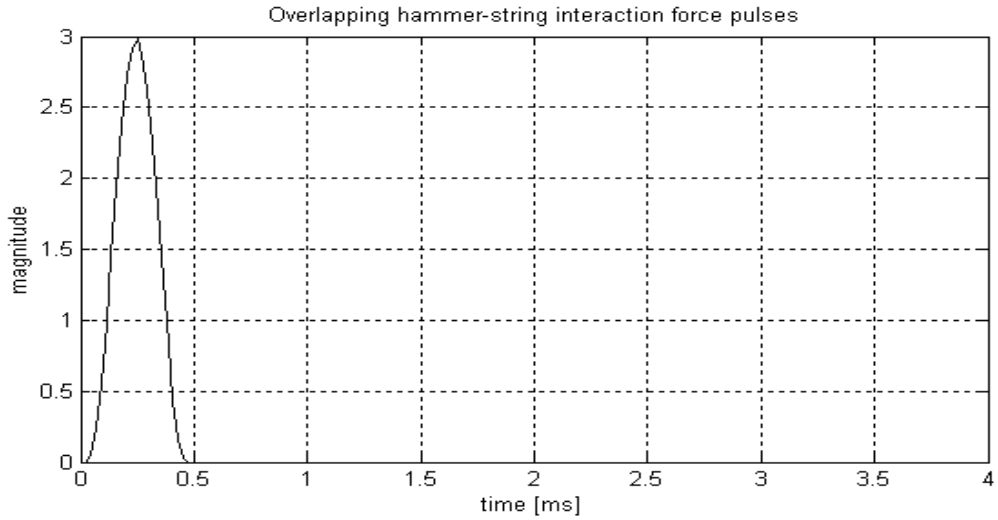

Figure 6.13: Hammer-string interaction force pulses for the note C7

6.4.2 The string interface

In a physical piano string, the hammer strikes the string between its two terminations, some distance from the agraffe and far from the bridge. This corresponds to Figure 6.3. Since linear and time-invariant elements can be commuted, the structure of this Figure can be rearranged, which leads to the following model.

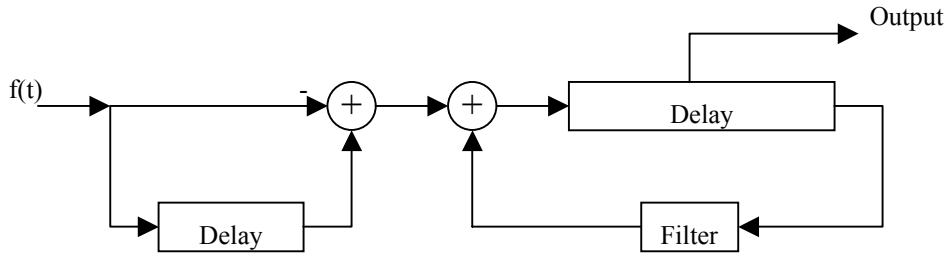

Figure 6.14: Model of the string interface used in the simulation

Thereby, the left delay equals the sum of the two delays on the left in Figure 6.3. It acts as a comb-filter delay corresponding to the striking point along the string. The right delay is equal to the sum of all delays in Figure 6.3. The new structure introduces a further delay, to be sure, but simplifies the calibration of the loop.

The length of the string

The fundamental frequency of the output signal is determined by the effective length of the delay loop. The delay length (in samples) can be computed as

$$L = \frac{f_s}{f_0} \tag{6.35}$$

where f_s is the sampling rate of the synthesis system, and f_0 is the fundamental frequency. In general, L is a positive real number. Thus, a fractional delay has to be used for the determination of the correct pitch.

The fractional part can be computed by linear interpolation between the two closest delay lengths. This yields the following problem. As linear interpolation acts as a lowpass filter in the delay loop, high frequency components will decay to fast.

An alternative method is to use an allpass filter to generate a fractional delay. For the simulation, a first order allpass filter is used. The z transform of this filter is

$$A(z) = \frac{a + z^{-1}}{1 + az^{-1}}. \tag{6.36}$$

The coefficient a can be derived by

$$a = (1 - D) \cdot (1 + D), \tag{6.37}$$

where D stands for the desired fractional delay. As allpass fractional delay filters are recursive, they are prone to transient effects when their delay is changed during the tone production (e.g., pitch bend, glissandi, and vibrato effects). Solutions to solve this problem are given in [37]. In the case of the piano, where the delay length is not changed during a key is depressed, this problem will not appear.

The loop filter

In the simulation, a second order FIR lowpass filter with symmetric impulse response is used to model the frequency dependent damping. Thereby, it is required that the loop gain never exceeds 1. The computational costs for this kind of implementation are quite low.

The dispersion filter

Two first order allpass filters in cascade where used to demonstrate the dispersion effect which is caused by the stiffness of the string. For a correct positioning of several tens of inharmonic partials, the allpass filter order has to be around 20 [4]. However, this would be to expensive for the target application. Thus, only the effect of inharmonicity and not the correct spacing of the inharmonic partials will be considered in this simulation.

Therefore it has to be noticed, that the use of these allpass filters introduces a further delay, which alters the pitch of the produced tone. As a result, the phase delay values of the individual allpass filters at the desired pitch have to be subtracted from the original length of the delay line to generate the correct pitch.

6.4.3 The entire simulation

For the entire simulation, all these previously discussed sections of the piano model have to be combined. Furthermore, the impulse response of the soundboard has to be generated. Therefore, exponentially decaying white noise is used. The impulse response is further commuted with the digital waveguide and the excitation system. Thus, it only has to be convolved with the short excitation signal instead of the output signal of the digital waveguide.

In the simulation, a gain factor g has to be attached to the feedback loop to control the decay rate of the produced tone. This is necessary for the release part of the synthetically generated piano tone.

To simulate the two-stage decay, two or three slightly detuned string models are used, which are driven by the same excitation signal. Thereby, the coupling of the strings by the use of a coupling filter (see Figure 6.4) is not implemented.

7 Comparison of the investigated synthesis techniques

This chapter deals with the memory effort and the computational expense that is needed for the implementation of the sound synthesis techniques discussed in the previous chapters.

7.1 FM synthesis

The FM synthesis algorithm discussed in 4.6 consists of four interconnected operators and their related envelope generators. The computationally most efficient implementation of the operators can be reached by the use of a table lookup oscillator to generate the individual sine values. Thereby, the results for non-integer values of the increment can be determined by the linear interpolation.

7.1.1 Estimated data amount

The amount of data is mainly determined by the implementation of the sine table. In consideration of the quarter-wave symmetry and of a resolution of 16 bits, the amount of table entries can be set to 64. Thereby, the SNR (in the case of linear interpolation) amounts 80 dB. The SNR results for different memory word lengths are given in Table 4.3. Furthermore, 100 entries of the modulation index, 12 entries of the key-scaling curves, and about 20 constants per operator have to be stored in memory. Table 7.1 summarizes the estimated amount of data.

	Bytes
Sine table with 64 entries	128
Modulation index table with 100 entries	200
Key scaling tables (4 key scales a´ 3 grid points)	24
20 constants per operator	160
Sum	**512**

Table 7.1: Estimated data amount for the FM piano instrument; memory word length is chosen to be 16 bits

7.1.2 Estimated calculation effort for one voice

Since the initial amplitude value as well as the initial modulation index depends only on the velocity and on the key number, they have to be calculated just once per keystroke and per voice. As a result, the computational expense is dominated by the calculation of the envelope and the operator functions, which must be calculated once per audio sample.

The table lookup can be performed by one addition of the phase value and linear interpolation (one addition and one multiplication).

	ADD/SUB	MPLY	Table lookups	Sum
Amplitude	4	4	-	8
Modulation index	8	-	4	12
Envelope	4	4	-	8
Operator	8	24	4	36

Table 7.2: Estimated calculation effort per output sample

The estimated calculation effort to generate one second of output samples by a given sample rate is

Amplitude + Modulation Index + Sample Rate(Envelope + Operator).*

Thus, a sample rate of 32 kHz for instance will lead to approximately 1,4 MIPS.

7.2 Sampling synthesis

Basically, the sampling algorithm is computationally very efficient. However, a large amount of memory is needed to store the recorded sounds of real instruments. Since memory is rare, the original samples have to be minimized in length. Thus, looping, pitch shifting and time-variant filtering must be used to simulate the behavior of the original sample. Consequently, the calculation effort increases.

7.2.1 Estimated data amount

Actually, the amount of data is only determined by the number of used samples and their corresponding lengths. If a sample of every 7-th semitone (interval of a quint) of a real piano is used, and if each sample can be shortened to the value of about 200 ms, the amount of required memory will be approximately 200 Kbytes (for a sample rate of 44,1 kHz, and a memory word length of 16 bit). However, there has to be considered that for the very low piano tones a sample length of 200 ms would be to short.

For the target application, the number of used piano samples can be reduced due to the limited bandwidth of the loudspeaker. For example, seven samples, starting with the C3 (130,8 Hz) would be sufficient. Furthermore, the sample rate can be reduced to the amount of 32 kHz. Therefore, the quantity of required memory would be 89 Kbytes.

7.2.2 Estimated calculation effort for one voice

The computational costs are mainly based on determining the envelope function and on filtering each sample of the output signal.

	ADD/SUB	MPLY	Sum
Amplitude	2	3	5
Envelope	1	1	2
Shelving-Filter	8	16	24

Table 7.3: Estimated calculation effort per output sample

Therefore, the estimated calculation effort to generate one second of output samples can be computed by

Amplitude + Sample Rate(Envelope + Shelving-Filter)*

Thus, a sample rate of 32 kHz for instance will lead to approximately 0,832 MIPS.

7.3 Waveguide synthesis

In the digital waveguide model, the required memory depends on the note number of the synthesized tone, which is different to the FM synthesis and to the sampling method, where the memory usage is equal for all notes (given that the lengths of the used samples in the sampling method are equal).

7.3.1 Estimated data amount

In the model, the pitch of the output signal is determined by the effective length of the delay line.

$$f_0 = \frac{f_s}{N} \tag{7.1}$$

For the synthesis of the C2 (65 Hz), this means that the length of the delay line has to be 489 samples, which should be the worst case. Thereby, the sample rate was chosen to be 32 kHz. Furthermore, the impulse response of the soundboard with a duration of about 300 ms has to be stored in the memory too. For the generation of the hanning windows that are used to determine the excitation signal, a quarter-wave of the cosine has to be stored in a table. As a result, the following estimation of the required memory can be given.

	Bytes
Delay line to generate a 65 Hz tone	984
Impulse response of the soundboard	19200
Cosine table with 64 entries	128
Sum	**20312**

Table 7.4: Estimation of the memory effort for one voice; the memory word length is chosen to be 16 bits

7.3.2 Estimated calculation effort for one voice

	ADD/SUB	MPLY	Sum
FIR loss filter	3	3	6
Allpass (correct pitch)	2	2	4
Allpass (dispersion)	4	4	8

Table 7.5: Estimated calculation effort per output sample

The estimated calculation effort to generate one second of output samples by a given sample rate is

Sample Rate(FIR loss filter + Allpass (correct pitch) +Allpass (dispersion)).*

Thus, a sample rate of 32 kHz for instance will lead to approximately 0,57 MIPS.

8 Conclusion

In this thesis, FM synthesis, sampling synthesis and digital waveguide synthesis have been investigated by means of the synthesis of the real piano tone.

Considering FM synthesis, it turned out that it is not easy to find a convenient interconnection of the individual operators. Furthermore, it was not a simple task to establish appropriate parameters. It was mainly based on a trail and error process. Thereby, the sound quality of the synthesized real piano tone was not very satisfying as it sounds more like an electric piano.

When it comes to the sampling synthesis, the sound quality strongly depends on the quality of the used samples, their lengths and an ideal determination of the loop points. In the simulation, the length of the samples has been reduced to about 300 ms and the looped portions have directly been stringed together. The use of the high-frequency shelving filter for the simulation of the time-varying spectral changes in the real piano tone has turned out to be a good choice.

The synthesis of the real piano tone by means of the digital waveguide synthesis was very interesting due to its flexibility. Thus, the structure of the resultant sound can be varied in many different ways. The use of filtered white noise acting as the impulse response of the soundboard is a good approximation to the real one. Besides, it is very simple to implement.

Some simplifications are conceivable in the case of the implementation of a sound synthesis technique on a digital signal processor of a mobile phone. As the bandwidth of the integrated loudspeaker of a mobile phone is limited, not the whole frequency range of the real piano can be radiated by it. Concerning sampling synthesis, only a few samples of the real piano have to be stored in memory. The lower piano tones can be generated by shifting down the pitch of a much higher one. Furthermore, the sampling rate and the word length can be reduced. If the whole midi standard with its 128 instruments should be implemented in a mobile phone, only the sampling synthesis would lead to well sounding results. However, the amount of required memory would be very high. To overcome this problem, it is possible to implement not only the sampling method, but also to use a mixed form of synthesis techniques. Thereby, the sampling method can be used to synthesize the tones of the natural instruments; all the synthetic sounds of the midi standard can be realized by the use of the FM synthesis.

Bibliography

1. Conklin, H.A., *Design and tone in the mechanoacoustic piano. Part III. Piano strings and scale design.* Journal of the Acoustical Society of America, 1996. **100**(3): pp. 1286-1298.
2. Conklin, H.A., *Design and tone in the mechanoacoustic piano. Part I. Piano hammers and tonal effects.* Journal of the Acoustical Society of America, 1996. **99**(6): pp. 3286-3295.
3. Conklin, H.A., *Design and tone in the mechanoacoustic piano. Part II. Piano structure.* Journal of the Acoustical Society of America, 1996. **100**(2): pp. 695-707.
4. Avanzini, F., et al., *Musical instrument modeling: the case of the piano*. p. 10.
5. Askenfelt, A. and Jansson, E.V., *From touch to string vibrations.I: Timing in the grand piano action.* Journal of the Acoustical Society of America, 1990. **88**(1): pp. 52-63.
6. Askenfelt, A., *From touch to string vibrations.II: The motion of the key and the hammer.* Journal of the Acoustical Society of America, 1991. **90**(5): pp. 2383-2393.
7. Boutillon, X., *Model for piano hammers:Experimental determination and digital simulation.* Journal of the Acoustical Society of America, 1988. **83**(2): pp. 746-754.
8. Chaigne, A. and Askenfelt, A., *Numerical simulation of piano strings. I. A physical model for a struck string using finite difference methods.* Journal of the Acoustical Society of America, 1994. **95**(2): pp. 1112-1118.
9. Stulov, A., *Hysteretic model of the grand piano hammer felt.* Journal of the Acoustical Society of America, 1995. **97**(4).
10. Giordano, N. and Korty, A., *Motion of a piano string: Longitudinal vibrations and the role of the bridge.* Journal of the Acoustical Society of America, 1996. **100**(6): pp. 3899-3908.
11. Smith, J.O. *Viewpoints on the history of Digital Synthesis*. in *Proceedings of the International Computer Music Conference*. 1991. Montreal,Cananda. pp. 1-10.
12. Dodge, C. and Jerse, T.A., *Computer Music*. 2 ed. 1984: Schirmer Books. p. 381.
13. Moore, R., *Elements of computer music*. 1990, New Jersey: PRENTICE-HALL, INC. p. 560.
14. Roads, C., *The computer music tutorial*. 5 ed. 1995: The MIT Press. p. 1234.
15. Tolonen, T., Välimäki, V. and Karjalainen, M., *Evaluation of Modern Sound Synthesis Methods*. 1998, Helsinki University of Technology.

16. Karplus, K. and Strong, A., *Digital Synthesis of Plucked-String and Drum Timbres.* Computer Music Journal, 1983. **7**(2): pp. 43-55.

17. Smith, J.O., *Phyical Modeling using Digital Waveguides.* Computer Music Journal, 1992. **16**(4): pp. 74-91.

18. Horner, A., Beauchamp, J. and Haken, L., *Methods for Multiple Wavetable Synthesis of Musical Instrument Tones.* Journal of the Audio Engineering Society, 1993. **41**(5): pp. 336-356.

19. McAulay, R. and Quatieri, T., *Speech analysis/synthesis based on a sinosoidal representation.* IEEE Transactions on Acoustics, Speech, and Signal Processing, 1986. **34**(6): pp. 744-754.

20. Hiller, L. and Ruiz, P., *Synthesizing Musical Sounds by Solving the Wave Equation for Vibrating Objects: Part I.* Journal of the Audio Engineering Society, 1971. **19**(6): pp. 462-470.

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

22. Chowning, J., *The synthesis of complex audio spectra by means of frequency modulation.* Journal of the Audio Engineering Society, 1973. **21**(7): pp. 526-534.

23. Gordon, J.W. and Smith, J.O., *A sine generation algorithm for VLSI application.* Proceedings of the 1985 International Computer Music Conference, 1985: pp. 165-168.

24. Moore, F.R., *Table Lookup Noise for Sinusoidal Digital Oscillators.* Computer Music Journal, 1977: pp. 26-29.

25. Bartsch, J., *Taschenbuch mathematischer Formeln*. 18 ed. 1998, München: Carl Hanser Verlag. p. 699.

26. Chowning, J. and Bristow, D., *FM Theory & Applications*. 1986, Tokyo: Yamaha Music Foundation. p. p.195.

27. Laroche, J. and Dolson, M., *Improved phase vocoder time-scale modification of audio.* IEEE Trans. on Speech and Audio Processing, 1999. **7**(3): pp. 323-332.

28. Zölzer, U., *DAFX Digital Audio Effects*. 2002, Baffins Lane, Chichester: John Wiley & Sons,Ltd. p. 533.

29. Rabiner, L. and Gold, B., *Theory and Application of Digital Signal Processing*. 1975, New Jersey: PRENTICE-HALL, INC. p. 762.

30. Yegnanarayana, B., *Design of Recursive Group-Delay Filters by Autoregressive Modeling.* IEEE Transactions on Acoustics, Speech, and Signal Processing, 1982. **30**(4): pp. 632-637.

31. Scalcon, F., Rocchesso, D. and Borin, G. *Subjective Evaluation of the Inharmonicity of synthetic Piano Tones*. in *ICMC Proceedings*. 1998. pp. 53-56.

32. Smith, J.O. *Efficent Synthesis of Stringed Musical Instruments*. in *ICMC Proceeding*. 1993. pp. 64-71.

33. Van Duyne, S., Pierce, J. and Smith, J.O. *Traveling Wave Implementation of a Lossless Mode-Coupling Filter and The Wave Digital Hammer*. in *ICMC Proceedings*. 1994. Aarhus. pp. 411-418.

34. Van Duyne, S. and Smith, J.O. *Developments for the Commuted Piano*. in *ICMC Proceedings*. 1995. Banff. pp. 319-326.

35. Fettweis, A., *Wave digital filters: theory and practice.* Proc. IEEE, 1986. **74**(2).

36. Smith, J.O. and Van Duyne, S. *Commuted Piano Synthesis*. in *ICMC Proceedings*. 1995. Banff. pp. 335-342.

37. Van Duyne, S., et al. *A Lossless, Click-free, Pitchbend-able Delay Line Loop Interpolation Scheme*. in *ICMC Proceedings*. 1997. pp. 252-255.

Appendix A Software

For each of the discussed sound synthesis techniques, some Matlab™ - functions have been created. These functions and their corresponding parameters are listed below.

A.1 FM synthesis

For the FM synthesis, one function for the whole instrument and two subfunctions have been programmed. The first subfunction is used to calculate the two-stage amplitude envelope. The second one represents a table-lookup oscillator, which is used to interpolate the sine function.

The entire FM instrument

```
Function:   out = piano(note,fs)

Input parameters:
   note…    input vector with input arguments [dur,pitch,vel]
              Dur…   duration of the generated sound
              Pitch… pitch of the generated sound
              Vel…   velocity value (0-127)
   fs…      sample rate
```

Envelope generator

```
Function:   out = envelope2(dur,dur1,time,level,fs)

Input parameters:
   dur…     maximum duration of the tone
   dur1…    duration of the generated tone
   time…    specifies a point in time within the whole envelope
   level…   magnitude of envelope at that point of time
   fs…      sample rate
```

Table-lookup oscillator

```
Function:   out = tableinterp(table,f,fs)

Input parameters:
   table…   table with entries of the sine function
   f…       desired output frequency
   fs…      sample rate
```

A.2 Sampling synthesis

The completely sampling instrument is included in one function. Furthermore, a function to equalize the amplitude envelope of the looped sample has been created.

The entire sampling instrument

```
Function:   out = sampler(fc,value,sample,vel,shift,dur)

Input parameters:
   fc…      cutoff frequency of the used shelving filter
   value…   defines the decay rate of the envelope generator
   sample…  input sample
   vel…     velocity value (0-127)
   shift…   value of pitch-shifting in semitones
   dur…     duration of the generated output signal
```

Equalization of the amplitude envelope

```
Function:   out = equalize(infile,taps,overhead)

Input parameters:
   infile…  sample, which should be equalized
   taps…    number of taps of the used moving average filter
   offset…  point within the sample, which is used for the
            normalization of the envelope
```

A.3 Digital waveguide synthesis

In the case of the digital waveguide synthesis, two functions have been used to generate the output signal. The first function contains a FIR lowpass filter for the frequency-dependent losses. The second one additionally has to 1^{st} order allpass filters in cascade to simulate the dispersion effect. Furthermore, subfunctions to create the excitation signal and to calculate the correct length of the delay line are used.

Model without dispersion

```
Function:   out = model1(force,pitch,dur,comb)

Input parameters:
   force…   hammer-string interaction force pulses
   pitch…   pitch of the created output signal
   dur…     duration of the generated output signal
   comb…    length of the comb-filtering delay in samples
```

Model with dispersion

```
Function:   out = model2(force,pitch,dur,ih1,ih2,comb)

Input parameters:
   force…   hammer-string interaction force pulses
   pitch…   pitch of the created output signal
   dur…     duration of the generated output signal
   ih1…     coefficient of the first allpass filter
   ih2…     coefficient of the second allpass filter
   comb…    length of the comb-filtering delay [samples]
```

Generation of the excitation signal

```
Function:   force = excitation(vel1,vel2,vel3,N,del,noise,dur)

Input parameters:
   vel1…    value of the amplitude of the first hanning window
   vel2…    value of the amplitude of the second hanning window
   vel3…    value of the amplitude of the third hanning window
   N…       number of points of the individual hanning windows
   del…     delay of the individual hanning windows [ms]
   noise…   noise signal for the impulse response of the soundboard
   dur…     duration of the exponentially decaying noise signal
```

Determination of the correct delay line length

```
Function:   out = corr_delayline1(pitch,a1,a2)

Input parameters:
   pitch…   pitch of the generated output signal
   a1…      coefficient of the first allpass filter
   a2…      coefficient of the second allpass filter
```

Appendix B Sound examples

For each of the discussed and implemented sound synthesis techniques, representative sound examples have been generated. Thereby, the different phases of the synthesis of the real piano sound should be clarified.

All sound examples are compiled on the accompanying Compact Disc and the following subchapters present the content of the CD.

B.1 FM synthesis

Title	
	Different velocity values
01	Note C2 with a velocity value of 40
02	Note C2 with a velocity value of 127
03	Note C4 with a velocity value of 40
04	Note C4 with a velocity value of 127
	Key scaling
05	Note C1 without key scaling
06	Note C1 with key scaling
07	Note C3 without key scaling
08	Note C3 with key scaling
09	Note C5 without key scaling
10	Note C5 with key scaling
	Envelope generators
11	Note C2 without envelope generators (except for the carrier oscillator)
12	Note C4 without envelope generators (except for the carrier oscillator)
13	Note C6 without envelope generators (except for the carrier oscillator)
14	Note C2 with envelope generators for all 4 operators
15	Note C4 with envelope generators for all 4 operators
16	Note C6 with envelope generators for all 4 operators

	Release part of the generated tone
17	Note C4 without release envelope
18	Note C4 with release envelope (duration of release part: 500 ms)
19	Note C4 with release envelope (duration of release part: 250 ms)
	Scale of C major
20	Scale starting at C4; velocity = 127; duration of each tone = 2 sec
21	Scale starting at C4; velocity = 30; duration of each tone = 2 sec
	FM sound with overlaid filtered attack noise of a real piano sample
22	Filtered attack noise (bandpass with f_1=300 Hz and f_2 = 1,6 kHz)
23	Note C4 with overlaid attack noise
24	Note C5 with overlaid attack noise

B.2 Sampling synthesis

Title	
	Original Piano Samples
25	Note C1; length = 2,4 sec
26	Note C3; length = 2,4 sec
27	Note C5; length = 2,4 sec
28	Note C7; length = 2,4 sec
	Equalization of the amplitude envelope of the looped sample
29	Note C5; no equalization; length of looped region = 532 ms
30	Note C5; with equalization; length of looped region = 532 ms
	Time-varying filtering of a short looped region without cross-fading (only the filtered output without envelope generator)
31	Note C1; loop-start = 221 ms; loop-length = 60 ms
32	Note C3; loop-start = 153 ms; loop-length = 23 ms
33	Note C5; loop-start = 173 ms; loop-length = 19 ms
34	Note C7; loop-start = 172 ms; loop-length = 28 ms

	Different velocity values (looping + filtering + envelope)
35	Note C3; velocity value = 10
36	Note C3; velocity value = 127
37	Note C5; velocity value = 10
38	Note C5; velocity value = 127
	Pitch-shifting (looping + filtering + envelope)
39	Note C5; without pitch-shifting
40	Note C5; pitch-shift = - 1 semitone
41	Note C5; pitch-shift = - 3 semitone
42	Note C5; pitch-shift = - 5 semitone
43	Note C5; pitch-shift = 2 semitone
44	Note C5; pitch-shift = 4 semitone
45	Note C5; pitch-shift = 5 semitone

B.3 Digital waveguide synthesis

Title	
	Losses with and without frequency dependence
46	Note C4; damping factor = 0.995; without 2nd order lowpass filter
47	Note C4; damping factor = 0.997 and with 2nd order lowpass filter
	Dispersion (two 1st order allpass filters in cascade)
48	Note C4; allpass coefficients: $a_1 = a_2 = 0,7$
49	Note C4; allpass coefficients: $a_1 = a_2 = 0,8$
50	Note C4; allpass coefficients: $a_1 = a_2 = 0,9$
	Generating the impulse response of the soundboard
51	Exponentially decaying white noise; duration = 500ms
52	Exponentially decaying white noise; duration = 1000ms
53	Noise (500 ms) convolved with the hammer force signal for Note C4
54	Noise (1000 ms) convolved with the hammer force signal for Note C4

	Resultant sound **(with lowpass filter, allpass filters and excitation signal)**
55	Note C2; excitation signal (500 ms); velocity = 100
56	Note C4; excitation signal (500 ms); velocity = 100
57	Note C4; excitation signal (1000 ms); velocity = 100
58	Note C4; excitation signal (1000 ms); velocity = 50
59	Note C7; excitation signal (500 ms); velocity = 100
	Different lengths of the comb-filtering delay (Position of the hammer)
60	Note C2; delay length = 10 Samples
61	Note C2; delay length = 60 Samples
62	Note C4; delay length = 10 Samples
63	Note C4; delay length = 20 Samples
	Two slightly detuned strings to generate the effect of beating
64	Note C4; one string
65	Note C4; two strings; frequency difference of 0,4 Hz
66	Note C7; one string
67	Note C7; two strings; frequency difference of 0,3 Hz