

# DISCRETE-TIME MODELING AND SYNTHESIS OF MUSICAL INSTRUMENTS

*Matti Karjalainen*

Helsinki University of Technology  
Laboratory of Acoustics and Audio Signal Processing  
Otakaari 5 A, 02015-HUT, Finland  
matti.karjalainen@hut.fi

## ABSTRACT

Physics-based modeling of sound sources, particularly of musical instruments, can be carried out in many forms. In this paper an overview is presented focusing on methods that are potential in real-time simulation and synthesis, in contrast to solving equations that characterize the behavior of the sound source under study. Modeling paradigms discussed here cover the digital waveguides (DWGs), finite difference time domain schemes (FDTDs), wave digital filters (WDFs), modal decomposition techniques, and source-filter models. The interrelations between these approaches are discussed and cases of applying them for real-time sound synthesis are presented.

## 1. INTRODUCTION

Real-world systems of interest in acoustics and computer music are typically continuous in time and space, and therefore their dynamic behavior is inherently described by partial differential equations [1]. Computer-based modeling and simulation of them requires, however, discretization of the underlying PDEs, which in a general case will correspond to the continuous analog system only when sampling rate approaches infinity, i.e., temporal and spatial sample intervals are made infinitesimally small.

In this paper we discuss several discrete-time modeling paradigms, particularly the *digital waveguides* (DWGs), *wave digital filters* (WDFs), and *finite difference time domain schemes* (FDTDs). Their properties are compared and their mixing to hybrid modeling techniques are probed further from previous studies. Case studies characterize the realization principles as applied to musical instruments.

### 1.1. General viewpoints

Discretization in time and space leads to interesting and difficult problems that are not found in the ordinary continuous case. Particularly when systems are *simulated* or *synthesized*, i.e., computed efficiently in the time domain by discrete techniques, in contrast to being *solved* from equations, the question of *localized discretization* (block-wise construction of models through interconnection of elements) and consistent *scheduling* (ordering of operations) are of major importance. It is advantageous to formulate discrete-time computations in terms of *digital signal processing* (DSP) algorithms that are optimized for computational efficiency and robustness [2].

In the analog world we may in the limit assume arbitrarily short (infinitesimal) but non-zero delays between spatial points of interest, and the order of events follows the causality principle of physics. In the discrete-time world, however, a single sample period is the shortest possible non-zero time interval, in which an explicit two-way physical interaction can happen. This leads easily to the problem of *delay-free loops*, i.e., implicit equations where the output of an operation needs an input value that may be dependent on the output value not yet known. Particularly with nonlinear elements this is a fundamental problem [3]. Another fundamental problem with nonlinearities in discrete-time computation is aliasing of frequency components [2].

It is now well-known that the delay-free loop problem is easier to overcome if computations are formulated by wave quantities instead of ordinary physical quantities [4]. Referring to electric circuits, the latter ones are often called the *Kirchhoff quantities*, in contrast to *wave quantities*. Using dual K-variables (K for Kirchhoff), such as voltage and current or force and velocity, is very intuitive for circuits and their mechanical equivalents, but in the discrete methodology they are not as easy to use. K-elements, formulated as “transfer functions” between dual K-variables, cannot form circuits and networks directly, because for example voltage across and current through an electric component don’t explicate causality. Both of them may depend on a full circuit topology so that only their interrelation is fixed by the port impedance of the element. Causality is obtained if the K-quantities are transformed to W-quantities, i.e., into wave-based formulation, in order to compute them by explicit causal relations through local interactions. This is different from solving system equations<sup>1</sup> which permits global (even delay-free) interactions.

This paper is organized as follows. Section 2 presents a condensed overview of the physical modeling paradigms of interest in this study. In Section 3 we investigate the interrelations of these paradigms. This is followed in Section 4 by a short presentation of a software tool called the BlockCompiler, developed for physics-based modeling. Section 5 is devoted to case studies of selected physics-based models, followed by a summary of the paper.

## 2. PARADIGMS OF DISCRETE-TIME MODELING

A characterization of different physical modeling paradigms is presented in this section, including digital waveguides, wave digital filters, finite difference time domain schemes, modal decomposition, and source-filter modeling.

For spatially distributed systems the wave equation is probably the most fundamental starting point. In a one-dimensional lossless medium the wave equation is written

$$y_{tt} = c^2 y_{xx} \quad (1)$$

where  $y$  is (any) wave variable, subscript  $tt$  refers to second partial derivative in time  $t$ ,  $xx$  second partial derivative in place variable  $x$ , and  $c$  is speed of wavefront in the medium of interest. For example in a vibrating string we are primarily interested in transversal wave motion for which  $c = \sqrt{T/\mu}$ , where  $T$  is tension force and  $\mu$  is mass per unit length of the string [1]. In the case of acoustic wave propagation, Eq. (1) holds for lossless longitudinal plane waves. The wave equation is an inherently important starting point particularly in digital waveguides and finite difference formulations below.

### 2.1. Digital waveguides

Digital waveguide modeling is based on the fact that wave propagation in a medium can be simply and efficiently simulated by two-directional delay lines [5], one delay-line for each directional wave component. This is closely related to the d’Alembert solution of the wave equation

$$y(t, x) = \vec{y}(t - x/c) + \overleftarrow{y}(t + x/c) \quad (2)$$

where the arrows denote the right-going and the left-going components of the total waveform. Assuming that the signals are bandlimited to half of the sampling rate, we may sample the traveling waves without losing any information by selecting  $T$  as the sample interval and  $X$  the position interval between samples so that  $T = X/c$ . Sampling is applied in a discrete time-space grid in which  $n$  and  $m$  are related to time and position, respectively. The discretized version of Eq. (2) becomes [5]:

$$y(n, m) = \vec{y}(n - m) + \overleftarrow{y}(n + m) \quad (3)$$

---

<sup>1</sup>Time-domain simulation can also be based on solving global equations for each time step, but that tends to be highly inefficient compared to block-based simulation by localized interactions.

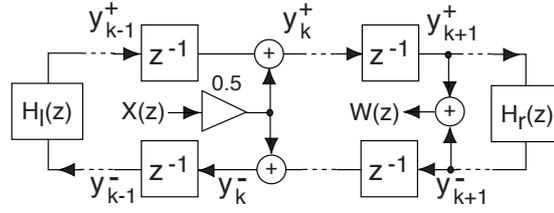


Figure 1: Digital waveguide structure based on traveling wave components from d'Alembert solution of lossless wave equation.  $X(z)$  is input excitation such as plucking force of a string and  $Y(z)$  is output such as transversal string velocity. Wave reflection at terminations is modeled by reflection filters  $H_l(z)$  and  $H_r(z)$ .

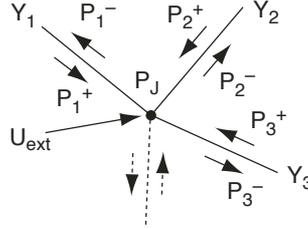


Figure 2: Parallel acoustic junction of admittances  $Y_i$  with associated volume velocity waves indicated. A direct volume velocity excitation  $U_{\text{ext}}$  is also attached.

From this it follows that the wave propagation can be computed by updating state variables in two delay lines by

$$\vec{y}_{k,n+1} = \vec{y}_{k-1,n} \quad \text{and} \quad \overleftarrow{y}_{k,n+1} = \overleftarrow{y}_{k+1,n} \quad (4)$$

i.e., by simply shifting the samples to the right and left, respectively. This kind of discrete-time modeling is called *Digital Waveguide* (DWG) modeling [5]. Figure 1 characterizes a simple 1-D digital waveguide which could describe for example the vibration of a string.

Low-pass and all-pass filters can be cascaded with delay elements to simulate damping and dispersion. Delay sequences between points of signal observation (output) and feed-in (input) can be consolidated into subsystems that are computationally highly efficient [5]. In *Commutated waveguide synthesis* this is utilized so that the elements of a string instrument, i.e., plucking, string, and body, can be rearranged so that plucking and body are realized as a wavetable that feeds a string model.

In addition to delays, junctions connecting elements are needed that fulfill physical continuity constraints, i.e., the Kirchhoff rules. For a parallel junction of acoustic components, see Fig. 2, we may write<sup>2</sup>:

$$P_1 = P_2 = \dots = P_N = P_J \quad (5)$$

$$U_1 + U_2 + \dots + U_N + U_{\text{ext}} = 0 \quad (6)$$

where  $P_i$  are pressure and  $U_i$  volume velocities at the ports of the junction,  $P_J$  is the common pressure of coupled branches and  $U_{\text{ext}}$  is an external volume velocity to the junction. When port pressures are represented by incoming wave components  $P_i^+$  and outgoing wave components  $P_i^-$ , admittances attached to each port by  $Y_i$ , and

$$P_i = P_i^+ + P_i^- \quad \text{and} \quad U_i^+ = Y_i P_i^+ \quad (7)$$

the junction pressure  $P_J$  can be obtained as:

$$P_J = \frac{1}{Y_{\text{tot}}} (U_{\text{ext}} + 2 \sum_{i=0}^{N-1} Y_i P_i^+) \quad (8)$$

<sup>2</sup>For discrete-time modeling we may assume that the variables and parameters are given by their z-transforms.

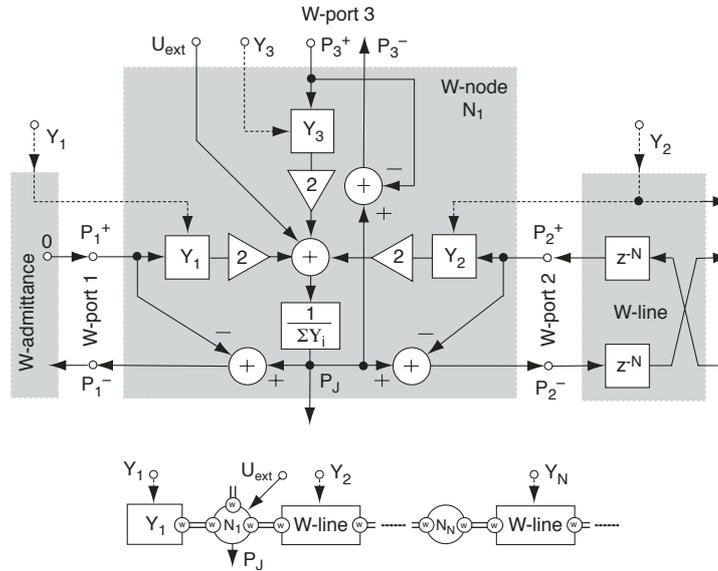


Figure 3: *Top*: A 3-port parallel scattering junction for acoustic pressure waves. Incoming pressures are  $P_i^+$ , outgoing ones  $P_i^-$ , and  $P_J$  is common junction pressure. Port 1 (left) is terminated by admittance  $Y_1$ , port 2 (right) is connected to a delay-line having wave admittance  $Y_2$ , and port 3 (top) is not connected. Possible admittance controls are marked by dashed lines. *Bottom*: Block diagram with abstracted blocks and how they can be connected to form a 1-D DWG waveguide.

where  $Y_{\text{tot}} = \sum_{i=0}^{N-1} Y_i$  is the sum of all admittances to the junction. Outgoing (scattered) pressure waves, obtained from Eq. (7), are then  $P_i^- = P_J - P_i^+$ . Figure 3 depicts this as a signal flow diagram for the computation of such a scattering junction.

The same diagram can be applied to a series connection and volume velocity waves so that pressures and volume velocities are interchanged and admittances are replaced by impedances.

In Fig. 3 (top) a signal flow diagram of a 3-port parallel scattering junction for acoustic pressure waves is shown. Port 1 is terminated by admittance  $Y_1$  and port 2 is connected to a delay-line of admittance  $Y_2$ . Notice that for a passive termination  $Y_1$  the incoming port signal is zero so that no computation is needed at that port. Port 3 is not connected. Shaded areas show how the signal processing elements are grouped for block-based (object-based) computation, and the bottom part of Fig. 3 depicts an abstraction diagram of these blocks. Arbitrary admittances can be given by their z-transforms as digital filters (FIR or IIR filters, or just as real coefficients if all attached impedances are real). Notice that each admittance is also represented in the term  $1/\sum Y_i$ . The formulation shown in Fig. 3 is useful particularly when complex time-invariant admittances are used. For real-valued admittances the scattering junction computation can be optimized for example to minimize the number of operations needed [5].

An example of DWG modeling in the mechanical domain is shown in Fig. 4, where velocity waves are used and series connected junctions have equal velocity at each port. A plucked string instrument with two strings is modeled by delay-line elements and termination impedances  $Z = F/V$ , which are ratios of force  $F$  and velocity  $V$ . Plucking is inserted as force to a point in the string and the output (da) is the vibration velocity of the bridge. Both strings have a common bridge impedance so that there is coupling between them. This can lead to *sympathetic vibrations*, whereby a vibrating string may transfer energy to another string, resulting in its vibration, too. For a full-scale model of the acoustic guitar each string should have two subsystems, one for vertical and another for horizontal vibration, and six such strings should be included.

Any number of element ports can be connected to a scattering junction of the type shown in Fig. 3. This makes it possible to form mesh-like structures where two-port delay-lines are used between nodes. Such structures are useful for constructing *digital waveguide meshes* in two and three dimensions [6].

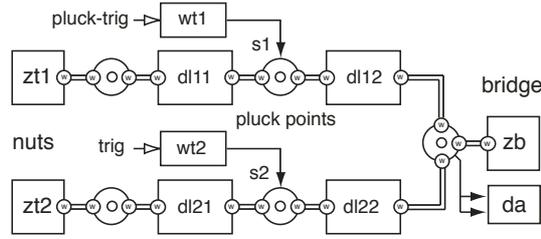


Figure 4: A DWG block diagram of two strings coupled through a common bridge impedance ( $Z_b$ ) and terminated at the other end by nut impedances ( $Z_{t1}$  and  $Z_{t2}$ ). Plucking points are for force insertion from wavelables into junctions in the delay-lines. Output is the bridge velocity.

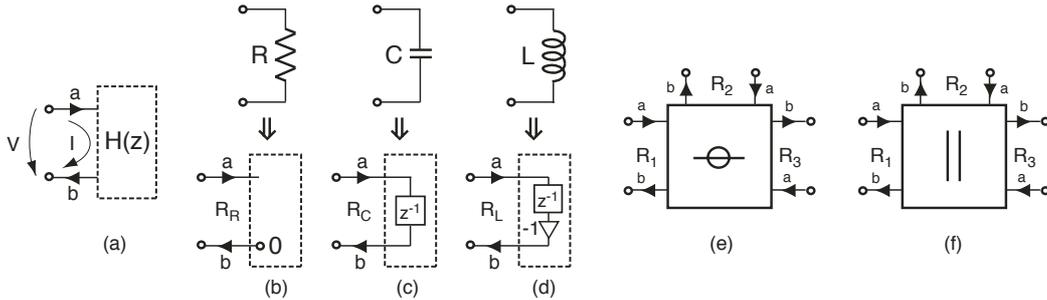


Figure 5: Wave digital filter components: (a) 1-port component generically, (b) WDF resistor  $R_R = R$ , (c) WDF capacitor  $R_C = T_s/2C$ , (d) WDF inductor  $R_L = 2L/T_s$ , (e) 3-port series adaptor, and (f) 3-port parallel adaptor.

## 2.2. Wave digital filters

Wave digital filters (WDFs) are models that were originally developed for discrete-time simulation of lumped element circuits and systems, as known from the theory of analog electric circuits [4]. The close relationship between them and digital waveguides is now well known [5, 7]. While DWGs emphasize delays and wave propagation, WDFs have emphasis on lumped element modeling. However, both are capable to both types, and actually they are compatible and complementary approaches for wave-based modeling.

The WDF formalism is based on a notation of ('voltage') waves  $a$  and  $b$  as

$$\begin{cases} a = V + RI \\ b = V - RI \end{cases} \Leftrightarrow \begin{cases} V = (a + b)/2 \\ I = (a - b)/2R \end{cases} \quad (9)$$

where  $a$  is in-coming and  $b$  is out-going wave in a port,  $V$  is voltage and  $I$  is current as Kirchhoff variables, and  $R$  is port resistance (reference resistance), as denoted in Fig. 5(a). Figures 5(b-f) present the most basic elements used in wave digital filters: resistor, capacitor, inductor, series adaptor, and parallel adaptor. The two latter ones are used to construct circuit models by series and parallel connections of the component blocks. There are several other important circuit elements available for WDFs [4], and the wave digital filter theory has also been extended to multidimensional circuits and systems [7].

The wave digital resistor in Fig. 5(b) is simple, and as for a resistive DWG junction in Fig. 3, no real computation is needed since there is no feedback path from wave input  $a$  to output  $b$ . Port reference resistance  $R_R$  is simply equal to the resistance  $R$  of the resistor.

Analog capacitors and resistors can only be approximately represented when using discrete-time techniques. The most appropriate way is to use the structures shown in Fig. 5(c) for a WDF capacitor and in Fig. 5(d) for a WDF inductor<sup>3</sup>. These feedbacks through a unit delay correspond to a bilinear mapping of the analog domain (Laplace

<sup>3</sup>It can be shown that these are the best lossless approximations for a capacitor and an inductor.

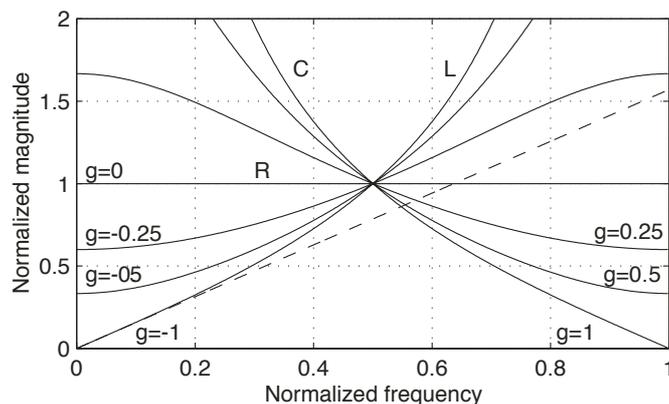


Figure 6: Normalized impedance  $|Z(z)/R_Z|$  for WDF elements with delay feedback coefficient  $g = -1 \dots 1$ . Frequency scale normalized to Nyquist frequency. Specific cases: L = inductance, C = capacitance, R = resistance. Admittance of an analog inductance, approximated by the WDF inductance, is plotted by dashed line.

domain) to discrete-time domain ( $z$ -domain). In general the impedance realized by feedback  $H(z)$  from  $a$  to  $b$  in WDF components results in impedance

$$Z_C(z) = \frac{1 + H(z)}{1 - H(z)} R_Z \quad (10)$$

where  $R_Z$  is the port resistance and  $Z(z)$  is the impedance achieved in a  $z$ -domain expression. For the WDF capacitor in Fig. 5(c) we get

$$Z_C(z) = \frac{1 + z^{-1}}{1 - z^{-1}} R_C \quad (11)$$

Figure 6, curve C, characterizes the impedance of  $Z_C(z)$ , which at low frequencies ( $f \ll 1/2T_s$ ) resembles an analog capacitor of capacitance value

$$C = T_s/2R_C \quad \text{and thus} \quad R_C = T_s/2C \quad (12)$$

where  $R_C$  is port resistance and  $T_s$  is the sampling period applied. Correspondingly for an inductor we get

$$Z_L(z) = \frac{1 - z^{-1}}{1 + z^{-1}} R_L \quad (13)$$

Curve L in Fig. 6 shows the impedance behavior of the WDF inductor, which at low frequencies now approximates an analog inductor of

$$L = T_s R_L/2 \quad \text{and thus} \quad R_L = 2L/T_s \quad (14)$$

More generally for  $H(z) = g z^{-1}$ , parameter  $g$  varying between  $-1 \dots 0 \dots 1$ , the impedance obtained varies from inductive (lossless) to resistive and to capacitive (lossless) as characterized in Fig. 6.

Notice that due to the bilinear mapping, the capacitors and inductors realized in Eqs. (11) and (13) do not show a good match with the equivalent analog components at frequencies approaching the Nyquist frequency. Only on a warped angular frequency scale  $\Omega$ ,

$$\Omega = (2/T_s) \tan(\omega T_s/2) \quad (15)$$

the admittances of the digital capacitance and inductance of Eqs. (11) and (13) match with their analog counterparts [4]. This is not necessary a problem, because the good match at low frequencies is often enough, or it is possible to design pre-warped frequency responses such that the final realization has a desired frequency behavior.

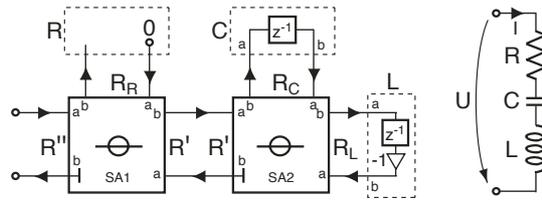


Figure 7: (Left) A WDF series connection of resistor (R), capacitor (C), and inductor (L) constructed by two three-port series adaptors (SA1 and SA2). (Right) Equivalent analog circuit.

In Fig. 7, a WDF model for a series connection of inductor L, capacitor C, and resistor R is depicted. It is constructed by two three-port *series adaptors* (SA1 and SA2) that implement wave scattering according to Kirchhoff laws. The adaptors in the circuit model also show how delay-free loops are avoided by impedance-matched reflection-free ports denoted by  $\dashv$  in the adaptors. For more details, see [4].

Instead of electric circuits, the WDF models can be used in other physical domains, such as mechanical and acoustical domains. In a direct analogy of a mechanical resonator, Fig. 7 can be interpreted as a mass-spring-damper system, where all components have the same velocity, force corresponding to voltage, velocity to current, mass to inductance, spring constant to capacitance, and damping coefficient to resistance. In the acoustical domain, force is replaced by pressure and velocity by volume velocity.

### 2.3. Finite difference models

Finite difference approximation is a popular method of numerical integration of partial differential equations [8]. In physical modeling it is used particularly for multidimensional mesh structures [7] but also for example in one-dimensional string modeling [9].

In the most commonly used way to discretize the wave equation by finite differences the partial derivatives in Eq. (1) are approximated by centered differences. The centered difference approximation to the spatial partial derivative  $y_x$  is given by [8]

$$y_x \approx (y(x + \Delta x/2, t) - y(x - \Delta x/2, t)) / \Delta x \quad (16)$$

where  $\Delta x$  is the spatial sampling interval. A similar expression is obtained for the temporal partial derivative, if  $x$  is kept constant and  $t$  is replaced by  $t \pm \Delta t$ , where  $\Delta t$  is the discrete-time sampling interval. Iterating the difference approximations, second-order partial derivatives in Eq. (1) are approximated by

$$y_{xx} \approx (y_{x+\Delta x,t} - 2y_{x,t} + y_{x-\Delta x,t}) / \Delta x^2 \quad (17)$$

$$y_{tt} \approx (y_{x,t+\Delta t} - 2y_{x,t} + y_{x,t-\Delta t}) / \Delta t^2. \quad (18)$$

where the short-hand notation  $y_{x,t}$  is used instead of  $y(x, t)$ . By selecting  $\Delta t = \Delta x/c$ , and using index notation  $k = x/\Delta x$  and  $n = t/\Delta t$ , Eqs. (17) and (18) result in

$$y_{k,n+1} = y_{k-1,n} + y_{k+1,n} - y_{k,n-1} \quad (19)$$

From Eq. (19) we can see that a new sample  $y_{k,n+1}$  at position  $k$  and time index  $n + 1$  is computed as the sum of its neighboring position values minus the value at the position itself one sample period earlier. Since  $y_{k,n+1}$  is a physical variable, not a wave variable, we will classify the finite difference models as *K-models*, referring to Kirchhoff type of physical variables.

In [10] an extension of the scheme is derived that allows FDTD structures with arbitrary connection admittances to be formed, see Fig. 8, in a way resembling the DWG diagram in Fig. 3. A generalized recursion formulation for the structure (without excitation  $U_{\text{ext}}$ ) becomes

$$P_{J,n+1} = \frac{2}{Y_{\text{tot}}} \sum_{i=0}^{N-1} Y_i P_{i,n} - P_{J,n-1} \quad (20)$$

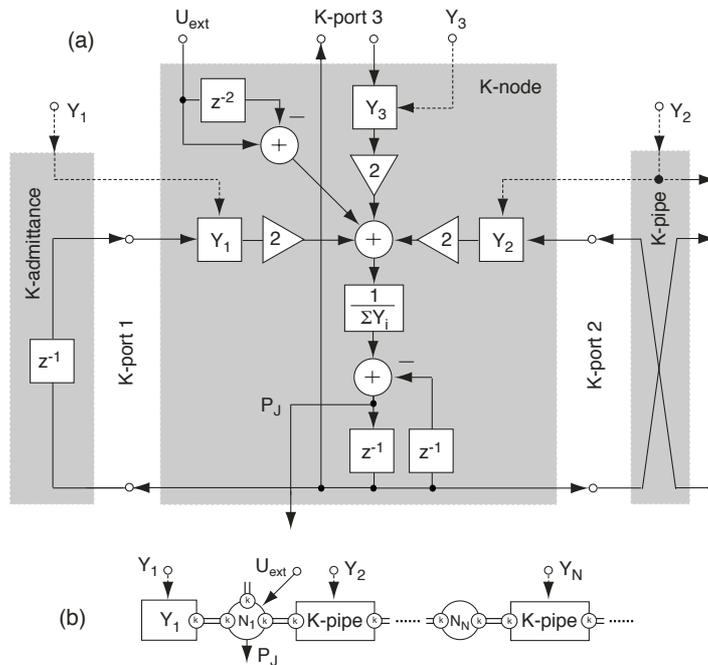


Figure 8: *Top*: An acoustic three-port parallel connection of FDTD type, corresponding to the DWG in Fig. 3. *Bottom*: Abstraction as a blockwise description of a 1-D FDTD waveguide.

It is also shown in [10] that volume velocity excitation  $U_{\text{ext}}$  must be fed through a simple FIR filter  $H(z) = 1 - z^{-2}$ , so that the zeros of this filter cancel out the poles at DC and Nyquist frequency inherent in the DFTD structure due to the two-unit-delay feedback as shown in Fig. 8. The equivalence of the DWG in Fig. 3 and the FDTD structure in Fig. 8 has been proven in [10]. Details of their relations will be discussed in Section 3.4.

## 2.4. Modal decomposition techniques

While the DWG, WDF, and FDTD methods above are explicit time-domain simulation techniques, *modal decomposition* methods rely on frequency-domain formulations of systems under study. They decompose the behavior of a system into decaying exponentials, whereby oscillatory components represent resonances or eigenmodes of the system. Modal decomposition methods include the traditional *modal synthesis* [11] and a newer approach called the *functional transformation method* (FTM) [12].

Modal decomposition models and modal synthesis can be realized as a technique by itself, such as the Modalys software [13], or by the WDF methodology. In Section 3.3 we will discuss the realization of arbitrary impedance or admittance WDF elements, including modal elements, and in Section 5.4 we will study semiphsical modeling of bells, based on the modal decomposition paradigm.

## 2.5. Source-filter modeling

Source-filter modeling is a reduced form of physics-based modeling so that the two-directional causal physical interaction is reduced to one-way interaction, therefore allowing normal signal processing realizations using a signal generator (source) and a filter. Classical examples thereof are speech synthesis algorithms using formant filters [14] or LPC (linear prediction) filter formulations [15].

The Karplus-Strong model [16] is a case of modeling musical instruments, particularly string instruments, considered as an extremely reduced form of physics-based modeling. Figure 9 shows a block diagram for a (modified) Karplus-Strong string model. It consists of a triggerable wavetable (wt, for example a noise burst), a

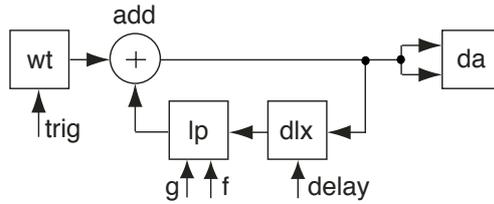


Figure 9: Block diagram of a (modified) Karplus-Strong model as a case of semiphysical modeling.

feedback loop with a (fractional) delay (dlx), and a low-pass filter (lp), in order to generate the decaying string sound by a recursive comb filter<sup>4</sup>. An advanced version of this model includes excitation wavetables obtained by inverse filtering from real instrument recordings, a dual-polarization model of a string, and connections between strings to simulate sympathetic vibrations [17].

### 3. INTERRELATIONS AND MIXING OF THE MODELING PARADIGMS

Physics-based modeling is typically done using only one of the paradigms presented above. Researchers have their favorite methods and for example a combination of different approaches to optimize the overall model is used only rarely. In this section we will explore the interrelations of the paradigms and how they can be mixed.

#### 3.1. K- vs. W-modeling

As mentioned above, modeling paradigms can be categorized to *K-modeling* with Kirchhoff quantities and *W-modeling* with wave quantities. Digital waveguides and wave digital filters are inherently based on W-modeling, while the FDTD formulation is based on K-modeling with a single K-variable (pressure in Fig. 8). Analog systems and circuits are typically best described by dual K-variables, such as voltage and current, force and velocity, pressure and volume velocity, etc. Some basic considerations of the K- vs. W-modeling relation will be discussed here.

It is well known that direct block-based modeling with K-methods is difficult or impossible due to the delay-free loop problem [4, 3, 7]. This can be understood by considering an electric component, such as a resistor, and how voltage and current are interrelated by  $V = RI$ . This is not a causal relation where one variable follows from another. Thus there is no order of computation  $V \rightarrow I$  or  $I \rightarrow V$ . However, if wave quantities are used as for DWGs in Section 2.1 or for WDFs in Section 2.2, there is a causal relation from wave input to wave output in a wave-based port. If that can be made explicitly computable, the delay-free problem can be avoided.

#### 3.2. Comparison of DWGs and WDFs

In a comparison between DWGs (Eq. (6-8) and Fig. 3) and WDFs (Eq. (9) and Fig. 7) we can see that by acoustical-to-electrical analogies  $P_i^+ = a$ ,  $P_i^- = b$ ,  $P_i = V$ ,  $U_i = I$ , and  $1/Y_i = R$ , for the DWG we get

$$\begin{cases} a = (V + RI)/2 \\ b = (V - RI)/2 \end{cases} \Leftrightarrow \begin{cases} V = (a + b) \\ I = (a - b)/R \end{cases} \quad (21)$$

which shows a difference in scaling compared to the WDF convention in (9). In fact we may select the scaling quite freely if we are interested just to get physically proper values of the Kirchhoff quantities  $V$  and  $I$ . One useful

<sup>4</sup>In the original K-S model the excitation was inserted as random numbers into the delay line and the loop filter was a simple two-tap FIR filter.

convention is to apply *power-normalized waves* [4, 5] by

$$\begin{cases} a = (V + RI)/2\sqrt{R} \\ b = (V - RI)/2\sqrt{R} \end{cases} \leftrightarrow \begin{cases} V = (a + b)\sqrt{R} \\ I = (a - b)/\sqrt{R} \end{cases} \quad (22)$$

which have the favourable property that port power  $P$

$$P = VI = a^2 - b^2 \quad (23)$$

is independent of port resistance  $R$ . This means that changing of  $R$  does not affect the energetic state of the WDF element, which is important in discrete-time modeling of nonlinear and time-variant systems.

In a further comparison between digital waveguides and wave digital filters we can find the following similarity. If the port admittances/impedances in DWG junctions (see Fig. 2) are real-valued, the DWG junctions and WDF adaptors implement the same computation of K-variables, thus they are just two different approaches to the same W-modeling principle. The formulation of DWGs above is sufficient for making parallel or series connections of two-way delay-lines and terminations by arbitrary impedances/admittances. It does not allow for combining subcircuits and networks hierarchically further by series and parallel connections, unless specific techniques are developed to avoid the delay-free loop problem. Wave digital filters are, however, developed particularly to allow such connections for arbitrary circuits and networks.

### 3.3. Arbitrary impedance as a part of W-model

To derive rules for attaching an arbitrary K-defined impedance to a wave port we can write:

$$a = V + RI, \quad b = V - RI, \quad \text{and} \quad Z = V/I \quad (24)$$

which can be solved for  $H(z)$  as a the ratio of  $b$  and  $a$  in the form

$$H(z) = \frac{b}{a} = \frac{V - RI}{V + RI} = \frac{Z - R}{Z + R} \quad (25)$$

Now it turns out that in general an adaptor with  $H(z)$  as port feedback is explicitly computable only when  $\hat{z}(n)$ , the inverse Z-transform of  $Z(z)$ , has a delay-free component  $\hat{z}(0) \neq 0$  and port resistance  $R$  is selected as  $R = \hat{z}(0)$ . We will next look at three common cases of impedance  $H(z)$  specification: Polynomial (FIR), rational (IIR), and a modal filterbank of second-order resonators.

**FIR type impedance:** When  $Z(z)$  is given as a Z-domain polynomial expression, i.e.,

$$Z_p(z) = \sum_{i=0}^{N-1} q_i z^{-i} \quad (26)$$

then the  $a$  to  $b$  scattering function  $H(z)$  will in a delay-containing case  $R = q_0$  become

$$H_p(z) = \frac{\sum_{i=0}^{N-1} q_i z^{-i} - R}{\sum_{i=0}^{N-1} q_i z^{-i} + R} = \frac{\sum_{i=0}^{N-2} \frac{q_{i+1}}{2q_0} z^{-i}}{1 + \sum_{i=1}^{N-1} \frac{q_i}{2q_0} z^{-i}} z^{-1} \quad (27)$$

which is an IIR filter cascaded with a unit delay.

**IIR type impedance:** When  $Z(z)$  is given as a Z-domain rational expression (for simplicity numerator and denominator orders are the same), i.e.,

$$Z_r(z) = \frac{\sum_{i=0}^{N-1} q_i z^{-i}}{\sum_{i=0}^{N-1} p_i z^{-i}} \quad (28)$$

then the delay-containing requirement for port resistance becomes  $R = q_0/p_0$  and  $H(z)$  according to Eq. (25) will be

$$H_r(z) = \frac{\sum_{i=0}^{N-2} \frac{q_{i+1} - Rp_{i+1}}{2Rp_0} z^{-i}}{1 + \sum_{i=1}^{N-1} \frac{q_i + Rp_i}{2Rp_0} z^{-i}} z^{-1} \quad (29)$$

which is again an IIR filter with a cascaded unit delay.

**Second order resonators:** Second order resonators are of special interest because they are useful in building modal decomposition models. For example a series RLC-resonator having poles at DC and Nyquist frequency can be simulated by

$$Z_s = K \frac{1 + q_1 z^{-1} + q_2 z^{-2}}{1 - z^{-2}} \quad (30)$$

that with  $R = K$  leads to realization

$$H_s(z) = 0.5 \frac{q_1 + (q_2 + 1)z^{-1}}{1 + 0.5q_1 z^{-1} + 0.5(q_2 - 1)z^{-2}} z^{-1} \quad (31)$$

### 3.4. Comparison of DWG vs. FDTD models

In [10] a careful analysis of DWGs and FDTDs (see Figs. 3 and 8) has been presented by proving their functional equivalence in processing K-variables related to these structures. The similarity is obvious when looking at the ‘scattering substructure’, i.e., summing of signals through admittances and inverse of the sum of the admittances. While in DWGs the delays in a waveguide structure are between scattering junctions, in the FDTD case the delays are implemented within the node itself by two unit delays. From this it is obvious that in two- or three-dimensional mesh-like structures the FDTD approach has the advantage of only two delays (memory positions) needed per node, while for the DWG mesh two delays are needed for each connection between junctions.

There are a couple of special questions related to the equivalence worth mentioning. The first one deals with the ‘sporadic’ or ‘spurious’ oscillations that easily appear in the FDTD structure but not in DWGs. The potential instabilities in the FDTD case are due to its inherent dual-delay feedback, see the bottom part of Fig. 8. This creates poles at DC and Nyquist frequency that need to be counteracted by the term  $H(z) = 1 - z^{-2}$  when feeding external excitation  $U_{ext}$ . This pole cancellation means lack of numerical robustness close to these frequencies. Without this term an impulse fed to the junction results in continuous Nyquist frequency oscillation plus step function propagating from the junction in an FDTD array, which means unbound generation of energy, thus being nonphysical in the sense of passive systems.

One may ask if the instability is possible in a DWG. It is only possible if the junction is fed through the inverse of  $H(z) = 1 - z^{-2}$ , i.e., through  $H(z) = 1/(1 - z^{-2})$ , which works as an external source to make continuous energy generation.

Another special case is the inherent integration property of an FDTD structure as utilized in [9] and [19]. If only the pole at the Nyquist frequency is cancelled by feeding the FDTD junction through  $H(z) = 1 + z^{-1}$ , the model is inherently integrating from velocity  $V(z)$  to displacement  $Y(z)$  by  $Y(z) = V(z)1/(1 - z^{-1})$ . This is useful for example in string modeling [19] when the desired pair of K-variables is force and displacement, instead of the ‘natural’ pair of force and velocity.

We can thus notice that DWGs in their basic form don’t integrate. If we desire to have that property, we must use an explicit integrator, approximated by  $H(z) = 1/(1 - z^{-1})$ , either in feeding the input to a junction, or when taking the velocity output from a junction. By such an external element we can again make the FDTD and DWG structures fully equivalent functionally.

### 3.5. Compatibility of DWG vs. FDTD models

In [10] a method is given to construct mixed models by combining FDTD and DWG elements (thus also WDFs) through a K-to-W converter, a two-port element shown in Fig. 10. The left-hand side shows an FDTD junction and

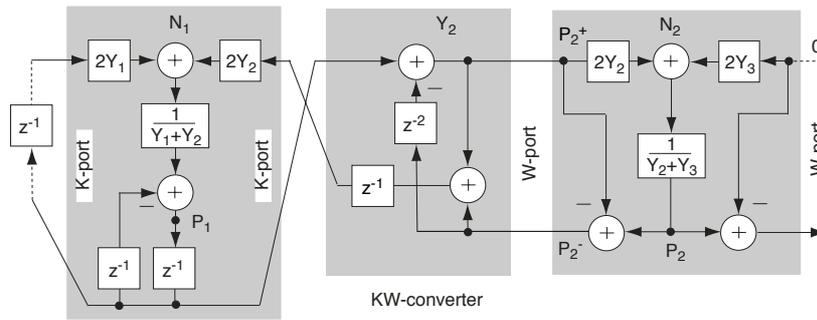


Figure 10: KW-converter for mixed modeling with FDTDs (left-hand side) and DWGs (right-hand side).

the right-hand side a DWG junction. The KW-converter maps the K-variable terminals of an FDTD junction to a wave port of a DWG junction and the other way around. In section 5.5 below an example of using this converter structure in a 2-D mesh is presented.

#### 4. REAL-TIME SIMULATION OF MIXED MODELS

The formulation of DWG, FDTD, WDF, etc., and mixed models above hints a way to construct computational physical models based on DSP structures. A software tool called the BlockCompiler has been developed that is particularly designed for flexible yet efficient experimentation with physics-based models [18].

Object-oriented specification and manipulation of blocks and their interactions in the BlockCompiler is implemented in the Common Lisp programming language. Automatic generation of C source code from interconnected block representation and compilation to a run-time executable provides efficient computation in real time or by sample-by-sample non-realtime simulation.

##### 4.1. BlockCompiler features

The main features and functional principles of the BlockCompiler are briefly described as follows:

- **Block structures and patches:** The basic level of block modeling supports DSP with directed data flow, such as adders, multipliers, delays, and digital filters, see Fig. 11 for a simple low-pass filter example. DSP blocks are instantiated by make-forms such as `(.coeff c1)` for a multiplier, `(.ad)` for sound input and `(.da)` for sound output, `(.d)` for a delay, and `(.add)` for an adder, etc. These are chained by forms `(-> (.ad) ...)` from output to input, output to input, etc. A full model is called a *patch*. The patch of Fig. 11 can be created interactively by scripting

```
(patch ((a (.add)))
  (-> (.ad) (.coeff 0.0666) (input a 0) (inputs (.da)))
  (-> a (.D) (.coeff -0.8668) (input a 1)))
```

The first line creates an adder bound to local variable `a` so that it can be referred twice in the two lines of connecting blocks. The model can be made streaming by command `run-patch`. Audio input is then continuously low-pass filtered to audio output. Inputs and outputs of blocks are normally used for synchronous data flow, but inputs may work also asynchronously for parametric control.

- **Macro blocks** can be defined as combinations of more elementary blocks. This is a useful abstraction mechanism whereby the details of the the new class objects are hidden and a hierarchical way of creating models is enabled.

- **Data types:** The BlockCompiler supports data types `{short, long, float, double}`, corresponding array types, as well as complex-valued data types.

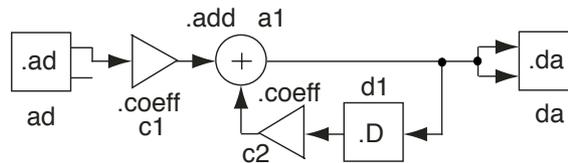


Figure 11: Simple low-pass filtering from sound input to output.

- **Multirate support:** Multirate processing is available so that each block can be given a relative sample rate (decimation and interpolation by integer ratio).
- **Code generation and compilation:** Before compilation the scheduling of operations within the blocks is carried out. Each block of a patch generates inline C code into a file. This file will define a function that carries out model computation in a sample-by-sample manner. The resulting C file is then compiled automatically by a call to a C compiler.
- **Runtime behavior:** While calling `run-patch`, the compiled file is linked and made streaming, being synchronized typically by audio-I/O. The whole process from scheduling to start of streaming takes only a couple of seconds for small models, and only tens of seconds for relatively complex models.

Further examples of BlockCompiler code are given below for some modeling cases.

## 5. MODELING EXAMPLES

Some cases of physics-based modeling of musical instrument synthesis and speech production are discussed in this chapter. This includes string instruments, speech production modeling, a wave digital bell, and a mixed modeling mesh structure that can represent for example a membrane of a drum. In the first cases the realization of the models in the BlockCompiler is also characterized.

### 5.1. Realization of a Karplus-Strong string model

A straightforward model of a string was described in Section 2.5 and depicted in Fig. 9. It consists of a triggerable wavetable (noise burst) and a (fractional) delay loop with a low-pass filter to generate the decaying string sound by a recursive comb filter. The model of Fig. 9 can be realized with the following patch:

```
(patch ((noise (qs::make-random-signal 10000.0
              :span (qs::make-span 0 882)))
        (wt (.wtable :data (qs::s-array noise)))
        (dlx (.delay :delay-time 0.01
                  :control '.ap1))
        (add (.add :inputs 2))
        (lp (.lp1))
        (da (.da)))
  (-> wt add dlx lp (input add 1) (inputs da)))
```

where local variables are: `noise` keeps a random noise burst of 882 samples (20 ms), `wt` is the wavetable of the noise samples, `dlx` keeps the fractional delay with all-pass filter type of length control (initial length 10 ms), `add` is an adder, `lp` is a first-order low-pass for string loss control, and `da` is sound output. These elements are interconnected by a single expression of input-output chaining in the last line of the code.

In the BlockCompiler it is easy to create simple control interfaces for the parameters of a real-time synthesis model. Figure 12 shows a graphic control panel that includes sliders for the loop delay (string length), two para-

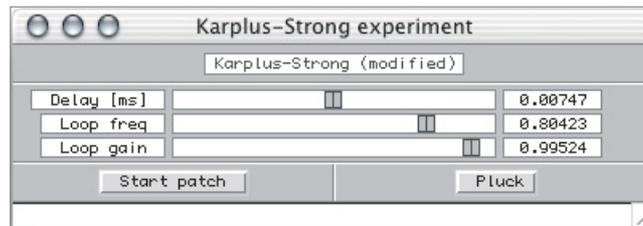


Figure 12: Control dialog for the Karplus-Strong model.

meters for loop filter to control the decay of string vibration, a button to start/stop running the patch, and a trigger button for the wavetable to pluck the string model.

## 5.2. DWG string instrument model

In Fig. 4 we have shown a block diagram of a 2-string DWG guitar where the strings are coupled through a common bridge impedance ( $z_b$ ). Each string consists of two delay lines with a pluck point node in between for inserting pluck excitation from a wavetable. The principle of creating a model for the guitar is characterized (only for the upper string) in BlockCompiler by script:

```
(patch ((zb (.Z :impedance *imped-zb*))
        (zt1 (.Z :impedance *imped-zt1*))
        (wt1 (.wtable *wt-data1*))
        (dl11 (.d-line :delay-length *length11*))
        (dl12 (.d-line :delay-length *length12*))
        ...))
(.ser zt (port dl11 0))
(-> wt1 (.ser (port dl11 1) (port dl12 0)))
(-> (.ser (port dl12 1) (port dl22 1) zb) da)
...)
```

Here the blocks are first instantiated (parameter data shown only symbolically) and bound to symbols used in Fig. 4. Then their ports are series connected by forms `(.ser ...ports...)` which return the node objects. Excitations and output are wired unidirectionally to and from node objects by `(-> ...)` forms.

Instead of building a full instrument model, such as a 6-string guitar, each string with two polarizations, as a single patch from basic elements, a better strategy is to define macro blocks, layer by layer. First a single polarization element, then two of them for a two-polarization string, further on using six of them, a bridge, and a body filter, to finally define a full guitar as a new macro block.

## 5.3. Modeling of speech production

The flexible properties of the BlockCompiler make it easy to create efficient simulation models also for speech production. Figure 13 illustrates an advanced DWG model, which follows traditional guidelines of transmission-line modeling. The tract is divided into a set of constant length sections, whereby acoustic admittances  $Y(i)$  can be controlled according to their cross-sectional area dependency  $Y(i) = A(i)/\rho c$ , where  $\rho$  is air density and  $c$  is speed of sound. Parametric control of vocal tract shape can be based for example on mapping from articulatory parameters, or by contextual lookup and interpolation in time. Fine-tuning of the tract length can be made at the lips by a single controllable fractional delay-line section.

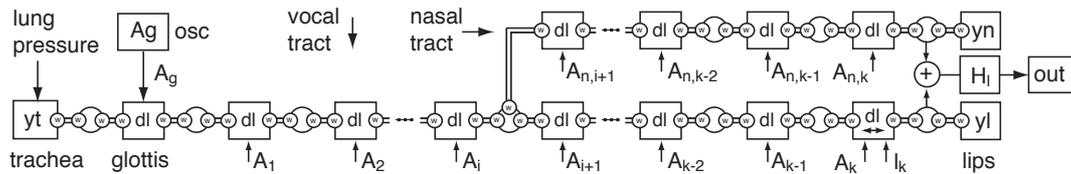


Figure 13: DWG transmission-line speech production model, including nasal-tract, made of (mostly) constant length sections.

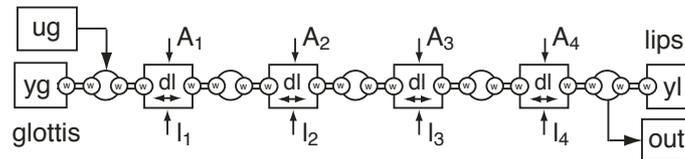


Figure 14: Speech production using vocal-tract sections of variable length ( $l_i$ ) and cross-area ( $A_i$ ). (No nasal tract is shown.)

The glottis is realized as a single section of vocal tract with varying area, controlled by a glottal waveform oscillator. Lung pressure makes the volume flow through the glottis in relation to its opening. More advanced nonlinear models of self-oscillation can also be used.

The termination in the model of Fig. 13 at lips and nostrils includes a filter  $H_l$  for detailed lip pressure to far-field radiation function. Other functionalities that can be experimented relatively easily are for example generation of turbulence friction and bursts in constrictions and during the opening of occlusion.

Another type of solution is depicted in Fig. 14 where the vocal tract is divided into variable length sections. 4–5 such sections can approximate the tract in an articulatory approach so that the sections inherently correspond to moving parts of the articulators, such as the tongue body. This principle was introduced originally in [20], and it is computationally efficient enough due to flexible control of fractional delay lines in the BlockCompiler. With additional impedances at junctions the principle can simulate also tract models with conical sections.

## 5.4. Wave digital bell

As a W-modeling case using modal decomposition, a ‘semiphysical’ driving point impedance of a bell is investigated next. Sound synthesis of bells has earlier been investigated using inharmonic digital waveguides and modal filterbanks [21, 22]. High-resolution analysis of modal data was achieved using the frequency-zooming ARMA analysis (FZ-ARMA) technique as developed in [23]. This is able to resolve for each partial a set of modes, even when very close in frequency, which produces the beating inherent in typical bell sounds.

Real bells are physical objects that, as 3-D structures, have an inharmonic set of partials [1]. By proper tuning several of the lower partials can be made approximately harmonic, but there always remains at least one perceptually important inharmonic component. In untuned small handbells there may not be any harmonic structure at all. Slight asymmetries are the reason why partials are mode groups, thus resulting in perceivable beating (warble).

In a detailed model of a real bell the modal decomposition should describe the spatial distribution of modal shapes so that a force excitation in any point could be solved for the spatial distribution of the exponentially decaying modes as well as for the sound radiated from bell surfaces. Instead we in this paper discuss a wave digital model based on modal decomposition that considers the bell only as a driving point impedance with related force and velocity. We may calibrate such a semiphysical model simply according to a sound recording of an existing bell and implement a model that sounds realistic, being efficient for real-time synthesis.

We have taken a particular bell and analyzed its prominent modal components using the FZ-ARMA analysis.

Table 1: Modal data for the bell of the case study including two modes per partial: mode frequencies ( $f_1$  and  $f_2$ ), initial amplitudes ( $A_1$  and  $A_2$ ), and decay time constants ( $\tau_1$  and  $\tau_2$ ) are given.

$n$	$f_1/\text{Hz}$	$\tau_1/\text{s}$	$A_1$	$f_2/\text{Hz}$	$\tau_2/\text{s}$	$A_2$
1	850.8	0.165	0.0723	851.3	0.749	0.0965
2	1702.3	0.464	0.1497	1703.1	0.421	0.0514
3	2026.7	0.355	0.1258	2032.8	0.048	0.0734
4	2787.2	0.131	0.0763	2792.5	0.079	0.0364
5	3404.7	0.251	0.0610	3407.0	0.098	0.0716
6	4552.1	0.028	0.0290	4559.6	0.110	0.0278
7	4889.6	0.149	0.0554	5050.5	0.259	0.0511
8	6881.5	0.149	0.1261	6889.2	0.051	0.0088
9	8549.8	0.153	0.0029	8631.9	0.023	0.0047
10	8695.0	0.109	0.0313	8842.0	0.153	0.0191

Table 1 lists the modal data from a bell recording<sup>5</sup>. Two modes are fitted to each partial below 10 kHz.

For a wave-based modeling we may use the analyzed modal data by constructing a wave port with a corresponding driving-point impedance. The task is now to construct a port compatible with W-modeling that implements a driving point impedance so that when the bell is hit by a proper hammer-like object (force excitation), it makes a desired sound (port velocity as output).

There are several ways to construct the port impedance of interest, for example:

1. Make the modes by basic WDF components as series resonators, as shown in Fig. 7. Pairs of modes are then combined in parallel into partials, which are further connected in parallel to make a full port impedance.
2. Implement each mode or a mode group for a partial as a filter structure according to the rules described in Section 3.3, and connect the partials in parallel.
3. Realize the whole impedance, composed of all modes, as a single filter, according to Eq. (29).

It is obvious that progressing from case 1 to 3 the computational efficiency of implementation can be improved, since consolidating functionality can utilize the advantages of DSP more than composing a model from lumped WDF elements. Case 2 benefits from the possibility of controlling the properties of each mode separately, while in case 3 only the composite filter coefficients are directly accessible. In case 1 the equivalent lumped components, corresponding in some way to masses, spring constants, and damping factors, are directly controllable.

One particular design issue with case 1 is the bilinear mapping when creating reactances with WDFs. The formulas of reference resistance for capacitance  $C$  and inductance  $L$  in electrical circuits from Eqs. (12) and (14) are:

$$R_C = 1/2Cf_s \quad \text{and} \quad R_L = 2Lf_s \quad (32)$$

where  $R_C$  and  $R_L$  are the related WDF port resistances and  $f_s$  is the sample rate. However, due to the bilinear mapping between analog angle frequency  $\omega = 2\pi f$  and discrete-time angle frequency  $\Omega$  by

$$\omega = g(\Omega) = (2/T_s) \tan(\Omega T_s/2) \quad (33)$$

the frequency scale is warped [4] so that while  $\Omega \rightarrow f_s/2$ , the corresponding analog angle frequency  $\omega \rightarrow \infty$ . Thus the inductances and capacitances in a WDF realization have to be prewarped to get correct modal frequencies and decay times. In cases 2 and 3 above the filter design process takes automatically care of proper frequencies and decay rates. In each case 1 to 3 the wave digital port realized is functionally equivalent.

Figure 15 shows the admittance function (dB scaled) of the port implemented from data in Table 1, and Fig. 16 plots the temporal envelope of partial number 3 showing beating in decay.

<sup>5</sup>This is a relatively high-pitch bell from the Belfort bell recordings, provided by Mark Leman.

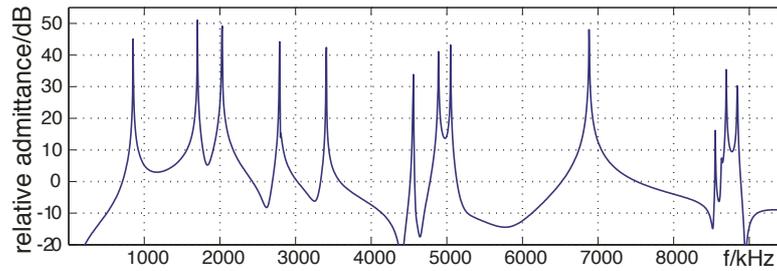


Figure 15: Synthesized bell driving point port admittance function.

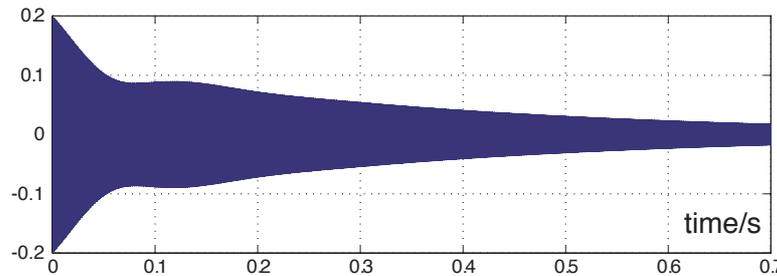


Figure 16: Temporal envelope of the third partial in a bell sound.

To make a full bell model with a hammer striking the driving point, the hammer has also to be modeled. The contact is a nonlinear (time-varying) one, thus special techniques are needed to make it energetically correctly and to guarantee stability. There are two principles available to this: using power-normalized waves discussed above (see also [24]) or by nonlinear reactance through mutator type of adaptor [3]. These techniques allow for complex nonlinear contact of the hammer and the bell, like it happens also with the piano hammer [24].

The WDF bell with impulsive hit sounds different from the recording used for calibration for two reasons. First, the attack needs a more complex hammer interaction model for more realistic sound, and secondly, reverberation and reflections of the space surrounding the bell. In other aspects the timbre is very realistic, including correct type of beating and decay envelope.

### 5.5. Mixed FDTD + DWG modeling of a 2-D mesh

Figure 17 illustrates a part of a 2-D mixed model structure that is based on a rectangular FDTD waveguide mesh for efficient memory-saving computation and DWG elements at boundaries. Such a model could be for example a membrane of a drum or in a 3-D case a room enclosed by walls. When there is need to attach W-type termination admittances to the model or to vary the propagation delays within the system, a change from K-elements to W-elements through converters is a useful property. Furthermore, variable-length delays can be used, e.g., for passive nonlinearities at the terminations to simulate gongs and other instruments where nonlinear mode coupling takes place [25]. The same principle can be used to simulate shock waves in brass instrument bores [26]. In such cases the delay lengths are made dependent on the signal value passing through the delay elements.

In Fig. 17 the elements denoted by  $kp$  are K-type pipes (delay-free connections) between K-type nodes. Elements  $kw$  are K-to-W converters and elements  $wl$  are W-lines (delay-lines), where the arrows indicate that they are controllable fractional delays. Elements  $yt$  are terminating admittances. In a general case, scattering within the mesh can be controlled by varying the wave admittances (not shown in the figure), although the computational efficiency is improved if the wave admittances are made equal. In a modern PC a 2-D mesh of a few hundred elements can run in real time at full audio rate. By decimated computation, bigger models can be computed if a lower cutoff frequency is permitted, allowing larger physical dimensions of the mesh.

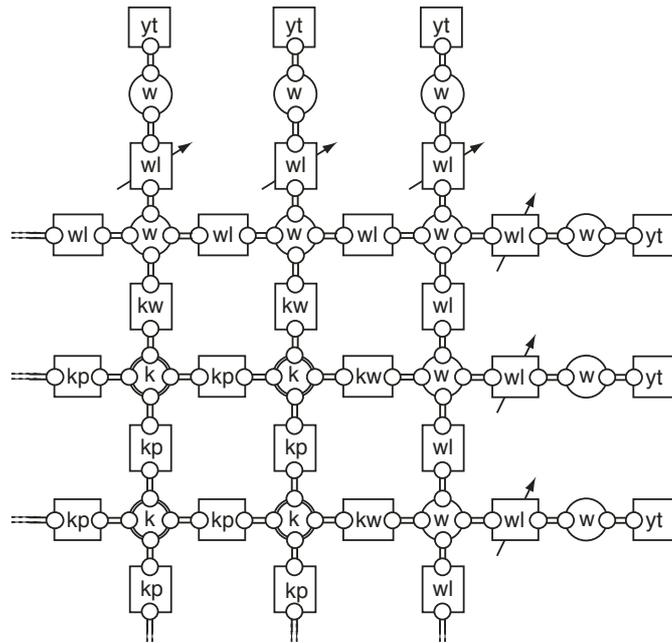


Figure 17: Part of a 2-D waveguide mesh composed of (a) K-type FDTD elements (left bottom): K-pipes ( $kp$ ) and K-nodes ( $k$ ), (b) W-type DWG elements (top and right): delay-controllable W-lines ( $wl$ ), W-nodes ( $w$ ), and terminating admittances ( $yt$ ), and (c) converter elements ( $kw$ ) to connect K- and W-type elements into a mixed model.

## 6. SUMMARY

This paper has presented an overview of different physics-based modeling paradigms, their interrelationships, and how they are used in modeling various types of sound sources, particularly musical instruments. The modeling paradigms include digital waveguides (DWGs), wave digital filters (WDFs), finite difference time domain models (FDTDs), modal decomposition techniques, and source-filter models. Real-time simulation techniques are briefly described and models of strings, membranes, bells, and the human vocal tract are described.

## 7. ACKNOWLEDGMENTS

This paper presents results from the following projects: IST/ALMA (ALgorithms for the Modelling of Acoustic interactions, IST-2000-30072) and Academy of Finland project ‘Technology for Audio and Speech Processing’ (53537). Special thanks are due to Cumhuri Erkut, Paulo A. A. Esquef, Vesa Välimäki, Lauri Savioja, and Hannu Pulakka.

## 8. REFERENCES

- [1] N. H. Fletcher and T. D. Rossing, *The Physics of Musical Instruments*, Springer-Verlag, New York, 1991.
- [2] A. Oppenheim and R. Schaffer, *Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1975.
- [3] A. Sarti and G. de Poli, “Towards Nonlinear Wave Digital Filters,” *IEEE Trans. on Sig. Proc.*, vol. 47, no. 6, pp. 1654–1668, June 1999.
- [4] A. Fettweis, “Wave Digital Filters: Theory and Practice,” *Proc. of the IEEE*, vol. 74, no. 2, pp. 270–361, Feb. 1986.

- [5] Julius O. Smith, *Principles of Digital Waveguide Models of Musical Instruments*, Chapter 10 in *Applications of Digital Signal Processing to Audio and Acoustics*, (ed. Kahrs and Brandenburg), Kluwer Academic Publishers, 1998.
- [6] L. Savioja, *Modeling Techniques for Virtual Acoustics*. PhD thesis, Helsinki Univ. of Tech., Espoo, Finland, 1999.
- [7] S. D. Bilbao, *Wave and Scattering Methods for the Numerical Integration of Partial Differential Equations*, PhD Thesis, Stanford University, May, 2001.
- [8] J. Strikverda, *Finite Difference Schemes and Partial Differential Equations*, Wadsworth and Brooks, Grove, Ca, 1989.
- [9] A. Chaigne, "On the Use of Finite Differences for Musical Synthesis. Application to Plucked Stringed Instruments," *J. Acoustique*, vol. 5, pp. 181–211, April 1992.
- [10] M. Karjalainen and C. Erkut, "Digital Waveguides vs. Finite Difference Schemes: Equivalence and Mixed Modeling," *J. Applied Sig. Proc.*, to appear in 2004.
- [11] J.-M. Adrien, *The Missing Link: Modal Synthesis*, Chapter in *Representations of Musical Signals*, (eds. de Poli, Piccialli, and Roads), MIT Press, Cambridge, Mass., 1991.
- [12] L. Trautmann, *Digital Sound Synthesis by Physical Modeling of Musical Instruments Using Functional Transformation Methods*, PhD Thesis, Univ. Erlangen-Nürnberg, Germany, 2002.
- [13] <http://www.ircam.fr/produits/logiciels/modalys-e.htm>
- [14] G. Fant, *Acoustic Theory of Speech Production*, Mouton and Co., 1960.
- [15] J. D. Markel and A. H. Gray, *Linear Prediction of Speech*, Springer Verlag, New York, 1976.
- [16] K. Karplus and A. Strong, "Digital Synthesis of Plucked-String and Drum Timbres," *Computer Music J.*, vol. 7, nr. 2, pp. 43–55, 1983.
- [17] M. Karjalainen, V. Välimäki, and T. Tolonen, "Plucked String Models: from Karplus-Strong Algorithm to Digital Waveguides and Beyond," *Computer Music J.*, vol. 22, no. 3, pp. 17-32, 1998.
- [18] M. Karjalainen, "BlockCompiler: Efficient Simulation of Acoustic and Audio Systems," *Preprints of AES114th Convention*, Paper 5756, Amsterdam, May 2003.
- [19] M. Karjalainen, "1-D Digital Waveguide Modeling for Improved Sound Synthesis," in *Proc. IEEE ICASSP'02*, Orlando, May 2002, pp. 1869–1872.
- [20] V. Välimäki, M. Karjalainen, and T. Kuusma, "Articulatory Control of a Vocal Tract Model Based on Fractional Delay Waveguide Filters," *Proc. IEEE ISSIPNN*, pp. 571-574, Hong Kong, April 1994.
- [21] M. Karjalainen, P. A.A. Esquef, and V. Välimäki, "Efficient Modeling and Synthesis of Bell-Like Sounds," in *Proc. DAFx'02*, Hamburg, Germany, Sept. 2002, pp. 181–186.
- [22] M. Karjalainen, V. Välimäki, and Paulo A.A. Esquef, "Making of a Computer Carillon," in *Proc. SMAC'03*, Stockholm, Sweden, Aug. 2003, pp. 181–186.
- [23] M. Karjalainen, P. A. A. Esquef, P. Antsalo, A. Mäkivirta, and V. Välimäki, "Frequency-Zooming ARMA Modeling of Resonant and Reverberant Systems," *J. Audio Eng. Soc.*, vol. 50, no. 12, pp. 1012–1029, Dec. 2002.

- [24] J. Bensa, S. Bilbao, R. Kronland-Martinet, and J. O. Smith, "A Power Normalized Non-Linear Lossy Piano Hammer," in *Proc. SMAC'2003*, Stockholm, Sweden, 2003, pp. 365–368.
- [25] J. R. Pierce and S. A. Van Duyne, "A passive nonlinear digital filter design which facilitates physics-based sound synthesis of highly nonlinear musical instruments," *J. Acoust. Soc. Am.*, vol. 101, pp. 1120–1126, Feb. 1997.
- [26] R. Msallam, S. Dequidt, S. Tassart, and R. Causse, "Physical model of the trombone including nonlinear propagation effects," in *Proc. Int. Symposium on Musical Acoustics (ISMA)*, (Edinburgh, UK), pp. 245–250, Sept. 1997.