COMPILATION OF UNIFIED PHYSICAL MODELS FOR EFFICIENT SOUND SYNTHESIS

Matti Karjalainen[1], *Cumhur Erkut*[1], *and Lauri Savioja*[2]

Helsinki University of Technology
[1]Laboratory of Acoustics and Audio Signal Processing
[2]Laboratory of Telecommunication Software and Multimedia
FIN-02015 HUT, Finland
{firstname.lastname}@hut.fi

ABSTRACT

This paper describes a systematic approach to specification and compilation of different physical modeling schemes particularly for sound synthesis studies. First we formulate theoretically a unified way of constructing physical interaction models which include elements that use both wave variables and Kirchhoff variables. These elements can be applied to build 1-D and multidimensional structures as well as lumped element models. In addition, typical signal processing algorithms are supported. A software environment (Block Compiler) has been developed, allowing for high-level object-based specification of physical models and their compilation to efficient code for execution.

1. INTRODUCTION

Physical modeling and model-based sound synthesis of musical instruments and other sound sources have become an important part of computer music and modern audio [11]. Different approaches and modeling techniques have been proposed to describe and realize spatiotemporal physical behavior that is found in musical instruments. For efficient computation these models are formulated through DSP algorithms.

While in abstract synthesis through DSP algorithms the interaction and signal flow are mostly directed (one-directional), in more physical approaches, such as digital waveguides [11], wave digital filters [5, 9, 1], finite difference models [12, 7, 2], etc., the interaction is two-directional. In such cases we can make distinction between models with *Kirchhoff variables* (K-variables) and *wave variables* (W-variables). This distinction is based on the fact that in a Kirchhoff formulation the total observable variable is considered, while with wave variables the Kirchhoff variable is divided into one (or more) directed wave component pair.

As a continuation of studies in [8, 3, 4], in this paper we are particularly interested to combine K-models and W-models in a coherent way while building practical models. Finite difference time domain (FDTD) models are basically physical models where a Kirchhoff variable, such as force or pressure, is taken as a free physical variable and the complementary one (such as velocity or volume velocity) is determined through a parameter, such as impedance Z or its inverse, admittance $G = 1/Z$. The relation of FDTD models and wave-based models is discussed and the combination of such submodels is studied in particular. The utilization of wave digital filter (WDF) components is also discussed.

The organization of the paper is as follows. First we compare the wave-based digital waveguides and their FDTD counterparts. Then we connect them together in a seamless way, including scattering junctions and multidimensional structures. Lumped element modeling is also discussed briefly, and finally a software environment is described which supports high-level object-based specification of physical models and their compilation to efficient code for real-time and non-realtime execution.

2. WAVE-BASED VS. FDTD MODELS

In a one-dimensional lossless medium the wave equation is written

$$y_{tt} = c^2 y_{xx} \tag{1}$$

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 μ is mass per unit length of the string [6].

The two common forms of discretizing the wave equation for numerical simulation are through traveling wave solution and by finite difference formulation.

2.1. Wave-based modeling

The traveling wave formulation is based on the d'Alembert solution of propagation of two opposite direction waves, i.e.,

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

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 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 [11]:

$$y(n,m) = \overrightarrow{y}(n - m) + \overleftarrow{y}(n + m) \tag{3}$$

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

$$\overrightarrow{y}_{k,n+1} = \overrightarrow{y}_{k-1,n} \quad \text{and} \quad \overleftarrow{y}_{k,n+1} = \overleftarrow{y}_{k+1,n} \tag{4}$$

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 [11].

The next step is to take into account the global physical constraints of continuity by Kirchhoff rules. This means to formulate the scattering junctions of interconnected ports, with given impedances and wave variables at related ports. For a scattering junction of Fig. 1, when the physical variables force F and velocity V are used and a series junction[1] impedance model of N ports is utilized [11], the Kirchhoff constrains are

[1]Parallel junctions and admittance models are not discussed for brevity.

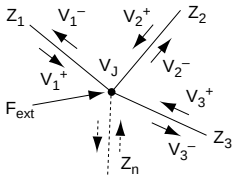

Fig. 1. Series junction of impedances Z_i with associated velocity waves indicated. A direct force input $F_{\rm ext}$ is also attached.

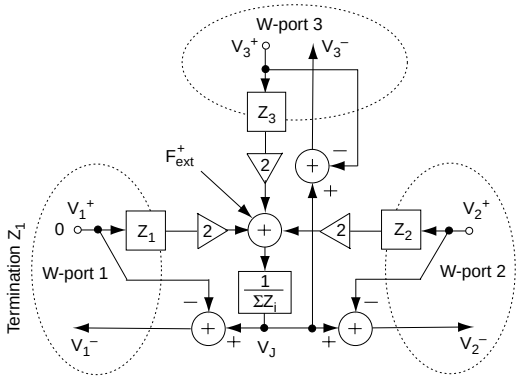

Fig. 2. N-port scattering junction (three ports are shown) of ports with impedances Z_i. Incoming velocities are V_i^+ and outgoing velocities V_i^+. W-port 1 terminated by impedance Z_1.

$$V_1 = V_2 = \ldots = V_N = V_{\rm J} \tag{5}$$

$$F_1 + F_2 + \ldots + F_N = F_{\rm ext} \tag{6}$$

where $V_{\rm J}$ is the common velocity of coupled branches and $F_{\rm ext}$ is an external force to the junction. When port velocities are represented by incoming wave components V_i^+, outgoing wave components by V_i^-, impedances attached to each port by Z_i, and

$$V_i = V_i^+ + V_i^- \quad \text{and} \quad F_i^+ = Z_i V_i^+ \tag{7}$$

the junction velocity $V_{\rm J}$ can be obtained [11] as:

$$V_{\rm J} = \frac{1}{Z_{\rm tot}}(F_{\rm ext} + 2\sum_{i=0}^{N-1} Z_i V_i^+) \tag{8}$$

where $Z_{\rm tot} = \sum_{i=0}^{N-1} Z_i$ is the sum of all impedances to the junction. Outgoing velocity waves, obtained from Eq. (7), are then $V_i^- = V_{\rm J} - V_i^+$. The result is illustrated in Fig. 2. When impedances Z_i are frequency-dependent, this diagram can be interpreted as a filter structure where the incoming velocities are filtered by the corresponding wave impedances Z_i times two, and their sum is filtered further by $1/Z_{\rm tot}$ to get the junction velocity $V_{\rm J}$. Displacement is obtained, for example in string modeling, by integrating the junction velocity $V_{\rm J}$ in time.

Two special cases can be mentioned based on Eq. (8). First, a (passive) loading impedance is the case with Z_i where no incoming velocity wave V_i^+ is associated. This needs no computation except including Z_i in $Z_{\rm tot}$ because the incoming wave is zero, see the left-hand termination in Fig. 2. Another issue is the external force $F_{\rm ext}$ effective to the junction. This is connected directly to the summation at the junction as shown in Fig. 2.

The wave variables and impedances at ports attached to a junction can be specified in any proper transform domain, but we are here interested in z-domain formulations for practical discrete-time computation. Notice that the impedances in Fig. 2 can be real-valued or frequency-dependent so that Z_i and the admittance $1/\sum Z_i$ can be realized as FIR or IIR filters, or just as real coefficients if all attached impedances are real. In the latter case, if we skip the external force $F_{\rm ext}$ of Eq. (8), we may write the equation using scattering parameters α_i as $V_{\rm J} = \sum_{i=0}^{N-1} \alpha_i V_i^+$, where $\alpha_i = 2Z_i/Z_{\rm tot}$. This and other special forms of scattering are efficient computationally when impedances are real-valued, but in a general case it is practical to implement computation as shown in Fig. 2 so that the term $1/\sum Z_i$ is a common filter.

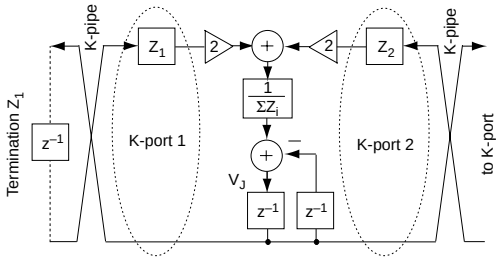

Fig. 3. Digital filter structure for finite difference approximation of a two-port scattering node with port impedances Z_1 and Z_2. Only total velocity V_J (K-variable) is explicitly available.

The freedom to use any impedance formulation allows also for applying measured data, such as bridge impedance/admittance as a part of an instrument model. The passivity condition is, as for a scattering junction in general, that Z_i are positive real. Notice also that the realization of junction nodes as shown in Fig. 2 is general for any linear and time invariant system approximation, also for 2-D and 3-D mesh structures.

2.2. Finite difference modeling

In the most common way to discretize the wave equation by finite differences the partial derivatives in Eq. (1) are approximated by second order finite differences

$$y_{xx} \approx (2y_{x,t} - y_{x-\Delta x,t} - y_{x+\Delta x,t})/(\Delta x)^2 \tag{9}$$

$$y_{tt} \approx (2y_{x,t} - y_{x,t-\Delta t} - y_{x,t+\Delta t})/(\Delta t)^2 \tag{10}$$

By selecting the discrete-time sampling interval Δt to correspond to spatial sampling interval Δx, i.e., $\Delta t = c\Delta x$, and using index notation $k = x/\Delta x$ and $n = t/\Delta t$, Eqs. (9) and (10) result in

$$y_{k,n+1} = y_{k-1,n} + y_{k+1,n} - y_{k,n-1} \tag{11}$$

which is a special case of multidimensional meshes as an FDTD formulation [11, 10]. From form (11) 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.

The equivalence of digital waveguides and FDTDs [1], although being computationally different formulations, is also applicable to expand Eq. (11) to a scattering junction with arbitrary port impedances. Figure 3 depicts one scattering node of a 1-D FDTD waveguide and the way to terminate one port by impedance Z_1. There can be any number of ports attached also here as for a DWG junction.

An essential difference between DWGs of Fig. 2 and FDTDs of Fig. 3 is that while DWG junctions are connected through 2-directional delay lines (*W-lines*), FDTD nodes have two unit delays of internal memory and delay-free *K-pipes* connecting ports between nodes. These junction nodes and ports are thus not directly compatible (see next subsection). One further difference,

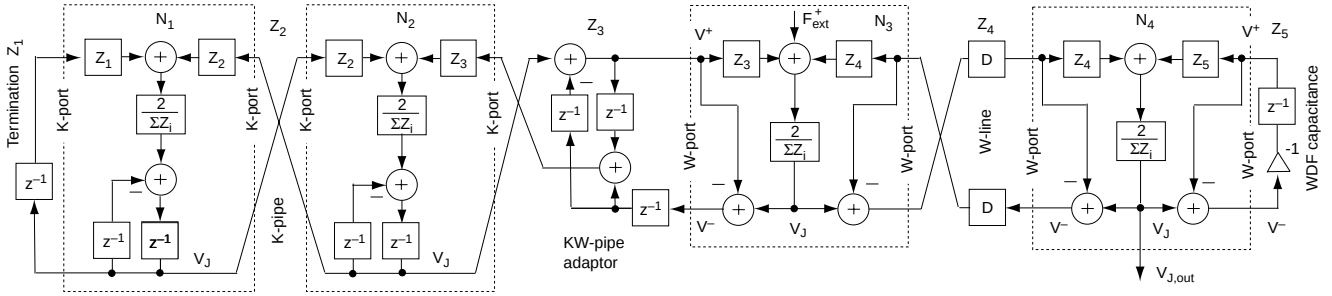

Fig. 4. FDTD elements (left) and DWG elements (right) forming a hybrid waveguide. Z_i are wave impedances of W-lines (for DWGs) and K-pipes (for FDTDs) between junction nodes (delimited by dashed lines). V_J are junction velocities, V^+ and V^- are wave components. Terminations: Z_1 (left) and WDF capacitor Z_5 (right). Delays marked by D may be of integer or fractional length. In: F^+_{ext}, Out: $V_{J,\mathrm{out}}$.

in addition to algorithmic and computational precision properties, is the possibility of 'spurious' responses in FDTDs, i.e., an initial state of finite energy may generate waves of infinitely expanding energy in them [12]. This 'non-physical' behavior needs extra computation in DWGs to create a similar behavior.

2.3. Interfacing of DWGs and FDTDs

The next question to discuss is the possibility to interface wave-based and FDTD-based submodels. In [4] it was shown how to interconnect a lossy 1-D FDTD waveguide with a similar DWG waveguide into a hybrid model using a proper interconnection element (adaptor). In a similar way, it is possible to make an arbitrary hybrid model of K-elements (FDTD) and W-elements having arbitrary wave impedances at their ports.

Figure 4 shows how this is done in a one-dimensional modeling case, but each junction node might connect any number of related ports as well, making also 2-D and 3-D meshes possible. The left-hand sybsystem in Fig. 4 is an FDTD waveguide through ports of specified wave impedances, and the right-hand part is a similar formulation for a wave-based model (DWG).

The function of the KW-pipe in the middle of Fig. 4 between the FDTD node N_2 and DWG element N_3 is to adapt the K-type port of an FDTD node and the W-type port of a DWG node, and it is delay-free. The proper functioning of the adaptor can be shown by testing the propagation of a left- and a right-traveling impulse through the adaptor. The equivalence and interfacing rules of wave-based and K-variable based (FDTD) models allow now for implementing mixed models where either of the approaches can be selected according to which one is more useful in the problem at hand.

2.4. Including lumped and nonlinear elements

As noted above, any linear and time-invariant system (1-D, multidimensional, or unstructured collection) of properly connected blocks can be computed using the described formalism. A useful additional formalism is to adopt Wave Digital Filters (WDF) [5, 1] as discrete-time simulators of lumped parameter elements. Based on wave variables, they are computationally fully compatible with the structures described above. A WDF *resistor* does not add much to systems above, but WDF *capacitors* and *inductors*, as well as *ideal transformers* and *gyrators*, are useful components [5]. As a physically bound choice for the case of this study, a WDF capacitor is a feedback from V^- wave of a port back to V^+ through a unit delay and coefficient -1, and having a port impedance $1/2f_sC$, see the right-hand termination in Fig. 4. A WDF inductor is a feedback through a unit delay and having a port impedance $2f_sL$. Here C is capacitance, L is inductance, and f_s is the sample rate (cf. [1]). A beneficial property of these elements is, since their wave impedances are real-valued, that junctions of such ports remain memoryless in the sense of Fig. 2, i.e., Z_i and $1/\sum Z_i$ are real. On the other hand, more flexibility is achieved by higher order approximations of Z_i than with WDF components.

The WDF formulation helps more essentially in another problem that appears when nonlinearities or fast parametric changes in a system are to be modeled, where delay-free loops may appear, requiring special solutions. Simply inserting an extra delay makes the model non-physical and is a source of potential instability. Several solutions to this problem have been proposed, one of them being to connect a nonlinear component through a WDF adaptor so that a delay-free loop is eliminated and the structure has the same energetic behavior as the corresponding analog system [9]. This problem is, however out of the scope of this paper.

3. BLOCK COMPILER: OBJECT-BASED REALIZATION OF MIXED K- AND W-MODELS

Different discrete-time approximations of wave behavior have much in common and each of them has its advantages and drawbacks [1]. The aim of the present study was to formulate a unified approach to W- and K-modeling and to make efficient simulation and synthesis possible from high-level specifications.

The best features of two different programming languages were combined in the implementation of the *Block Compiler*. Common Lisp and CLOS object system was used as a high-level symbolic processing environment and the C language for efficient numeric computation. A short description of the Block Compiler implementation is:

- A full model specification is called a *patch*, which consists of interconnected *block-item* units. Table 1 lists the code specifying the waveguide structure of Fig. 4. It contains from the left a termination Z_1, an FDTD (K-node) N_1, a K-pipe with impedance Z_2, another K-node N_2, an adaptor (KW-pipe) Z_3, a DWG node (W-node) N_3, a fractional delay-line Z_4, another DWG node N_4, and a WDF capacitance Z_5.
- The **patch*** macro (Table 1) first instantiates the computational blocks: K-pipes, W-lines, series connection nodes between them, and terminating impedances, and binds these objects to local variables for further reference. Then the generic function **connect** is used to implement the interconnectivity between block ports. Connecting a line/pipe port to a node creates a proper type of port for the node.

Table 1. Lisp code specification of hybrid waveguide in Figure 4.

```
(patch* ((n1 (.K-node)) (n2 (.K-node)) ;; nodes 1&2
         (n3 (.W-node)) (n4 (.W-node)) ;; nodes 3&4
         (k2 (.k-pipe :impedance 1.0)) ;; pipe Z2
         (kw (.kw-line :impedance 2.0)) ;; line Z3
         (w4 (.w-line :impedance 1.0 :length 10.5))
         (z1 (.Z :impedance 10.0)) ;; termination
         (z5 (.C :capacitance 1.0e-3)))  ;; WDF
  (connect (port k2 0) n1) (connect (port k2 1) n2)
  (connect (port kw 0) n2) (connect (port kw 1) n3)
  (connect (port w4 0) n3) (connect (port w4 1) n4)
  (connect (port z1) n1) (connect (port z5) n4)
  (connect (output (.AD) 0) (force n3)) ;; input
  (connect (velocity n4) (inputs (.DA)))) ;; output
```

Finally the sound driver input (.AD) and output (.DA) are connected to the waveguide for real-time streaming. (There are shorthand notations available to make the scripting less verbose.)

- Multirate processing is available so that each block can be given a relative sample rate.
- Macro blocks can be defined as containers of more elementary blocks. Macro definitions are flexible so that parameters can be given to specify the details of a macro block during instantiation.
- Data types $\{short, long, float, double\}$ and corresponding array types are available for signal data. Data is normally transferred between blocks in port-related global variables (synchronous data-flow). This can be optimized further by using register variables. While data flow is (multirate) synchronous, there is parameter control flow available that may be asynchronous as well.
- When a given model specification (e.g., Table 1) is evaluated as a Lisp script, an object-based patch is created and memory is allocated for data structures. Generation of executable code is done in the next steps:
- The patch is scheduled by walking the hierarchical structure and ordering the elementary operations. If there are delay-free loops or illegal structures, an error is reported.
- Each block writes inline C code into a file to create a single function with related data and declarations. C code generation of each block class is defined in 'pseudo-C' which looks like C code but data references are to Lisp. The resulting C file is then compiled by a call to a C compiler.
- The function pointer of the compiled code is taken and connected to the sample stream of the sound driver for real-time processing, or it can be called in a single step mode.
- While real-time streaming is running, the patch is fully controllable from Lisp, allowing for highly flexible control and inspection of the model behavior.

The object organization of blocks in Figs. 2-4 characterizes the object structure in BlockCompiler, i.e., the partitioning of data and operations. A delay line element in a DWG is nothing but delaying data and delegation of impedance changes to connected ports in the case of parametrically controlled model (this is not shown here). The same holds for pipes between FDTD nodes except that data transfer is delay-free.

The principles introduced above can be applied to form arbitrary networks, also multidimensional ones, of linear and time-invariant structures, by combining the DWG, FDTD, and WDF approaches.

The BlockCompiler prototype system works presently only on the Macintosh OS X operating system, but all software components (Lisp, C compiler, PortAudio sound driver) are available for other major platforms as well. The system has been tested both by various DSP-oriented and physical modeling tasks. It is found highly interactive due to simple scripting of model definitions and fast compilation to efficient run-time code. It is a flexible tool for basic research and application development. Simulation examples are available at 'www.acoustics.hut.fi/software/BlockCompiler'.

3.1. Future tasks

There is presently no graphic editor for visual programming of models. Writing a graphic editor is relatively straightforward but tedious. Such an editor is important to users that will do easy modeling using existing pre-defined blocks, although a 'power-user', able to write Lisp code, can do much more advanced modeling by textual programming.

The most essential theoretical future step will be to study systematically how to specify and automatically compile models with nonlinear elements.

4. ACKNOWLEDGMENT

This work is supported by the IST/ALMA project (ALgorithms for the Modelling of Acoustic interactions, IST-2000-30072).

5. REFERENCES

[1] S. D. Bilbao, *Wave and Scattering Methods for the Numerical Integration of Partial Differential Equations*, PhD Thesis, Stanford University, May 2001.

[2] 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.

[3] C. Erkut and M. Karjalainen, "Virtual strings based on a 1-D FDTD waveguide model," *Proc. AES 22nd Int. Conf.*, pp. 317–323, Espoo, Finland, 2002.

[4] C. Erkut and M. Karjalainen, "Finite Difference Method vs. Digital Waveguide Method in String Instrument Modeling and Synthesis," *Proc. ISMA'2002*, Mexico City, Dec. 2002.

[5] A. Fettweis, "Wave Digital Filters: Theory and Practice," *Proc. IEEE*, 74(2), pp. 270–372, 1986.

[6] N. H. Fletcher and T. D. Rossing, *The Physics of Musical Instruments*, Springer-Verlag, New York, 1991.

[7] L. Hiller and P. Ruiz, "Synthesizing Musical Sounds by Solving the Wave Equation for Vibrating Objects: Part I, Part II," *J. Audio Eng. Soc.* vol. 19, nr. 6 and nr. 7, pp. 452–470 and 542–551, 1971.

[8] M. Karjalainen, "1-D digital waveguide modeling for improved sound synthesis," *Proc. IEEE ICASSP'2001*, pp. 1869–1872, Orlando, 2002.

[9] A. Sarti and G. De Poli, "Toward Nonlinear Wave Digital Filters," *Proc. IEEE Trans. Signal Processing*, vol. 47, no. 6, pp. 1654–1668, June 1999.

[10] L. Savioja, *Modeling Techniques for Virtual Acoustics*. PhD thesis, Helsinki Univ. of Tech., Espoo, Finland, 1999.

[11] J. O. Smith, "Principles of Waveguide Models of Musical Instruments," in *Applications of Digital Signal Processing to Audio and Acoustics*, ed. M. Kahrs and K. Brandenburg, Kluwer Academic Publishers, Boston 1998.

[12] J. Strikverda, *Finite Difference Schemes and Partial Differential Equations*, Wadsworth and Brooks, Grove, Ca, 1989.