Rudolf Rabenstein, Stefan Petrausch, Augusto Sarti, Giovanni De Sanctis, Cumhur Erkut, and Matti Karjalainen

Block-Based Physical Modeling for Digital Sound Synthesis

Building multicomponent structures

Block-based physical modeling is a methodology for modeling physical systems with different subsystems. It is an important concept for the physical modeling of real or virtual musical instruments where different components may be modeled according to different paradigms. Connecting systems of diverse nature in the discrete-time domain requires a common interconnection strategy. This contribution presents suitable interconnection strategies that incorporate a wide range of modeling blocks and considers the automatic implementation of block-based structures. Software environments are presented, which allow to build complex sound synthesis systems without burdening the user with problems of block compatibility.

Physical modeling for digital sound synthesis evolved around 1990 [1]–[4] and has since then produced a variety of modeling paradigms for all kinds of sound production mechanisms [5]–[9]. Despite or because of this progress, modeling of real or virtual musical instruments seems to be a more complex task than ever. Since musical instruments consist of different components with different physical nature, there is no best method for physical modeling. However, choosing a different model for each component leads to a mixture of different numerical methods that have to be carefully interfaced for a meaningful and reliable real-time operation and control.

A strategy well known from the simulation of dynamical systems is to separate the tasks of *component modeling* and *model interaction*. Modeling a component of a physical system means to identify the relevant quantities (force, deflection, pressure, velocity, etc.), to set up a mathematical description based on the basic laws of physics and to find a suitable discretization for

1053-5888/07/$25.00©2007IEEE

the computational realization. At the end of this process stands a discrete-time and possibly a discrete-space algorithm for the real-time simulation of the corresponding system component. Encapsulating the algorithm into an appropriate input-output description leads to a component model that processes output-samples from input-samples, preferably in real time and with no latency. Such a component model is also called a *block* because it acts as a building block for larger systems.

Creating the correct interaction between component models means to provide a meaningful connection between the input and output ports of each block in the system. This task is not as easy as it might seem. The two major problems are port incompatibility and delay-free loops. Both types of problems are discussed in detail in "Signals, Ports, and Waves."

Block-based modeling is an established method in the simulation of one-dimensional systems. A variety of simulation languages and graphical simulation tools exists. However, none of these considers all the special requirements imposed by musical applications such as extension to distributed parameter systems, interactive human control, real-time operation, and low latency.

This article reconsiders block-based modeling [10] with special emphasis on digital sound synthesis. Physically correct descriptions of the various components of musical instruments are assumed to be known [11]. Instead, the discretization of given physical models and the interconnection of the resulting discrete-time blocks are covered. These ideas have been developed in a joint project [12] with the aim of building a library of component blocks such that the different components of musical instruments can be modeled with a canonic set of blocks.

BLOCK-BASED PHYSICAL MODELING WITH WAVE-BASED INTERCONNECTIONS

SIGNALS, PORTS, AND WAVES

When dealing with physical models and their realization with discrete-time systems, it is important to distinguish the domain of signals and systems from the domain of physics. Some views and concepts from these domains are discussed in this section. In particular, signals, ports, and wave variables are reviewed.

SIGNALS

Signals are variable quantities that carry information. They need not have any immediate physical reality. However, for transmission, storage, and processing, signals are usually represented by physical quantities like electrical voltage, magnetical flux, light intensity, etc. Systems for signal processing are often described by signal flow graphs, as shown in Figure 1. The assumption in such a signal flow graph representation is that the output signal $x_1(t)$ of Block 1 does not change its values when it is connected to Block 2 as the input signal. The same holds for $x_2(t)$.

Signal flow diagrams are the basis for block-oriented simulation programs such as Simulink [13] and many others. They are also used for the graphical interface of signal-based computer music programs like pure data (pd) [14].

Block 1-3 in Figure 1 may also be realized by physical systems, e.g., electrical circuits. However, care has to be taken not to violate the above assumption. Using block realizations with high-input impedance and low-output impedance ensures that their output voltages are valid representations of the corresponding signals.

PORTS

Interconnections of physical models are described by *ports*. A port is a pair of potential and flow variables (port variables) that can be connected to a corresponding port of another model. Typical port variables are voltage and current, pressure and flow, or force and velocity. An interconnection of two ports is shown in Figure 2 where the potential variable of each port is denoted by u_1 and u_2, while the flow variables are i_1 and i_2.

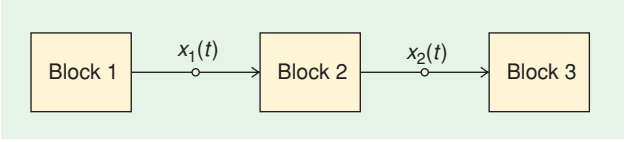
The interconnection of two ports does affect the port variables of both ports. Their interdependence is given by the Kirchhoff laws [15]. Therefore port variables are also called *Kirchhoff variables* or shortly *K-variables*. In the simple case of Figure 2, the Kirchhoff laws state that the potentials have to be equal and the flows add up to zero

$$u_1 = u_2, \qquad i_1 + i_2 = 0\,. \tag{1}$$

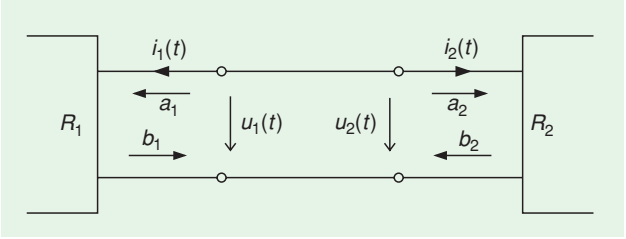
PORT INCOMPATIBILITY AND DELAY FREE LOOPS

In Figure 2 and in (1), it has been tacitly assumed that the physical nature of the port variables of both ports match, e.g., that both potentials are pressure variables. Otherwise, these ports cannot be connected. This case is referred to as *port incompatibility*.

Another problem with interconnected ports arises when the continuous-time variables $u_1(t)$, $u_1(t)$, $i_1(t)$, and $i_1(t)$ are sampled. Denote the sampled signals for $t = kT$ by $u_1[k] = u_1(kT)$ where k is the discrete time index and T the sampling interval. Further assume that the left block is approximated by a discrete-time model f, which computes the potential $u_1[k]$ from the flow sequence $i_1[k]$ by $u_1[k] = f\{i_1[k]\}$. An



[FIG1] Signal flow graph of a simple system.



[FIG2] The interconnection of two one-ports described by the physical port variables u_n and i_n and alternatively by the incident waves a_n and the reflected wave b_n, $n = 1, 2$.

example for a discrete model of a storage element is given below. In a similar way, the right block is approximated by $i_2[k] = g\{u_2[k]\}$. If both blocks contain a direct path between the input and output, then the loop of commands

$$\begin{aligned} u_1[k] &= f\{i_1[k]\}, \quad u_2[k] = u_1[k], \\ i_2[k] &= g\{u_2[k]\}, \quad i_1[k] = -i_2[k] \end{aligned} \tag{2}$$

is not computable since the value $i_1[k]$ computed by the last instruction is already required by the first one. Such a chain of commands is called a *delay-free loop* and is not realizable on any digital signal processing system.

Thus, although the discrete-time models f and g may be correctly implemented, the blocks cannot be directly connected. A naive way to break the delay-free loop is to insert a delay in the interconnection, e.g., by computing $u_2[k] = u_1[k-1]$, but this crude measure changes the dynamics of the system in an unpredictable way.

WAVE VARIABLES

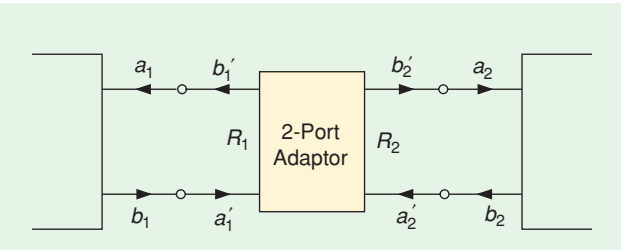
An approach to avoid delay-free loops is the transition from the physical port variables to wave variables. It demonstrated for an energy storage element, e.g., a cavity that can store air under a certain pressure or a capacitor which can store charge under a certain voltage. The Kirchhoff variable for pressure or voltage is denoted by $u(t)$ and for mass flow or electrical current by $i(t)$. They are related by integration with respect to time t,

$$u(t) = \frac{1}{C}\int_{-\infty}^{t} i(\tau)\, d\tau\,, \tag{3}$$

where the constant C denotes the storage capacity. Numerical integration by the trapezoidal rule performs the time discretization according to the bilinear transformation

$$u(kT) = u((k-1)T) + \frac{T}{2C}\,[i(kT) + i((k-1)T)]\,. \tag{4}$$

However, when this storage element is connected to another element, both u and i are affected (see Figure 2 and (1)). It is not possible to compute $u(kT)$ and $i(kT)$ simultaneously from the known values of the previous time step $(k-1)T$. To avoid the resulting implicit equation (i.e., a delay-free loop) the unknown quantities are sorted according to time (now with $u[k] = u(kT)$ etc.)



[FIG3] Interconnection of two wave ports with a parallel adaptor.

$$u[k] - \frac{T}{2C}\,i[k] = u[k-1] + \frac{T}{2C}\,i[k-1]\,. \tag{5}$$

Then, the so-called discrete *wave variables* are introduced

$$a[k] = u[k] + R\,i[k]\,, \qquad b[k] = u[k] - R\,i[k]\,, \tag{6}$$

where $a[k]$ is the incident, $b[k]$ is the reflected wave and R is the reference or port resistance [15], [16]. Expressing (5) in terms of the wave quantities with the reference resistance $R = T/2C$ gives

$$b[k] = a[k-1]\,. \tag{7}$$

Thus, the numerical integration (5) is computable by expressing the reflected wave $b[k]$ at the time instant k by the incoming wave $a[k-1]$ at the previous time $k-1$.

The use of wave variables is the key element of the wave digital principle, which has been introduced as a method for designing digital filters (wave digital filters, WDF) from analog counterparts. A unifying treatment of theory and application of wave digital filters is given in a classical paper by A. Fettweis [16]. Modern descriptions of the wave digital principle as a tool for numerical integration and modeling are given in [9], [15], [17]. Wave variables are also called *W-variables*, and *WD principle* is used as shorthand for wave digital principle.

INTERCONNECTION OF PORTS WITH WAVE VARIABLES

Since the interconnection of two ports in Figure 2 forces the port variables to obey the Kirchhoff laws, the wave variables also have to satisfy certain relations. The definitions of the wave variables for Figure 2 and their sum and difference are compiled below (the time index k is omitted)

$$\begin{aligned} a_n &= u_n + R_n i_n, \quad 2u_n = a_n + b_n, \qquad n = 1, 2\,, \\ b_n &= u_n - R_n i_n, \quad 2i_n = G_n(a_n - b_n), \quad G_n = R_n^{-1}. \end{aligned} \tag{8}$$

Expressing the Kirchhoff laws (1) for u_n and i_n in terms of the wave variables a_n and b_n results in the matrix equation

$$\begin{bmatrix} 1 & -1 \\ G_1 & G_2 \end{bmatrix}\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} - \begin{bmatrix} -1 & 1 \\ G_1 & G_2 \end{bmatrix}\begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{9}$$

It can be solved for the incident waves a_n in terms of the reflected waves b_n as

$$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \left(\begin{bmatrix} \alpha_1 & \alpha_2 \\ \alpha_1 & \alpha_2 \end{bmatrix} - \mathbf{I}_2\right)\begin{bmatrix} b_1 \\ b_2 \end{bmatrix}, \quad \alpha_n = \frac{2G_n}{G_1 + G_2}, \tag{10}$$

where $\mathbf{I}_2$ is the 2×2 identity matrix. Such relations between incident and reflected waves are also called *scattering relations*.

When the ports in Figure 2 are not expressed in terms of the port variables but in terms of the wave variables (*wave ports*), then the validity of (10) has to be ensured by a special block. It can be regarded as an impedance matcher between the port resistances and is called *adaptor*. Figure 3 shows the two ports from Figure 2

as wave ports with the corresponding two-port adaptor. For the adaptor, the roles of the incident and reflected waves are interchanged such that its defining equations are [see (10)]

$$b_1' = (\alpha_1 - 1)a_1' + \alpha_2 a_2', \tag{11}$$
$$b_2' = \alpha_1 a_1' + (\alpha_2 - 1)a_2'. \tag{12}$$

Similar relations as for the two-port junction in Figures 2–3 hold also for junctions with multiple ports where general multiport adaptors are required. They can be assembled from three-port serial and parallel adaptors according to Figure 4. The adaptor equations are derived in a similar way as for the two-port adaptor [see (8)–(10)].

INTERCONNECTION NETWORKS

Three-port adaptors and their combinations enable the building of interconnection networks of arbitrary size. The advantage of this approach is that delay-free loops are avoided and thus the computability of the whole structure is ensured. A disadvantage is that the class of compatible blocks is restricted to those that communicate via wave variables. In other words, only blocks with wave ports can be connected to the adaptors presented above. This would exclude many physical modeling and non-physical sound-synthesis approaches.

Two different directions are presented to open the wave-based interconnection strategy to other types of sound synthesis methods:

- The connection of blocks in state space representation to the wave-based interconnection structure from above is shown in the remainder of this section. Then an automated synthesis procedure is described in "Automatic Implementations of Block-Based Structures."
- The reformulation of the wave-based interconnections in terms of signal inputs and outputs with so-called wave and K-nodes is shown in "Block-Based Physical Modeling with Wave and K-Nodes."

In both cases, a so-called *KW-converter* is introduced to connect blocks based on K-variables to blocks based on W-variables.

STABILITY CONSIDERATIONS

When different blocks are connected to form a network, the stability of the complete system has to be ensured. In physical modeling, this problem is usually solved in the context of passivity. For example, a WD network is stable if each block does not return more energy than it receives and if the Kirchhoff laws are obeyed through suitable adaptors at each interconnection. It is also reasonable to expect such a condition for networks with blocks that are designed according to other paradigms. However, a formal proof for the general case of arbitrary nonlinear blocks cannot be given. A recent approach in the context of port-Hamiltonian systems appears to be too involved for sound synthesis applications [18]. Practical approaches in sound synthesis with nonlinear blocks are monitoring the internal energy balance [8], [19], or problem specific discretization schemes [20].

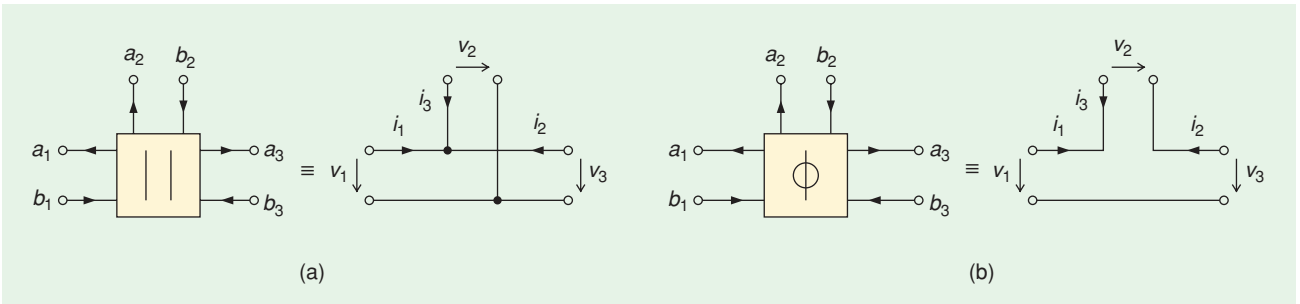
INTERCONNECTION OF STATE-SPACE MODELS TO WD MODELS

Discrete-time state-space structures with the state-space matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, and $\mathbf{D}$, (see Figure 5) are a generic representation for different kinds of modeling paradigms. When input and output variables are given as flow and potential variables, then the direct path has the physical dimension of an impedance. To connect such a physical model in K-variables to an adaptor, it is necessary to convert the K-variables to W-variables [21]. The resulting KW-converter is now described.

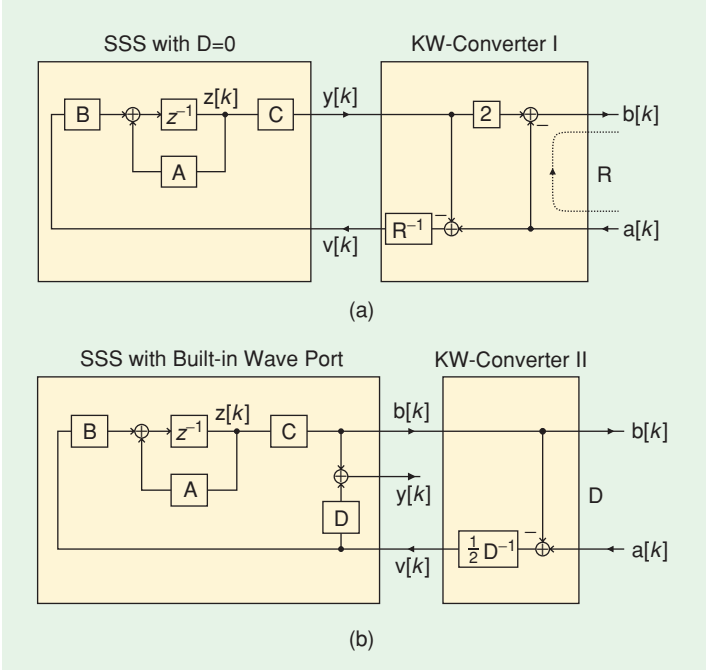
The definition of the vectors of incident and reflected waves $\mathbf{a}[k]$ and $\mathbf{b}[k]$ with the matrix of port resistances $\mathbf{R}$ and the identity matrix $\mathbf{I}$ is given in the first line of (13). This definition can be solved for the quantities that should leave the KW-converter, i.e., for the input $\mathbf{v}[k]$ of the state-space structure and for the reflected wave $\mathbf{b}[k]$ [(13) second line. The result defines a first version of the KW-converter called KW-converter I. It is only applicable if the direct path in the state space structure, i.e., the matrix $\mathbf{D}$ is zero. Otherwise, a delay-free loop would result [see Figure 5(a)].

$$\begin{bmatrix} \mathbf{a}[k] \\ \mathbf{b}[k] \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{R} \\ \mathbf{I} & -\mathbf{R} \end{bmatrix} \begin{bmatrix} \mathbf{y}[k] \\ \mathbf{v}[k] \end{bmatrix},$$
$$\begin{bmatrix} \mathbf{v}[k] \\ \mathbf{b}[k] \end{bmatrix} = \begin{bmatrix} -\mathbf{R}^{-1} & \mathbf{R}^{-1} \\ 2\mathbf{I} & -\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{y}[k] \\ \mathbf{a}[k] \end{bmatrix}. \tag{13}$$

The corresponding KW-converter for $\mathbf{D} \neq \mathbf{0}$ follows by inserting the output equation of the state-space representation into the definition of the wave variables (13). Setting the port resistance $\mathbf{R} = \mathbf{D}$ and solving for the input $\mathbf{v}[k]$ and the reflected wave $\mathbf{b}[k]$ yields the KW-converter II, which avoids delay free loops [see Figure 5(b)].



[FIG4] (a) Three-port parallel adaptor and equivalent network with Kirchhoff variables. The direction of the wave variables is given with respect to the connected block elements. (b) Three-port serial adaptor and equivalent network with Kirchhoff variables.



[FIG5] (a) State-space structure (SSS) without direct feedback (D = 0) and a KW-converter type I with an arbitrary port resistance R. (b) Connection of a general linear K-variable model in state-space structure to W-variables.

The introduction of the KW-converter type I and II is the key to the connection of models in state-space structure to wave variables. Using the wave digital interconnection strategy, an arbitrary number of such models may be connected with each other, with wave digital equivalents of network elements, and with other models using wave ports. Existing models also can be reused when an appropriate KW-converter is attached. Compared to the wave digital principle introduced before, the range of possible blocks for model building is greatly enhanced.

APPLICATION OF BLOCK-BASED PHYSICAL MODELING WITH WD INTERCONNECTION STRATEGY

The block-based physical modeling approach presented before is now discussed in more detail. It is shown how to construct complex systems with different kinds of models using the WD interconnection strategy.

Frequently used physical, K-variable models for strings, membranes, and plates, etc., are modal synthesis [1], [22] and functional transformation method (FTM) models [23]. FTM models are a direct transformation of the underlying partial differential equation (PDE) into the frequency domain and have a number of advantages for digital-sound synthesis, like real-time capability, stability, and parameter variation during runtime. It is therefore of interest to investigate how the block-based modeling methods presented before allow the use of an FTM model as a building block in a larger structure. This investigation is also valid for modal synthesis, since both methods lead to a similar computational structure shown in Figure 6. For a detailed comparison of FTM and modal synthesis, see [23].

A MEMBRANE MODEL WITH THE FUNCTIONAL TRANSFORMATION METHOD

The physical model of vibrating structures is derived from basic laws of physics [11] and relates the input force $f_e(\vec{x},t) = \gamma_0(\vec{x}-\vec{x}_e)\,f_e(t)$ at the excitation position $\vec{x}_e$ ($\gamma_0(\vec{x})$ denotes the two-dimensional spatial delta-impulse) with the deflection $y(\vec{x},t)$ of the two-dimensional ($\vec{x} = [x_1\ x_2]^T$) vibrating body by

$$\ddot{y} - c^2\nabla^2 y + S^4\nabla^4 y + d_1\dot{y} - d_3\nabla^2\dot{y} = f_e \qquad (14)$$

where $\dot{y}$ denotes temporal derivative and $\nabla^2 y$ denotes the two-dimensional Laplace-operator, i.e., a second order spatial derivative of the membrane's deflection.

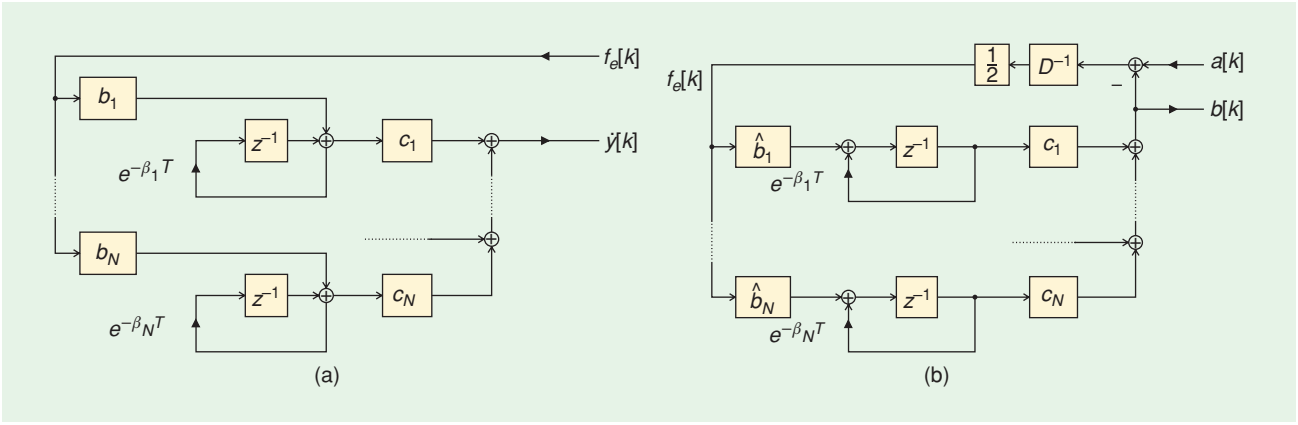
The model includes the lossless wave equation, with the speed of sound c. Two additional damping terms with the coefficients d_1 and d_3 model frequency independent and frequency dependent damping. The fourth order spatial derivative models the stiffness of the membrane. It is scaled by the dispersion constant S^4, which is derived from Young's modulus, the thickness of the membrane, the mass density of the membrane, and Poisson's ratio. The functional transformation method converts the PDE together with suitable initial and boundary conditions into a discrete-time algorithm [23]–[25]. In detail, the vibration of the membrane is modeled by the parallel arrangement of first-order systems with complex frequencies β_n, $n = 1, \ldots, N$. With the weighting constants b_n and c_n following directly from the application of the functional transformation method [25], the output is calculated by the discrete system

$$\mathbf{z}[k] = \begin{bmatrix} e^{-\beta_1 T} & 0 & \cdots & 0 \\ 0 & e^{-\beta_2 T} & \cdots & 0 \\ \vdots & & \ddots & \\ 0 & 0 & \cdots & e^{-\beta_N T} \end{bmatrix} \mathbf{z}[k-1] + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix} f_e[k]$$

$$\dot{y}[k] = [\,c_1 \quad c_2 \quad \ldots \quad c_N\,]\,\mathbf{z}[k],$$

which is also shown in Figure 6(a). Each first-order system is driven by samples of the excitation $f_e[k]$ and delivers a contribution to the velocity of the vibration $\dot{y}[k]$. However for the communication with other blocks, the Kirchhoff variables force $f_e[k]$ and velocity $\dot{y}[k]$ have to be converted to the wave variables $a[k]$ and $b[k]$.

Adding a KW-converter of type II, according to "Interconnection of State Space Models to WD Models," attaches a wave port as shown in Figure 6(b). The constant D is the sum of all N direct path gains from input to output (a), i.e., $D = \sum_{n=1}^{N} b_n \cdot c_n$. The remaining paths from the input to the delay elements are weighted with $\hat{b}_n = b_n \cdot e^{-\beta_n T}$. Thus, the



[FIG6] (a) Structure of the membrane implementation with N complex harmonics. (b) Structure of the membrane implementation that uses the wave variables $a[k]$ and $b[k]$ for input and output instead of the Kirchhoff variables $f_e[k]$ and $\dot{y}[k]$.

discrete-time model that results from the functional transformation method can be connected to wave digital equivalents [26].

MODELS OF PERCUSSION INSTRUMENTS

The membrane model introduced in the previous section is now used to model percussion instruments. The structure in Figure 6(b) is used as a block model for a membrane. It is implemented within a program called BCT, which has been developed at the Politecnico di Milano [27], [28] in the course of the European project ALMA [12].

Figure 7 uses the membrane as a basic building block. The excitation mechanism is a felt-covered hammer. The inductor and the resistor model the dynamics of the hammer motion. The nonlinear block on the left contains the force-dependent stiffness of the hammer felt. All these blocks are connected by an interconnection network consisting of a series and a parallel three-port adaptor. The motion of the hammer is triggered by external events from a musical instrument digital interface (MIDI) keyboard.

AUTOMATIC IMPLEMENTATION OF BLOCK-BASED STRUCTURES

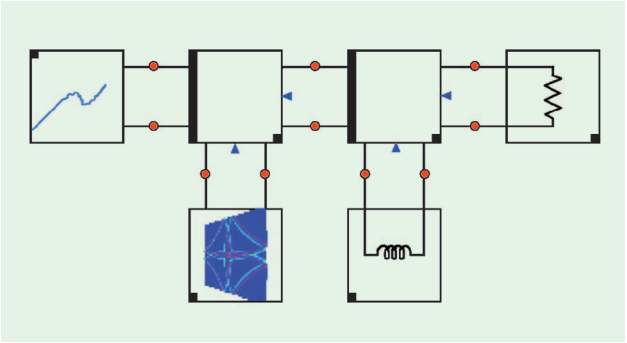
One aspect that has received less attention in the literature of sound synthesis through physical modeling is the systematic characterization of the interactions between existing models. What generates and supports vibrational phenomena in a resonating structure is a nonlinear interaction between that structure and another one. This problem was already approached and solved for developing the Cordis-Anima system [29], which automated interactions between elements of a limited class of elementary building blocks. This section considers the more general problem of modeling the interaction between a wider class of models.

Although the literature is rich with ad-hoc solutions for the modeling of excitational interactions of various nature, no attempts appear to have been made to develop systematic and automatic strategies for modeling and implementing such interactions between many pre-existing models in a simple and automatic fashion. Situations of this sorts are encountered, for example, in the sonification of acoustic events in virtual reality, or just in the modeling of rich percussion sets. A rather general approach to automated modeling of WDFs is given in [30], but it bears no relations to the specific problems of sound synthesis.

Here, an overview of two approaches to the automated modeling of block-wise interactions between physical models is provided. The first one is based on the iterative scanning of a certain WD interconnection structure, the binary connection tree (BCT) and is described in the remainder of this section. The second one, a multi-paradigm, block-based software environment called BlockCompiler (BC), is described in "Block-Based Physical Modeling with Wave and K-Nodes."

IMPLEMENTATION OF BLOCK-BASED PHYSICAL MODELING WITH WD INTERCONNECTION

Adopting a WD approach for the physical modeling of virtual musical instruments can be seen as a problem of interconnecting WD blocks under specific interconnection constraints. As described in "Signals, Ports, and Wavs," the interconnection between one-port WD elements through parallel and serial adaptors gives rise to computable WD filters



[FIG7] Example network to simulate the excitation of a membrane with a mallet. The membrane is modelled by a distributed parameter model according to Figure 6 (bottom left). The mass of the mallet is represented by a lumped parameter reactive element (bottom right) and the elasticity of the mallet felt by a memoryless nonlinearity (top left). The dissipation of the excitation mechanism is contained in a lumped parameter resistive element (top right). The parallel and series adaptors are the backbone of the interconnection network.

because the corresponding signal flow diagram does not contain any delay-free directed loops [16], [31]. This condition of realizability implies that any connection between two ports gives rise to no delay-free directed loop and that no other delay-free directed loop can be created via some outer path [16]. This is indeed guaranteed if the network of adaptors has a tree-like structure, which is always true with the sound-production mechanisms that are encountered in musical acoustics.

The above considerations are valid not just for interconnections of standard WDF adaptors but also for more general networks of dynamic WD adaptors [32], [33], which are (parallel or series) adaptors whose port resistances R_i are replaced by port impedances $Z_i(z)$, therefore incident port waves are subject to non-instantaneous (filtered) scattering caused by the reflection filters. A tree-like network of such dynamic adaptors is here referred to as a (dynamic) Macro-Adaptor (MA), and a port that exhibits no instantaneous reflection is called *adapted* port. As reflected waves at this port are either absent or delayed, this port can be freely connected to another port without causing computability problems. Realizability issues (a nonadapted port can only be connected to an adapted one) imply that the MA can only have up to one *adapted* port, just like the adaptors that it is made of. The WD structures of interest in musical acoustics are thus made of a number of two-ports connected together through one of such MAs. The adapted port of the MA is assumed to be connected to a NonLinear Element (NLE), which models the nonlinear interaction between two WD subsystems (e.g., a hammer and a string).

It was recently proved [34], [35] that a computable tree-like interconnection of adaptors with memory is completely equivalent to a memoryless macro-adaptor (a computable tree-like interconnection of standard WDF adaptors, i.e., instantaneous adaptors) whose outer ports are connected to mutators (two-port adaptors with memory) [32], [33]. It is thus easy to define a process of memory extraction from [35] that preserves the interconnection topology. This approach simplifies the implementation to WD structures based on one-port WD elements (generally with memory) interconnected by a tree-like arrangement of standard WDF adaptors. The nonlinear element is generally connected to the adapted port of the resulting instantaneous MA, possibly through a mutator. A method [27], [36] for automatically implementing a WD structure of this sort through a direct inspection (scanning) of the tree-like topological representation of the reference model (connection tree) is described below.

AN INTRODUCTORY EXAMPLE OF CONNECTION TREE

Consider a chain of L adaptors (see Figure 8), each connected to the adapted port of the previous one, except for adaptors at the extremes of the chain, which are connected to one-port elements. In particular, the adapted port is connected to a nonlinear element (NLE). This particular situation helps to introduce the method in the more general case of tree-like structures.

Now compute the vector of reflected waves $\mathbf{b}$ from the known elements of the vector of incident waves $\mathbf{a}$. One should keep in mind that the only elements of $\mathbf{a}$ that are known are those that correspond to the ports connected with linear two-ports

- $a_i[k] = 0$, when the two-port is an adapted resistor;
- $a_i[k] = V_i$, when the two-port is an adapted real generator;
- $a_i[k] = \pm b_i(k-1)$, if the two-port is a reactance.

The steps for computing $\mathbf{b}$ from the known elements of $\mathbf{a}$ are

1) once the model is initialized, the vector of incident waves will be

$$\mathbf{a} = [a_1, a_2, \bar{a}_3, \bar{a}_4, a_5, \bar{a}_6, \ldots, a_{3L-1}, \bar{a}_{3L}]^T$$

where $\bar{a}_i$ denotes an unknown, while a_i represents a known variable.

2) Then start from the first adaptor, which is the only one in which both the inputs of the nonadapted ports (a_1 and a_2) are known. The output of the adapted port can be readily computed as $b_3 = f(a_1, a_2) = k_{11}a_1 + k_{12}a_2$, k_{11} and k_{12} being appropriate transmission coefficients that depend on the reference resistances of the other two ports of the adaptor. Notice that b_3 does not depend on a_3 because port 3 is adapted.

3) Port 3 is connected to port 4, therefore both a_5 (from the two-port) and $a_4 = b_3$ are computed in the previous step.

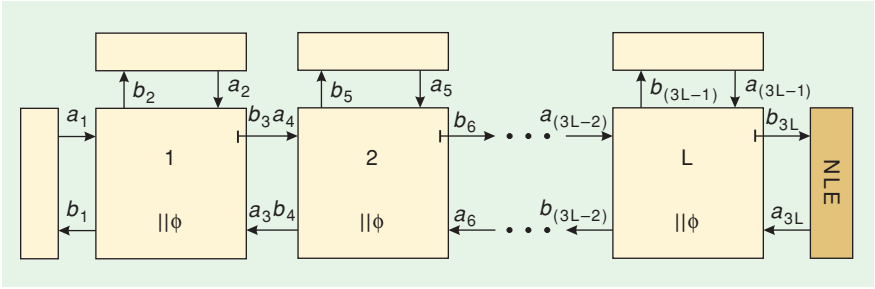
4) Now, repeat steps 1 and 2 until the last adaptor is reached. This way, some of the elements of $\mathbf{a}$ and $\mathbf{b}$ are computed:

$$\mathbf{a} = [a_1, a_2, \bar{a}_3, a_4, a_5, \bar{a}_6, \ldots, a_{3L-2}, a_{3L-1}, \bar{a}_{3L}]^T$$
$$\mathbf{b} = [\bar{b}_1, \bar{b}_2, b_3, \bar{b}_4, \bar{b}_5, b_6, \ldots, \bar{b}_{3L-2}, \bar{b}_{3L-1}, b_{3L}]^T.$$

5) Using the characteristic of the nonlinear element gives $a_{3L} = f_{\mathrm{NLE}}(b_{3L})$.

6) Now, compute, if necessary, b_{3L-1} and/or b_{3L-2} and go through the whole model computing b_{3i-1} and b_{3i-2} until the first adaptor. Notice that not all the reflected waves b_{3i-j} $(j = 1, 2)$ need to be computed. For example, the reflection at those nonadapted ports that are connected to a resistor (no reflection) can be skipped.

7) Once $\mathbf{b}$ is specified, it is possible to update $\mathbf{a}$ using the I/O descriptions of the two ports. Then go back to the beginning of the procedure to compute the next value of $\mathbf{b}$.



[FIG8] Chain structure: each adaptor is connected to at least one two-port, and there is no branching.

BUILDING THE CONNECTION TREE

As already said earlier, the WD structures in which we are interested are obtained by interconnecting WD blocks through a tree-like network of standard WDF adaptors. As any parallel (series) N-port adaptor can always be implemented as a chain of $N-2$ parallel (series) three-port adaptors connected together, we can generally assume that our tree-like interconnection is only made of three-port parallel and/or series adaptors.

The chain-like model described above can be generalized by allowing the other two nonadapted ports to be connected to adapted ports of other three-port junctions. The topology remains tree-like, and it is built only with three-port adaptors.

Given a binary connection tree (BCT) that describes the interconnection topology of a WD structure, an automatic implementation strategy for the WD model is defined by assigning a hierarchical ordering to the elements of the tree according to the following rules:

- The **root** of the binary tree corresponds to the adaptor to which the nonlinear (NL) element connects.
- The **nodes** of the tree are three-port standard WDF adaptors, and the branching topology matches the actual adaptor's interconnection topology.
- The **leaves** correspond to the linear two-ports.

The method for the chain-like model can be readily extended to this new situation by defining a forward scan (from the leaves towards the NLE) and a backward scan (from the NLE towards the leaves). In fact, the computation starts from the leaves of the tree, which contain the "memory elements" with the initial conditions. The computation then proceeds by following a forward scan and, when the working point of the NLE is found, the backward scanning propagates the computation back toward the leaves, where all memory cells are refreshed.

Figure 9 shows that the adapted port of a junction is either connected to a nonadapted port of another adaptor or to a nonlinear element. If there are no nonlinearities, then there is a spare adapted port, or one degree of freedom in the choice of the reflection filters. Algorithms described by a BCT can be implemented efficiently since their computational cost and their memory requirements increase linearly with the number of adaptors.

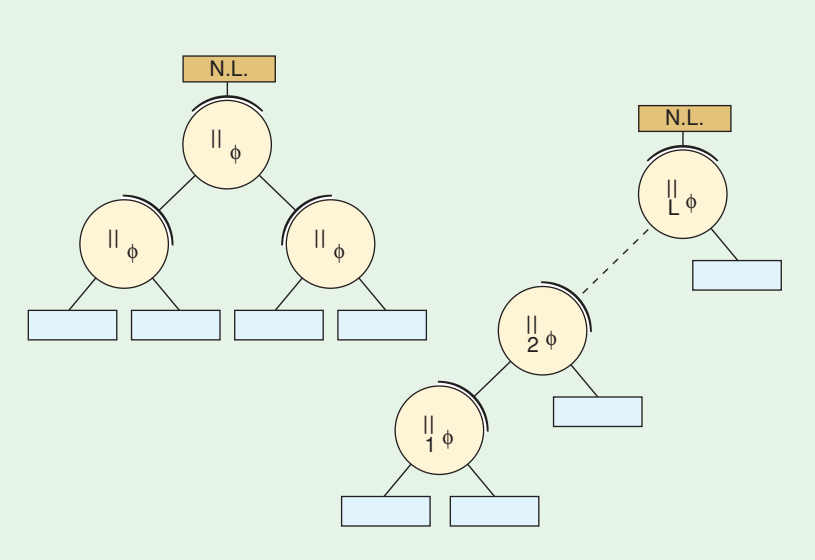
INITIALIZATION ISSUES

The leaves of the connection tree usually represent dynamic elements such as capacitors (springs) or inductors (masses). This means that the corresponding memory cells need to be assigned values that correspond to the initial conditions of the model. They are determined from the voltage/current pair that corresponds to the element's port. Notice that the initial condition on that element specifies only one variable of the pair, therefore the other one needs to be determined. One way to do so is to solve the model in which all the dynamic elements have been replaced by ideal generators to set the corresponding K variable.

As reactances are now formally replaced by ideal generators, it is not possible to use wave variables directly, otherwise the structure would turn out to be noncomputable. However, the tree structure that describes the model topology can still be used, regardless of whether we are working in the wave domain or in the K domain. In particular, it is quite simple to compute the potential/flow characteristic of the subtree at a generic node, given those of the two "children nodes." In fact, as the model portion described by the subtree is linear, such characteristics are *lines* that can be modeled according to the Thevenin or Norton equivalent model corresponding to the subtree. Such equivalent is specified by a pair of parameters: the magnitude associated to the ideal generator (intersection with one of the axes), and the resistance/conductance of its internal resistor (slope of the line). Knowing the lines that model the two subtrees and the type of node that combines them (series or parallel), the line that describes the combination of such two-ports is computed. More details about initialization issues can be found in [36].

MANAGING TIME-VARYING STRUCTURES

Physical models in musical acoustics are quite obviously expected to be time-varying. Temporal changes in such models usually concern the nonlinearity that models the interaction or the current/voltage produced by properly defined generators. Although infrequently, there may be changes in port impedances with the result of having time-varying reflection filters. In this last case, a parametric change may affect many other parameters as they are all bound to satisfy global adaptation conditions. Temporal changes of port resistances



[FIG9] Two examples of BCTs: a generic one (*a*) and that of a chain-like model (*b*). The circular box represents an instantaneous adaptor, in which the adapted port is clearly specified. This particular notational choice simplifies the drawing of connection trees with a great deal of branching.

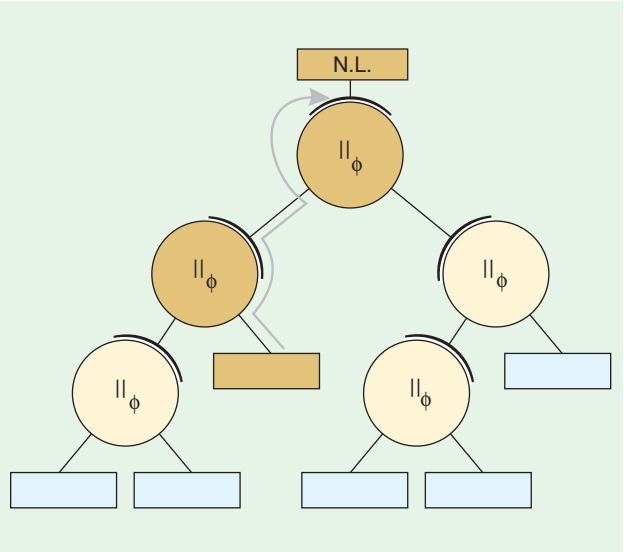
are, in fact, implemented through a recomputation of the model parameters on the behalf of a process that works in parallel with the simulator. Using the BCT method, when the value of a leaf changes, the adaptors that need to be updated can be limited only to those that lie on the path that links the leaf to the root (Figure 10).

BLOCK-BASED PHYSICAL MODELING WITH WAVE AND K-NODES

WD components, state-space models with wave port interfaces, and three-port interconnection adaptors discussed so far satisfy the requirements of block-based physical modeling with wave variables. This strategy can be extended to accommodate physically-inspired models, such as *commuted synthesis*, as well as terminal-based nonphysical blocks such as DSP filters and effects [37]. For instance, a *quasi-physical* wavetable can be defined as a voltage source that reads and outputs the content of a wavetable, and its port resistance can be used to control the gain. This new element, in turn, can be imported to BCT (see "Automatic Implementation of Block-based Structures") as a plug-in.

However, it is also desirable to keep the terminal-based structure of the nonphysical blocks and interconnect them freely with the port-based physical blocks. This might be the case when a library of such blocks is readily available. The key idea here is to reformulate the interconnection elements to accept and provide *signal* inputs and outputs, respectively. In the sequel, we call these elements *nodes*, provide definitions for wave and K-nodes, and discuss their interconnection.

While the wave and K-nodes could be formulated for multiparadigm, block-based physical modeling without any reference to a specific software system, we consider their implementation in block compiler (BC) in this section [38].



[FIG10] Typial BCT configuration with tree updating after a two-port value change. Root and leaves are represented by rectangles. The nodes consist of serial and parallel adaptors according to Figure 4 and are represented by circles.

BEYOND ADAPTORS: WAVE NODES, K-NODES, AND THEIR INTERCONNECTION

WAVE NODES

The wave-node formulation follows directly from the Kirchhoff laws; it is derived here by using the *scattering junctions*, which implement the Kirchhoff laws in the theory of *digital waveguides* (DWGs) [4], [37]. A basic DWG element is a bidirectional delay line pair of a specific port admittance Y. Usually, this admittance is positive real, and scattering occurs in case of admittance change across a junction. For instance, in a parallel junction of N waveguides in the electrical domain, the Kirchhoff constraints in (1) become

$$U_1 = U_2 = \cdots = U_N = U_J, \qquad I_1 + I_2 + \cdots + I_N + I_{\text{ext}} = 0 \tag{15}$$

where the complex z-transform variable has been dropped in notation for easier reading, U_i and I_i are the voltage and current of the ith branch, respectively, U_J is the common voltage of coupled branches, and I_{ext} is an external current source. In DWG theory, it is customary to decompose the branch voltages into incident wave components towards the junction U_i^+ and reflected wave components away from the junction U_i^-, and represent port admittances by Y_i, so that

$$U_i = U_i^+ + U_i^- \text{ and } I_i^+ = Y_i U_i^+. \tag{16}$$

The junction voltage U_J can then be obtained as

$$U_J = \frac{1}{Y_{\text{tot}}}\left(I_{\text{ext}} + 2\sum_{i=1}^{N} Y_i U_i^+\right), \tag{17}$$

where $Y_{\text{tot}} = \sum_{i=1}^{N} Y_i$ is the sum of all admittances to the junction. Reflected voltage waves are obtained from (16) to yield

$$U_i^- = U_J - U_i^+. \tag{18}$$

Equations (17)–(18) may be used to define an N-port scattering junction, which is denoted as a *wave node*. Note that (17) contains the physical variables I_{ext} and U_J in signal form as an input and output, respectively.

K-NODES

Functional equivalents of wave nodes have been also formulated for particular *finite difference time domain* (FDTD) structures operating on the K-variables [39]. Consider a wave node in a source-free structure for $N = 4$ branches of equal admittance. In this case, (17) becomes

$$U_J = \frac{1}{2}\sum_{i=1}^{4} U_i^+, \tag{19}$$

which is the junction voltage equation for a 2-D rectilinear DWG mesh [37]. By summing the wave components up according to (16) and accounting for the propagation delays, this equation can be rearranged [37] to yield

$$(1+z^{-2})U_J = \frac{1}{2}\sum_{i=1}^{4} z^{-1}U_{J,i}, \tag{20}$$

which is a finite difference equation of the ideal 2-D wave equation on a square grid with the propagation velocity

$$c = \frac{X}{T\sqrt{N/2}} = \frac{X}{T\sqrt{2}}$$

where T and X are temporal and spatial sampling intervals, respectively. This relation can be generalized for a junction of N branches of admittances Y_i with a source term [39]. The resulting N-port element has been called a *K-node*. A K-node is characterized by the following equation:

$$(1+z^{-2})U_J = \frac{1}{Y_{\text{tot}}}\left((1-z^{-2})I_{\text{ext}} + 2\sum_{i=1}^{N} z^{-1}Y_i U_{J,i}\right). \tag{21}$$

Note that, unlike a memoryless wave node, a K-node has a memory of two delays. This property has an advantage and a drawback. The advantage is that, regardless of N, the memory need remains constant at two unit delays, since the signal transmission between the K-nodes is carried out by delayless *K-pipes*. This should be compared to the requirement of N delays between each pair of wave nodes. However, the memory should be updated with care during parameter changes, as this may cause instabilities.

INTERCONNECTION OF WAVE NODES AND K-NODES

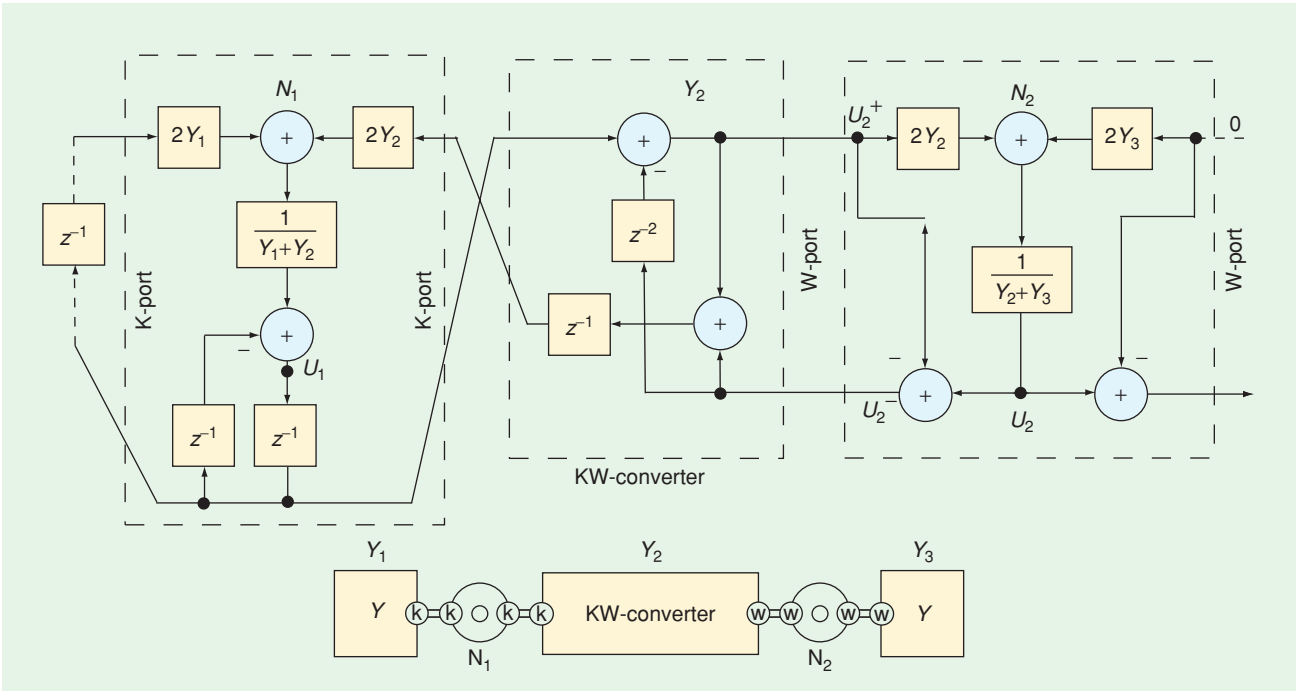
In the signal flow diagram of Figure 11, a K-node N_1 (a) and a wave node N_2 (b) are aligned in a source-free structure with the spatial indices $i = 1$ and 2, respectively. Note that the junction voltages are available in both types of nodes but not at the wave ports of the wave node. Nevertheless, the similarity of the nodes is used in [39] to obtain the following transfer matrix of the two-port KW-converter element of type III, which interconnects the K-node and the wave node in Figure 11

$$\begin{bmatrix} U_2^+ \\ z^{-1}U_2 \end{bmatrix} = \begin{bmatrix} 1 & -z^{-2} \\ 1 & 1-z^{-2} \end{bmatrix}\begin{bmatrix} z^{-1}U_1 \\ U_2^- \end{bmatrix}. \tag{22}$$

IMPLEMENTATION IN BLOCKCOMPILER

BC is a block-based software environment for research, an authoring tool for model building, and a runtime synthesis engine with a high-level specification of computational models [38], [40]–[42]. While BCT (see "Automatic Implementation of Block-Based Structures") can manage dynamic topologies, BC is designed to efficiently support topologies fixed at compile time. Efficient simulation in the BlockCompiler is based on optimized code that is produced from block-based description and compiled to run-time code. The high-level object-oriented part is written and scriptable in common lisp, which allows for flexibility in manipulating computational structures. This level generates C code from block-based specifications, which can be also exported to other software environments, and a C compiler compiles this code into one that is executable.

Multiparadigm modeling is another key design principle in BC. The first category of objects supports conventional DSP and one-directional signal data flow between object terminals. This includes elementary blocks such as adders, multipliers,



[FIG11] Top: A K-node (left) and a wave node (right) forming a part of a hybrid waveguide in a source-free structure. There is a KW-converter (type III) between K- and wave nodes, and Y_2 is its wave admittance. U_1 and U_2 are the junction voltages of the K-node and wave node, respectively. Y_1 and Y_3 provide terminations for the corresponding node, following the principles developed in [39]. Bottom: Abstraction of the structure.

```
(patch ((kw (.kw-converter :admittance 2.0) ;;     KW line Y2
        (y1 (.y :admittance 10.0 :type 'K)) ;; K-termination Y1
        (y3 (.y :admittance 10.0 :type 'W))) ;; W-termination Y3
    (-> (.ad) (in (.par y1 (port kw 0)))) ;; connect input
    (-> (out (.par y3 (port kw 1))) (.da))) ;; connect output
```

nonlinear functions, filters, transformations, and sound I/O. A more advanced category of objects supports modeling of physical interactions. The elements are connected to the wave or K-nodes through two-way ports that carry physical signal variables.

In both categories, each block can be given a relative sampling rate, and thus, multirate processing is supported. In addition, macro blocks can be defined as containers of more elementary blocks.

In the rest of this section, we focus on examples in block-based physical modeling. For a more detailed description of the BC including its data structures, code generation, and DSP-based or more advanced physical applications, see [38] and [40]–[42].

PHYSICAL BLOCKS AND INTERCONNECTION IN BC

In BC, a model specification script is called a *patch*, which consists of labeled description of the blocks and their interconnection. Currently, BC does not have a graphical user interface for model creation; the patches are written in Lisp-syntax.

Consider the abstracted structure in Figure 11 with the addition of a current source input into N_1 and voltage output from N_2. The lower-case letters k and w at the node ports distinguish the variables of the node and attached blocks. The blocks contained in the diagram are the termination admittances Y_1 and Y_3, and the kw-converter of type III [see (22)] of admittance Y_2. The other blocks supported by BC but not included in this example are `impedance`, `k-pipe`, and `w-line`. A `w-line` is a digital waveguide, and a `k-pipe` is a delayless connection element between the K-nodes.

Next we focus on the interconnection of these blocks. BC supports parallel and series connection functions, which are denoted by `.par` and `.ser`, respectively. These functions are called with block labels as arguments. When passing a two-port block of type `k-pipe`, `w-line`, or `kw-converter` as an argument, the `port` number [0,1] should be also given. An external source, which can be obtained for example from an A-D converter (instantiated by the function `.ad`) can be directed as an input `in` to a connection function, and the junction output `out` can be streamed to a D-A converter `.da`. The patch shown at the top of the page specifies the model.

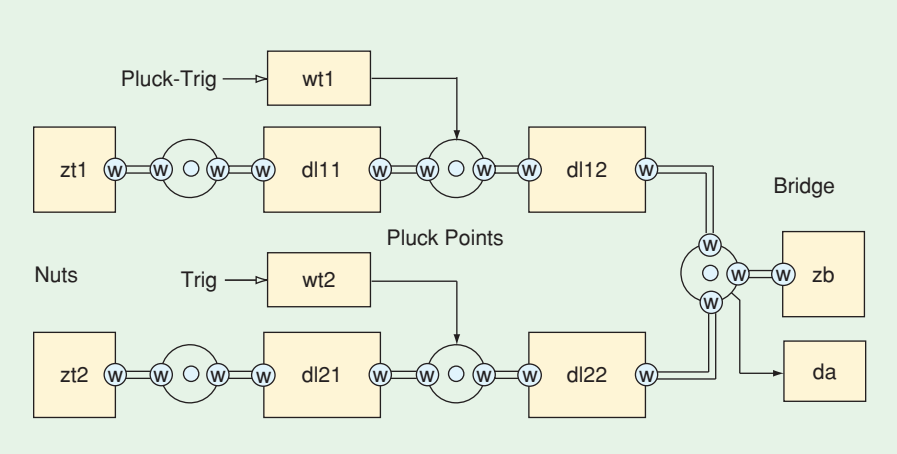
In the block creation phase, the admittances are initialized with positive real values $Y_1 = Y_3 = 10$ and $Y_2 = 2$. Note that interconnections are performed within the streaming using the short-hand signal chaining function `->`.

A PLUCKED STRING MODEL

The next example demonstrates how to build a string synthesis model for two plucked-strings [43], [44], using blocks with physical, two-directional interaction. Figure 12 depicts a mechanical-domain block diagram for two strings with pluck wavetable inputs and strings being interconnected through a common bridge impedance, as well as bridge velocity output and nut end string terminations. We do not present the patch code here, but it follows the principles discussed in the previous subsection.

The strings are divided into two subsections so that the pluck excitation force can be injected from a triggerable wavetable (`wt1` and `wt2`) through force inputs of the w-nodes into a controllable plucking point. The string blocks (`dl11`, `dl12`, and `dl21`, `dl22`) are of type `w-line` with controllable fractional delays, and string losses are lumped to nut end termination impedances `zt1` and `zt2`, implementing frequency-dependent losses by IIR type of impedance specifications.

The common bridge impedance `zb` results in a sympathetic coupling between the strings and it can be given any specification of FIR or IIR type, including a measured or modeled impedance of relatively high order. The output signal is probed from the bridge velocity. It could be processed further through a body filter to simulate the sound radiation of an acoustic guitar through a body but, for simplicity, it is fed directly to sound output. An acoustical guitar would require six strings (each one



[FIG12] Block diagram of two strings coupled through a common bridge impedance (Z_B).

with dual-polarizations, i.e., two submodels). Furthermore, finger-string interaction could be simulated by a fingertip model with varying parameters, based on the contact state of the finger and the string.

A SHORT NOTE ON THE SOFTWARE PLATFORMS

Two software environments for automated modeling of block-wise interactions between physical models have been presented. Some differences concerning their design principles were indicated tacitly; here, it is recapitulated that one of the most important differences between BCT and BC relates to how each platform manages the interaction topologies. BC is designed to efficiently support topologies fixed at compile time and the compiled models typically start from an initially relaxed state. On the other hand, as described in "Automatic Implementation of Block-Based Structures," BCT properly handles the initialization by taking the advantage of a high-level abstraction based on tree-structures and network analogies.

BCT, however, cannot cover all cases of interest for the generic user. For example, at its current state of development, it cannot deal with multiple nonlinearities (however, see [45]). In addition, many users feel that it is important to be able to accommodate other types of modeling paradigms. While the KW-converters extend the BCT in the latter sense, these needs are more directly addressed by BC, with its multiparadigm support and different categories of objects (See "Implementation in BlockCompiler").

In conclusion, BCT and BC are very different in nature and, in particular, in their level of abstraction, as they are associated to differently constrained modeling environments. In other words, while BCT provides the user with a "vertical" (specialized) approach to sound modeling, the BC can be thought of as a "horizontal" (general purpose) approach to sound synthesis and modeling.

CONCLUSIONS

Research on physical modeling digital sound synthesis has created a multitude of different models for all kinds of components of musical instruments. These are based on several modeling paradigms that are formulated either in Kirchhoff variables (e.g., the finite difference time domain method or the functional transformation method) or in wave variables (digital waveguides, wave digital structures). This variety makes it difficult to build a multicomponent structure from different kinds of block models. A solution to this problem can be found in the selfcontained world of wave digital filters. They provide a reliable strategy for the discretization and interconnection of network elements. This strategy can be extended to include other discrete-time blocks in the form of state-space structures or other interconnection approaches based on digital waveguides. Solutions to this problem are presented in this article. Automated design procedures and programs for both kinds of extensions exist in the form of the BCT and the BC, respectively. They allow the building of complex synthesis structures from a library of different kinds of component models without bothering the user with problems of block compatibility. The results are automatically generated synthesis algorithms for real-time operation, interactive human control, and parameter variations with low latency.

AUTHORS

Rudolf Rabenstein (rabe@LNT.de) received the degrees "Diplom-Ingenieur" and "Doktor-Ingenieur" in electrical engineering and the "Habilitation" degree in signal processing, all from the University of Erlangen-Nuremberg, Germany in 1981, 1991, and 1996, respectively. He worked with the physics department of the University of Siegen, Germany, and with the Telecommunications Laboratory at the University of Erlangen-Nuremberg where he is currently a member of the faculty. His research interests are in the fields of multidimensional systems theory and multimedia signal processing. He is the author and coauthor of more than 100 scientific publications, has contributed to various books, and holds several patents in audio engineering.

Stefan Petrausch (stepe@LNT.de) received the "Diplom-Ingenieur" degree in electrical engineering and signal processing from the University of Erlangen-Nuremberg in 2002, where he also received the diploma-award of the Department of Electrical, Electronic, and Communication Engineering in 2003. He later joined the Telecommunications Laboratory at the University of Erlangen-Nuremberg where he worked with the EU shared-cost project ALMA from 2002–2004. He is the author and coauthor of more than 25 scientific publications and is currently finishing his "Doktor-Ingenieur" degree in the same institution. His primary research interest is physical modeling with particular applications to sound synthesis and acoustics.

Augusto Sarti (sarti@elet.polimi.it) received the "Laurea" degree (1988, cum laude) and the Ph.D. (1993) in electrical engineering from a joint program between the University of Padua, Italy, and the University of California at Berkeley. In 1993, he joined the Dipartimento di Elettronica e Informazione of the Politecnico di Milano, where he is currently an associate professor. His current research interests are in the area of digital signal processing, with particular focus on computer vision, image and sound processing. He is the author of over 150 scientific publications on international journals and congresses and numerous patents on image and sound processing.

Giovanni De Sanctis (desancti@elet.polimi.it) received the "Laurea" degree in electronic engineering from the Politecnico di Milano in 2003. He then worked for three years as a contract researcher with the Image and Sound Processing Group (ISPG) at the Politecnico di Milano, working on sound synthesis through physical modeling and acoustic source localization for human-computer interaction. He is currently doing a Ph.D. on parameters estimation for physical modeling at the Sonic Arts Research Centre, Queen's University, Belfast, Northern Ireland. His research interests are on signal processing, with an emphasis on sound source modeling and tactile interfaces.

Cumhur Erkut (Cumhur.Erkut@tkk.fi) received his B.Sc. (1994) and M.Sc. (1997) in electronics from the Yildiz Technical

University, Istanbul, Turkey, and his Dr.Sc.(Tech.) degree (EE) from the Helsinki University of Technology (TKK), Espoo, Finland, in 2002. He is currently a postdoctoral researcher in the same institution where he contributes to the research projects "Sound to Sense, Sense to Sound" (S2S^2) and "Modeling and Perception of Sound Sources" (MAPS). His primary research interests are physics-based sound synthesis and musical acoustics.

Matti Karjalainen (Matti.Karjalainen@tkk.fi) received M.Sc. and Dr.Sc. (Tech.) degrees in electrical engineering from the Tampere University of Technology, in 1970 and 1978, respectively. In 1980, he became a professor of acoustics and audio signal processing at the Helsinki University of Technology in the Department of Electrical and Communications Engineering. His main interest is in audio signal processing, such as DSP for sound reproduction and auralization, music DSP, and sound synthesis, as well as perceptually based signal processing. He has written 350 scientific and engineering papers and contributed to organizing several conferences and workshops. He is an AES fellow and silver medalist as well as member of several scientific and engineering societies.

REFERENCES

[1] J. Adrien, "Dynamic modeling of vibrating structures for sound synthesis, modal synthesis," in *Proc. AES 7th Int. Conf. Toronto, Canada: Audio Engineering Society*, 1989, pp. 291–299.

[2] A. Chaigne, "On the use of finite differences for musical synthesis: Application to plucked stringed instruments," *J. d'Acoustique*, vol. 5, no. 2, pp. 181–211, 1992.

[3] G. Borin, G. De Poli, and A. Sarti, "Algorithms and structures for synthesis using physical models," *Comput. Music J.*, vol. 16, no. 4, pp. 30–42, 1992.

[4] J.O. Smith III, "Physical modeling using digital waveguides," *Comput. Music J.*, vol. 16, no. 4, pp. 74–91, 1992.

[5] J.O. Smith III, "Physical modeling synthesis update," *Comput. Music J.*, vol. 20, no. 2, pp. 44–56, 1996.

[6] S.A. van Duyne, J.R. Pierce, and J.O. Smith, "Traveling wave implementation of a lossless mode-coupling filter and the wave digital hammer," in *Int. Computer Music Conf. (ICMC)*, Aarhus, Denmark, 1994, pp. 411–418.

[7] G. De Poli and D. Rocchesso, "Physically based sound modeling," *Organised Sound*, vol. 3, no. 1, pp. 61–76, 1998.

[8] S.D. Bilbao, J. Bensa, and R. Kronland-Martinet, "The wave digital reed: A passive formulation," in *Digital Audio Effects (DAFx-03)*, London, Sept. 2003, pp. 225–230.

[9] V. Välimäki, J. Pakarinen, C. Erkut, and M. Karjalainen, "Discrete time modeling of musical instruments," *Rep. Prog. Phys.*, vol. 69, no. 1, pp. 1–78, Jan. 2006.

[10] R. Rabenstein and S. Petrausch, "Block-based physical modeling," in *Proc. 5th Int. Symp. Mathematical Modeling (MATHMOD)*, I. Troch and F. Breitenecker, Eds., Vienna, Austria, 2006, pp. 2-1–2-17.

[11] N.H. Fletcher and T.D. Rossing, *The Physics of Musical Instruments,* 2nd ed. New York: Springer-Verlag, 1998.

[12] European Union shared cost project ALMA, IST-2001-33059, ALgorithms for the Modeling of Acoustic interactions, 2004, [Online]. Available: http://www-dsp.elet.polimi.it/alma/.

[13] "Simulink" [Online]. Available: http://www.mathworks.com/products/simulink/

[14] Pure Data (PD) [Online]. Available: http://puredata.info/

[15] S.D. Bilbao, *Wave and Scattering Methods for the Numerical Integration of Partial Differential Equations*. New York: Wiley, May 2004.

[16] A. Fettweis, "Wave digital filters: Theory and practice," *Proc. IEEE*, vol. 74, no. 2, pp. 270–327, 1986.

[17] K. Ochs, "Passive integration methods: Fundamental theory," *Int. J. Electron. Commun. (AEÜ)*, vol. 55, no. 3, pp. 153–163, 2001.

[18] A. Macchelli, R. Pasumarty, and A. van der Schaft, "Control of port Hamiltonian systems by interconnection and energy shaping via generation of Casimir functions. An overview," in *Proc. 5th Int. Symp. Mathematical Modeling (MATHMOD)*, I. Troch and F. Breitenecker, Eds., Vienna, Austria, Feb. 2006, pp. 5-1–5-11.

[19] L. Trautmann and R. Rabenstein, "Stable systems for nonlinear discrete sound synthesis with the functional transformation method," in *Proc. Int. Conf. Acoustics, Speech, Signal Processing. (ICASSP'02)*. IEEE, 2002, pp. 1861–1864.

[20] S.D. Bilbao and J.O. Smith, "Energy-conserving finite difference schemes for nonlinear strings," *Acta Acust. United Acust.*, vol. 91, no. 2, pp. 299–311, Mar. 2005.

[21] S. Petrausch and R. Rabenstein, "Interconnection of state space structures and wave digital filters," *IEEE Trans. Circuits Syst. II*, vol. 52, no. 2, pp. 90–93, Feb. 2005.

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

[23] L. Trautmann and R. Rabenstein, *Digital Sound Synthesis by Physical Modeling Using the Functional Transformation Method*. Norwell, MA: Kluwer, 2003.

[24] R. Rabenstein and L. Trautmann, "Digital sound synthesis of string instruments with the functional transformation method," *Signal Process.*, vol. 83, no. 8, pp. 1673–1688, Aug. 2003.

[25] S. Petrausch and R. Rabenstein, "A simplified design of multidimensional transfer function models," in *Int. Workshop Spectral Methods Multirate Signal Processing (SMMSP2004)*, Vienna, Austria, Sept. 2004, pp. 35–40.

[26]S. Petrausch and R. Rabenstein, "Application of block-based physical modeling for digital sound synthesis of string instruments," in *Audio Engineering Society (AES) 118th Conv.*, Barcelona, Spain, May 2005, AES paper 6365.

[27] G.D. Sanctis, A. Sarti, and S. Tubaro, "Automatic synthesis strategies for object-based dynamical physical models in musical acoustics," in *Digital Audio Effects (DAFx-03)*, London, Sept. 2003, pp. 198–202.

[28] G.D. Sanctis, A. Sarti, G. Scarparo, and S. Tubaro, "Automatic modeling and authoring of nonlinear interactions between acoustic objects," in *4th Int. Workshop on Multidimensional Systems (NDS 2005)*, Wuppertal, Germany, July 2005.

[29] C. Cadoz, A. Luciani, and J.-L. Florens, "CORDIS-ANIMA: A modeling and simulation system for sound and image synthesis—The general formalism," *Comput. Music J.*, vol. 17, no. 4, pp. 19–29, 1993.

[30] M. Vollmer, "Automatic generation of wave digital structures for numerical integrating linear symmetric hyperbolic pdes," *Multidimensional Syst. Signal Process.*, vol. 16, no. 4, pp. 369–396, Oct. 2005.

[31] D. Fränken, J. Ochs, and K. Ochs, "Generation of wave digital structures for networks containing multiport elements," *IEEE Trans. Circuits Syst. I*, vol. 52, no. 3, pp. 586–596, Mar. 2005.

[32] A. Sarti and G.D. Poli, "Toward nonlinear wave digital filters," *IEEE Trans. Signal Processing*, vol. 47, no. 6, pp. 1654–1668, June 1999.

[33] F. Pedersini, A. Sarti, and S. Tubaro, "Object-based sound synthesis for virtual environments using musical acoustics," *IEEE Signal Processing Mag.*, vol. 17, no. 6, pp. 37–51, Nov. 2000.

[34] A. Sarti and G.D. Sanctis, "Structural equivalences in wave digital systems based on dynamic scattering cells," in *14th European Signal Processing Conf. (EUSIPCO-06)*, Florence, Italy, Sept. 2006 [CD-ROM].

[35] A. Sarti and G.D. Sanctis, "Memory extraction from dynamic scattering junctions in wave digital structures," *IEEE Signal Process. Lett.*, vol. 13, no. 12, pp. 729–732, Dec. 2006.

[36] A. Sarti and G.D. Sanctis, Systematic methods for the implementation of nonlinear wave digital structures, submitted for publication [Online]. Available: http://www-dsp.elet.polimi.it/share0/IEEE-SPM/Sarti-TrCAS06.pdf

[37] J.O. Smith III, Physical audio signal processing: For virtual musical instruments and digital audio effects, 2006 [Online]. Available: http://ccrma.stanford.edu/~jos/pasp/

[38] M. Karjalainen, "BlockCompiler—A tool for physical modeling and DSP" [Online]. Available: http://www.acoustics.hut.fi/software/BlockCompiler/

[39] M. Karjalainen and C. Erkut, "Digital waveguides versus finite difference structures: Equivalence and mixed modeling," *EURASIP J. Appl. Signal Process.*, vol. 2004, no. 7, pp. 978–989, July 2004.

[40] M. Karjalainen, "BlockCompiler: Efficient simulation of acoustic and audio systems," in *Proc. 114th AES Conv.*, Amsterdam, Netherlands, Mar. 2003, AES paper 5756.

[41] M. Karjalainen, "BlockCompiler: A research tool for physical modeling and DSP," in *Proc. COST G6 Conf. Digital Audio Effects*, London, Sept. 2003, pp. 264–269.

[42] M. Karjalainen, C. Erkut, and L. Savioja, "Compilation of unified physical models for efficient sound synthesis," in *Proc. ICASSP*, vol. 5, Hong Kong, Apr. 2003, pp. 433–436.

[43] D.A. Jaffe and J.O. Smith III, "Extensions of the Karplus-Strong plucked-string algorithm," *Comput. Music J.*, vol. 7, no. 2, pp. 56–69, 1983.

[44] M. Karjalainen, V. Välimäki, and T. Tolonen, "Plucked-string models: From the Karplus-Strong algorithm to digital waveguides and beyond," *Comput. Music J.*, vol. 22, no. 3, pp. 17–32, 1998.

[45] S. Petrausch and R. Rabenstein, "Wave digital filters with multiple nonlinearities," in *Proc. 12th European Signal Processing Conf. (EUSIPCO-2004)*, Vienna, Austria, Sept. 2004, pp. 77–80.

SP