A Real-Time Implementation of Physical Models

Frederick L. Malouf
Apple Computer, Inc.
355 S. Mariani Ave., MS: 71AB
Cupertino, CA 95014
malouf@apple.com

Mark Lenczner
Braying at the Moon Studio
1209 Villa St.
Mountain View, CA 94041

Chris Chafe
CCIRMA/Music Department
Stanford University
Stanford, CA 94305
CC@CCIRMA.Stanford.edu

ABSTRACT: A new approach to Physical Modeling synthesis is discussed. Its
main objectives are unification of different types of physical models (such as clarinet
and string models), unification of static external signals (envelopes and sampled
sounds) and internal dynamic signals (delay lines), and the ability to be implemented
in real time on modern signal processors. The equations for the synthesis method
and two example instruments are presented.

We set out to implement Physical Modeling synthesis for users in a gigantic environment
with the ability to construct a large number of instruments and sounds. After reviewing the previ-
ous physical models of McIntyre, Schumacher, and Woodhouse, and Smith, and exploring the
needs of users, we were lead to the following three goals: 1) Unify the various forms of junctions
(i.e. string and wind). 2) Include envelopes and sound samples anywhere in an instrument, and 3)
Create generic code units that could be used for any configuration.

To meet these goals we created an instrument modeling set that consists of two elements:
types: junctions and signals. All the elements required to meet our design goals fall into these two
categories. This ensures that a small number of DSP code modules can efficiently implement a
wide variety of physical models, and that users creating new instruments face minimal restrictions
in their construction.

Junctions

The junction in our system is similar to those in other physical model implementations: it
has one or more pairs of inputs and outputs and functions for computing the outputs from the in-
puts on each sample. The difference is in the functional form (shown below) that allows us to use
the same function for all input/output pairs (including excitation control inputs), and use the same
junction code for all junction types (i.e. string contact points, reeds, etc.).

The form of our junction is derived from Smith (72). However, in our formulation all inputs
to the junction come from signal element input/output pairs. Mouth pressure and bow velocity are
not treated as special inputs in the junction equations.

(1) \( a_i^- = s_j \cdot a_j^+ \) where: \( a_i^- \) is the outgoing wave of the \( j \)th connected delay line
\( s_j \) is the incoming wave of the \( j \)th connected delay line
\( a_j \) is the resultant junction value

(2) \( a_j = \Delta \cdot f(\Delta) \)

(3) \( \Delta = \sum_{i=1}^{N} a_i^- \) the sum of the inputs

ICMC GLASGOW 1990 PROCEEDINGS
In many instances the functional part of the junction value can be a simple constant:

(4) \[ x_j = \Delta \times K \]

Our equation (1) easily implements the clarinet model in Smith’s work. However, the string model cannot be expressed in the form of equation (1). We show here that our equations can be used to create an equivalent model. To clarify the difference, the subscript ‘o’ stands for our equations while the ‘i’ stands for theirs (Smith’s).

We start with the following difference:

\[
\begin{align*}
\Delta_{o} &= w_{o} + a^{o} + b^{o} \\
\Delta_{i} &= w_{i} - (a^{i} + b^{i})
\end{align*}
\]

where \( a^{o} \) and \( b^{i} \) are the velocity inputs from the string and \( w \) is the bow velocity. Our equation (5a) is in the form of equation (1). We sum the inputs to get the differential velocity whereas theirs has a subtraction. These equations lead us to assume that:

\[
\begin{align*}
w_{o} &= w_{i} \\
\Delta_{o} &= \Delta_{i}
\end{align*}
\]

The output equations show the fundamental difference between the two models. Their output is computed from the other (non-corresponding) input, whereas ours is computed from the matching input. (This makes theirs specific to two inputs, and ours generalizable to N).

\[
\begin{align*}
a^{o} &= f_{o}(\Delta_{o}) \cdot a^{i} + b^{o} \\
a^{i} &= f_{i}(\Delta_{i}) \cdot a^{i} + b^{i}
\end{align*}
\]

Notice that (7a) corresponds to (1) and (2). We want the output of the two models to be the same, so set equations (7a) and (7b) equal and solve for \( f_{o} \):

\[
\begin{align*}
\Delta_{o} &= f_{i}(\Delta_{i}) \cdot a^{i} + b^{i} \\
\Delta_{o} &= f_{i}(\Delta_{i}) \cdot a^{i} + b^{o}
\end{align*}
\]

Using the substitutions from (5a) and (6b) we get:

\[
\begin{align*}
f_{o}(\Delta_{o}) &= \left( f_{i}(\Delta_{i}) - \frac{a^{i} + b^{i}}{w_{i} + a^{i} + b^{i}} \right)
\end{align*}
\]

To solve this we make following simplification:

\[
\frac{x}{k + x} = \frac{a^{i} + b^{i}}{w_{i} + a^{i} + b^{i}}
\]

If \( x \gg x \) then (10) = 1/k. If \( x \gg k \) then (10) = 1. It is valid to assume that the bow velocity \( w \) is slow; moving or static with respect to \( a^{i} \) and \( b^{i} \), which are the string velocity waves. Therefore (10) is a constant we will call \( c \) (chosen based on the relative magnitudes of \( x \) and \( k \)) and we have the following:

\[
\begin{align*}
f_{o}(\Delta_{o}) &= f_{i}(\Delta_{o}) - c
\end{align*}
\]

In the common case that \( c = 1 \) or

\[
\begin{align*}
a^{i} + b^{i} &= w_{i} + a^{i} + b^{i}
\end{align*}
\]

and from (9) we have

\[
\begin{align*}
f_{o}(\Delta_{o}) &= f_{i}(\Delta_{o}) - 1
\end{align*}
\]

Therefore, our function is the function, inverted about the y axis, translated down 1, and inverted about the x axis. And we can implement a string with our equations.

**Signal Elements**

The signal elements in our system can take several different forms: classic physical model bidirectional lines (for strings and wind bera), radiator elements (bells and soundboards), and

**ICMC GLASGOW 1996 PROCEEDINGS**
sampled signals (envelopes for mouth pressure or bow velocity, sampled sounds). Again, a single consistent interface allows us to freely interchange any of these elements in a instrument model without altering the implementation of the junctions or other delay lines.

The delay line interface consists of input and output registers and the ability to clock each sample. A simple delay line shifts the samples, preserving an input N samples later at the output. A junction source, which may be an envelope or sampled sound, presents successive values at the output while ignoring its input. Sample input and output are implemented as two sample delay lines with the side effect of requiring an A/D or writing a D/A.

Filters are implemented as special forms of delay line. On each sample, the contents of the delay memory are convolved with a set of coefficients to obtain the output. Filters are usually used in conjunction with another form of delay line, called a filter return, which presents the filter's output (or as opposite: high pass if the filter was low pass) to another junction.

We close with two examples of how these elements can be connected together. Note that
the junctions are all the same kind of element, with one, two, or three connections. In particular, the last two junctions of both models typically use a function, P(α)=1, which provides a simple pass through junction.

---
