Efficient computational modeling of piano strings for real-time synthesis using mass-spring chains, coupled finite differences, and digital waveguide sections

Smith, Kuroda, Perng, Van Heusen, Abel
CCRMA, Stanford University

ASA-2010

November 16, 2010
Goals

- **Overall Goal:** *Ultimate Virtual Piano*

- **Current Focus:** *Audibly Perfect Piano-String Synthesis*

- **Method:** *Start with High Accuracy, then Simplify*
Outline

- Brief History of Virtual Strings
  - Ideal Strings
  - Stiff Piano Strings
  - Nonlinear Piano Strings
- General Mass-Spring String Model
- Finite Difference Implementation
Brief History of String Sound Synthesis
D’Alembert’s PDE for the Ideal String (1747)

\[ K \frac{\partial^2 y}{\partial x^2} = \epsilon \frac{\partial^2 y}{\partial t^2} \]

i.e., \[ Ky'' = \epsilon \ddot{y} \]

\[ K \Delta \text{ string tension} \]
\[ \epsilon \Delta \text{ linear mass density} \]
\[ t = \text{time (s)} \]
\[ x = \text{position along string axis} \]

Wave equation PDE is derived from Newton’s second law \( f = ma \) by equating Brook Taylor’s “restoring force” \( Ky'' \) (1713) to mass-density times acceleration \( \epsilon \ddot{y} \).
In the same paper, d’Alembert also showed that the ideal-string PDE was satisfied by traveling waves in either direction:

\[ y(t, x) = y_r \left( t - \frac{x}{c} \right) + y_l \left( t + \frac{x}{c} \right) \]

where

\[ c = \sqrt{\frac{K}{\epsilon}} = \text{wave propagation speed} \]

These ideas were developed into essentially modern form by Euler (1707–1783).
We *sample* d’Alembert’s traveling-wave components to make *digital waveguide models*.

Lossless digital waveguide \( \triangleq \) *bidirectional delay line* at some wave impedance \( R = \sqrt{K/\epsilon} \).

Useful for *efficient* models of strings, bores, plane waves, conical waves, and more.
Digital waveguide string models are extremely efficient computationally \([\mathcal{O}(1)]\) complexity per sample of output.

- Load each delay line with half of initial string displacement.
- Sum of upper and lower delay lines = string displacement.
- Insert a linear, time-invariant filter in the loop for any desired attenuation and dispersion as a function of frequency.
Velocity waves $v$ easily converted to force waves $f$ which are proportional to string slope $y'$:

$$-K y' \triangleq f = f^+ + f^- = R v^+ - R v^-$$
Plucked/Struck Sound Example (1988)

Bach A-Minor Concerto—Orchestra Part: (WAV) (MP3)

- Executed in real time on one Motorola DSP56001
  (20 MHz clock, 128K SRAM)
Plucked/Struck Sound Example (1988)

Bach A-Minor Concerto—Orchestra Part: (WAV) (MP3)

- Executed in real time on one Motorola DSP56001 (20 MHz clock, 128K SRAM)

- Developed for the NeXT Computer introduction at Davies Symphony Hall, San Francisco, 1988
Plucked/Struck Sound Example (1988)

Bach A-Minor Concerto—Orchestra Part: (WAV) (MP3)

- Executed in real time on one Motorola DSP56001 (20 MHz clock, 128K SRAM)

- Developed for the NeXT Computer introduction at Davies Symphony Hall, San Francisco, 1988

- Solo violin part was played live by Dan Kobialka of the San Francisco Symphony
Stiff Strings

- Piano strings are *stiff*, resulting in significant *dispersion*:

\[
\epsilon \ddot{y} = Ky'' - \kappa y''''
\]  

(Stiff string PDE)

The new “stiffness term” is proportional to

\[
y'''' \triangleq \frac{\partial^4}{\partial x^4} y(t, x)
\]

1. *Faster wave propagation at higher frequencies*  
   (hence the “dispersion” of traveling-wave shapes)
   - Provided in digital waveguide strings using an *allpass filter* having a delay that decreases with frequency  
     (typically on the order of 10 poles and zeros)

2. *A spring-resistance to corner formation* in the string  
   (usually left out in string synthesis models!)

See, *e.g.*, Cremer 1984
Example waveguide string, three spatial-samples long:

\[ y^+(n) \rightarrow H(z) \rightarrow y^+(n-1) \rightarrow H(z) \rightarrow y^+(n-2) \rightarrow H(z) \rightarrow y^+(n-3) \rightarrow \cdots \]

\[ y^-(n) \leftarrow H(z) \leftarrow y^-(n+1) \leftarrow H(z) \leftarrow y^-(n+2) \leftarrow H(z) \leftarrow y^-(n+3) \leftarrow \cdots \]

\( (x = 0) \quad (x = c(\omega)T) \quad (x = 2c(\omega)T) \quad (x = 3c(\omega)T) \)

\[ H_a(z) = \text{allpass giving desired delay vs. frequency for one sample} \]
In practice, we pull out one or more samples of pure delay:

\[
\begin{align*}
  y^+(n) &\quad \overset{\text{z^{-1}}}{\rightarrow} \quad zH_a^3(z) \quad \overset{\text{z^{-1}}}{\rightarrow} \quad y^+(n-3) \\
  y^-(n) &\quad \overset{\text{+}}{\rightarrow} \quad y(nT,0) \quad \overset{\text{+}}{\rightarrow} \quad y(nT,3c(\omega)T) \\
  \text{(x = 0)} &\quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \text{(x = 3c(\omega)T)}
\end{align*}
\]

- \(zH_a^3(z)\) (or \(z^2H_a^6(z)\)) is designed as a single allpass filter
- \(\approx 10\) poles and zeros can handle hundreds of samples of dispersion with perceptual equivalence
- Not valid for distributed nonlinear behavior (discussed later)
Digital Waveguide Piano Sound Example

- Piano: (WAV) (MP3) (Stanford Sondius Project, ca. 1995)
Digital Waveguide Piano Sound Example

- Piano: (WAV) (MP3) (Stanford Sondius Project, ca. 1995)
- Uses the *commuted synthesis* technique
Nonlinear Piano Strings

- Vertical excitation by hammer couples nonlinearly to the longitudinal direction.\(^1\)

\[ \epsilon \ddot{\xi} = ES\xi'' + \frac{1}{2}ES[(y')^2]' \]  
(longitudinal wave PDE)

where \(\xi = \text{longitudinal displacement}\)

- Nonlinear coupling driven by

\[ [(y')^2]' \triangleq \frac{\partial}{\partial x} \left[ \frac{\partial}{\partial x} y(t, x) \right]^2 \]

- Slope of square of string slope drives longitudinal waves

- Coupling from longitudinal back to transverse is smaller and typically neglected

\(^1\)E.g., Morse & Ingard 1968
Effects of Nonlinear Mode Coupling in Piano Strings

Main audible effects of nonlinear transverse-to-longitudinal coupling:

1. Initial *longitudinal attack pulse*
   (the initial “shock noise” audible in a piano tone)

2. Inharmonic longitudinal modes

3. “Phantom partials”
   (ongoing intermodulation products from transverse partials)

See Conklin lecture in “Five Lectures on the Acoustics of the Piano,” edited by Anders Askenfelt (and listen to the sound examples)
Nonlinear Piano-String Synthesis

If one-way coupling is accurate enough (transverse to longitudinal but not vice versa), we can model its effects separately based on observations of transverse waves (Bank and Sujbert, JASA 2005):

- Longitudinal modes implemented as second-order resonators (“modal synthesis”)
- Slope of squared slope projected onto longitudinal modes

Up-to-date summary:

“A Modal-Based Real-Time Piano Synthesizer”
Balázs Bank, Stefano Zambon, and Federico Fontana
IEEE Trans. Audio, Speech, and Language Processing
May, 2010 (IEEE-ASLP)
Synthesis Strategies

Model complexity grows with dynamic level:

1. Linear Superposition: Transverse and longitudinal waveguides *decouple* into separate modes

2. Transverse → Longitudinal Coupling

3. Transverse ↔ Longitudinal Coupling
Synthesis Strategies

Model complexity grows with dynamic level:

1. Linear Superposition: Transverse and longitudinal waveguides \textit{decouple} into separate modes

2. Transverse $\rightarrow$ Longitudinal Coupling

3. Transverse $\leftrightarrow$ Longitudinal Coupling

Proposed Synthesis Strategy:

- Initial striking force determines the starting regime (1, 2, or 3)
- Maximum slope $|y'|$ over one or more periods indicates regime transitions
- String model simplifies as it decays
The preceding involved several approximations:

- Neglected terms in PDEs
- Simplified synthesis models
Checking the Approximations

The preceding involved *several* approximations:

- Neglected terms in PDEs
- Simplified synthesis models

The following questions naturally arise:

- How do we know for sure our approximations are inaudible?
- We can listen, but could we miss an audible effect?
- Could a difference become audible after more listening?
The preceding involved *several* approximations:

- Neglected terms in PDEs
- Simplified synthesis models

The following questions naturally arise:

- How do we know for sure our approximations are inaudible?
- We can listen, but could we miss an audible effect?
- Could a difference become audible after more listening?

What we really want is a *truth reference*—an “exact model”!
Checking the Approximations

The preceding involved several approximations:

- Neglected terms in PDEs
- Simplified synthesis models

The following questions naturally arise:

- How do we know for sure our approximations are inaudible?
- We can listen, but could we miss an audible effect?
- Could a difference become audible after more listening?

What we really want is a truth reference—an “exact model”!

- There are software tools (e.g., from perceptual audio coding) for measuring the audible equivalence of two sounds:
  - “Original” and “Encoded” for a CODEC
  - “Exact” and “Computationally Efficient” for piano models
Mass-Spring String Model
Mass-Spring Model in 3D Space

- Hooke’s Law:

\[ || \mathbf{f}_1 || = k \cdot |l_1 - l_0| \quad (l_0 = \text{spring rest length}) \]

- Vector Equation of Motion \( (f_i \in \mathbb{R}^3, \ x_i \in \mathbb{R}^3) \):

\[
\mathbf{f}_1 = k \cdot (|| \mathbf{x}_2 - \mathbf{x}_1 || - l_0) \cdot \frac{\mathbf{x}_2 - \mathbf{x}_1}{|| \mathbf{x}_2 - \mathbf{x}_1 ||} \\
= k \left[ 1 - \frac{l_0}{|| \mathbf{x}_2 - \mathbf{x}_1 ||} \right] (\mathbf{x}_2 - \mathbf{x}_1) = m_1 \ddot{x}_1
\]
Mass-Spring String Model

\[
\begin{align*}
\mathbf{f}_{i-1} & \rightarrow \mathbf{f}_i \rightarrow \mathbf{f}_{i+1} \\
\cdots & \quad m \quad k \quad m \quad k \quad m \quad k \quad m \quad \cdots \\
\mathbf{x}_{i-1} & \quad \mathbf{x}_i \quad \mathbf{x}_{i+1}
\end{align*}
\]

\[
\mathbf{f}_i = \alpha_i \cdot (\mathbf{x}_{i+1} - \mathbf{x}_i) + \alpha_{i-1} \cdot (\mathbf{x}_{i-1} - \mathbf{x}_i)
\]

\[
= \alpha_{i-1} \mathbf{x}_{i-1} - (\alpha_{i-1} + \alpha_i) \mathbf{x}_i + \alpha_i \mathbf{x}_{i+1}
\]

where

\[
\alpha_i \triangleq k \cdot \left[ 1 - \frac{l_0}{\| \mathbf{x}_{i+1} - \mathbf{x}_i \|} \right]
\]
Adding Stiffness

- Goals
- Outline

String Synth History

Mass-Spring Model
- 3D Mass-Springs
- Mass-Spring String
- Adding Stiffness
- Shear Stiffness
- Three-Spring Design
- Shear Springs

Finite Differences
Need Shear Stiffness Too
Three-Spring Design (Before Shear Springs Added)
Shear Springs

Goals

Outline

String Synth History

Mass-Spring Model

- 3D Mass-Springs
- Mass-Spring String
- Adding Stiffness
- Shear Stiffness
- Three-Spring Design
- Shear Springs

Finite Differences
Finite Difference Implementation
Digitizing the Flexible String

Recall:

\[ f_i = \alpha_i \cdot (x_{i+1} - x_i) + \alpha_{i-1} \cdot (x_{i-1} - x_i) \]

\[ = \alpha_{i-1} x_{i-1} - (\alpha_{i-1} + \alpha_i) x_i + \alpha_i x_{i+1} \]

where

\[ \alpha_i \triangleq k \cdot \left( 1 - \frac{l_0}{\|x_{i+1} - x_i\|} \right) \]
Equations of Motion, State-Space Form

\[ f_1 = m_1 \ddot{x}_1 = \alpha_1 \cdot (x_2 - x_1) \]

\[ \vdots \]

\[ f_i = m_i \ddot{x}_i = \alpha_{i-1} x_{i-1} - (\alpha_{i-1} + \alpha_i) x_i + \alpha_i x_{i+1} \]

\[ \vdots \]

\[ f_M = m_M \ddot{x}_M = \alpha_{M-1} \cdot (x_{M-1} - x_M) \]

or \((3M \times 1)\):

\[ \mathbf{F} = \mathbf{M} \ddot{\mathbf{X}} = \mathbf{A} \mathbf{X} \]

Digitization (Explicit Finite Difference Scheme):

\[ \mathbf{X}_{n+1} = [2\mathbf{I} + \mathbf{M}^{-1} \mathbf{A}] \mathbf{X}_n - \mathbf{X}_{n-1} + \mathbf{B} \mathbf{u}_n \]

where \( \mathbf{u}_n \) is the external input vector, and digitization is based on

\[ \ddot{x}_n \triangleq x_{n+1} - 2x_n + x_{n-1} \]
Results to Date

Presented this morning by Junji Kuroda
Summary

- String-Synthesis History Reviewed
  - Ideal Strings
  - Stiff Piano Strings
  - Nonlinear Piano Strings

- Mass-Spring Model = Accurate Benchmark Reference
  - Accuracy $\sim$ time/space sampling density
  - Approximations can be quantified in practical cases
  - Given faster and more parallel computing, maybe we’ll just go ahead and use it for real-time sound synthesis from performance!