CENTER FOR COMPUTER RESEARCH IN MUSIC AND ACOUSTICS
DECEMBER 1993

Department of Music
Report No. STAN-M-81

PHYSICAL MODELING AND SIGNAL PROCESSING

CCRMA PAPERS PRESENTED AT THE 1993 INTERNATIONAL COMPUTER MUSIC CONFERENCE

Chris Chafe, Perry R. Cook, Dexter Morrill,
Julius O. Smith, Scott A. Van Duyne

CCRMA
DEPARTMENT OF MUSIC
Stanford University
Stanford, California 94305

CCRMA Papers on Physical Modeling and Signal Processing from the 1993 International Computer Music Conference

Chris Chafe: Tactile Audio Feedback

Perry Cook: IGDIS (Instrument for Greek DIction and Singing): A Modern Greek Text to Speech/Singing Program for the SPASM/Singer Instrument

Perry Cook, Dexter Morrill, and Julius Smith: A MIDI Control and Performance System for Brass Instruments

Julius Smith: Efficient Synthesis of Stringed Musical Instruments

Scott Van Duyne and Julius Smith: Physical Modeling with the 2-D Digital Waveguide Mesh

Tactile Audio Feedback

Chris Chafe
Center for Computer Research in Music and Acoustics,
Department of Music, Stanford University, Stanford, CA 94305

Abstract

Vibrotactile feedback is common in handheld musical instruments. Useful cues received by the player via this sense are presented, and a design which incorporates vibrotactile feedback in new controllers for physical models is proposed.

I. Introduction

Quite some time ago, in many forms, humans evolved a medium of expression through whacking, plucking, blowing and bowing various acoustic and mechanical oscillators. Adept control of the vibration of these systems is as necessary to music as it is to precise vocal communication. Like the voice, handheld instruments are intimately connected with our sense of touch. Primary feedback arrives by ear, but the *feel* of crafting a sound from a brass, wind or stringed instrument is an important secondary sense and is learned early on in training. Resistances and "give" are felt kinesthetically and vibration arrives directly through the tactile sense. Contact points in the cello, for example, are five: two legs, two hands and chest. These points register motion that adds to the player's sense of the inssrument's response to controlling gestures.

II. Background

The psychophysics of the vibrotactile sense has been described at length in the literature. Verrillo in [1] presented a review of the field framed by a discussion of issues relevant to musical performance. These general concepts are of importance in the following discussion:

- The fingers are among our most sensitive sites.
- Frequency response ranges from near 0 to approximately 1000 Hz.
- Frequency discrimination is very poor.
- The subjective sensation changes across frequency bands.

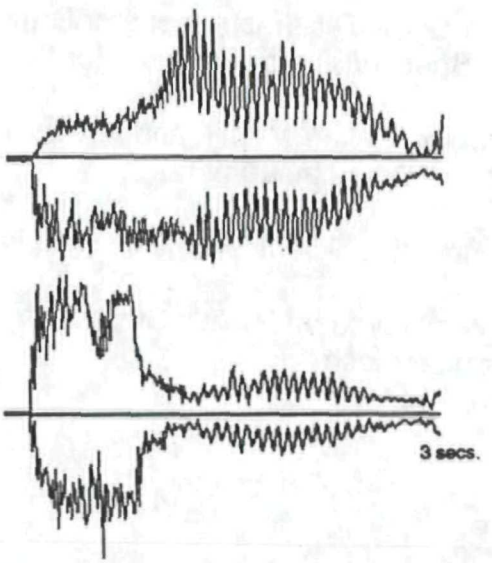


Figure 1: Tone 1 amplitude plots are from an 880 Hz. natural harmonic played on the celletto. Top: bridge transducer. Bottom: left-hand index finger accelerometer.

Amplitude sensitivity measured with sinusoidal stimuli varies with location. It is suggested that high sensitivity, such as has been measured at the fingers, is in relation to representation area in the somatosensory cortex [2]. Four independent physiological channels are known and are separable with regard to amplitude and frequency sensitivity [3]. Differences may play a role in bracketing vibrations of an instrument into distinct cues, as will be shown to occur below.

A mound-shaped curve describes overall cutaneous frequency sensitivity between 0.3 and 1000 Hz. The region of best sensitivity extends from about 100 to 500 Hz. Frequency discrimination is poor compared with the ear: The finger is only able to detect differences on the order of 20 or 30%. The quality of the sensation changes from a localizable "buzz" below about 100 Hz. to a diffuse, smoother sensation for higher frequencies. From various earlier studies, it can be concluded that only certain musical dimensions are representable, specifically

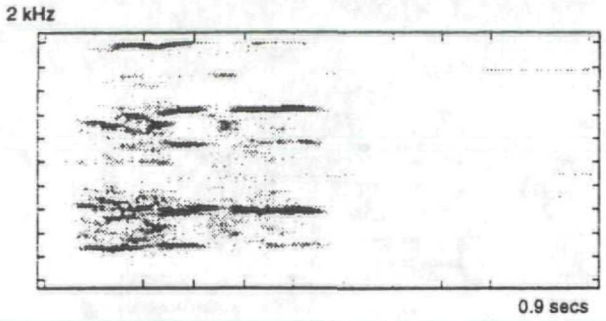


Figure 2: Tone 1. Low frequency components are shown in a spectogram of finger vibration at note onset. These components disappear as the stable oscillation sets up (seen after the cursor mark).

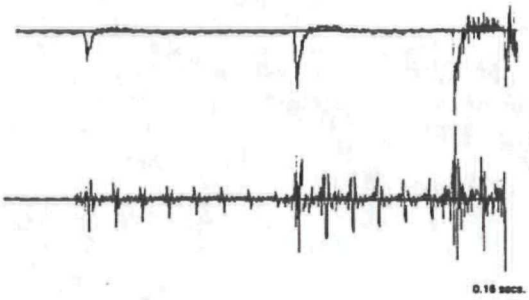


Figure 3: Tone 2. Amplitude plots are shown of first episode (see text) of a note onset at 110 Hz. Top: bridge transducer. Bottom: left-hand index finger accelerometer.

timing, amplitude, and spectral weighting, but not precise pitch.

III. A Cellist's Left Hand

Recordings were made of cello tones to discover some functional vibrotactile cues. The fingertip was chosen for an initial site because of its good sensitivity. Tones were played arco on the celletto, an electronic cello, and two channels were digitized simultaneously at a sampling rate of 44.1 kHz. Output from the instrument was recorded with a bimorph piezoceramic bridge transducer designed by Max Mathews. Finger motion was obtained from an accelerometer (PCB model 330a) affixed to the nail of the index finger which was stopping the string.

The following analyses demonstrate two cues by which a cellist senses stability of oscillation. Tone 1 is a high pitched natural harmonic (played with the finger stopping the string lightly, not fully to the fingerboard) and sounding 880 Hz. (**A5**), played on the instrument's first string (**A3**). Tone 2 is fully stopped at 110 Hz. (**A2**) and played on the third string. Both tones were played with intentionally long bow attacks to exaggerate the note onset transient.

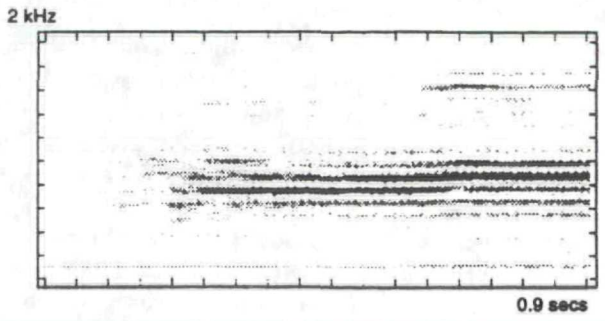


Figure 4: Tone 2. The spectogram shows the progression of finger vibration from note onset to stable Helmholtz motion. The pitch starts high (sporadic episode), dips below (unstable Helmholtz episode), and then arrives at the final fundamental pitch.

Tone 1 is pitched above the upper frequency limit for sensation. Continuous vibration is only felt for tones pitched below about a perfect fifth lower than this note. Despite this fact, transients are still felt at pitches that are too high. Note onsets, bow direction changes and abrupt stops are sensed as brief vibrations at the fingertip. Dual amplitude plots of Tone 1 in Figure 1 contrast the output signal and the signal that passes through the fingertip. From the noisy onset into strong stable oscillation the bridge waveform shows amplitude growth. However, the fingertip recording shows a low-pass filter response: diminishing amplitude as the oscillation locks in on a pitch that is too high. During the transient, low components that are in the region of sensitivity are transmitted through the finger. These are seen in the spectogram of Figure 2. At this pitch, the cellist has a cue that discriminates transient events from stable oscillation through presence vs. absence of vibration.

Tone 2 is pitched to lie with at least 7 of its harmonics in the region of sensitivity. Continuous vibration *is* felt through the entire course of this tone. The transient portion is still sensed as a discrete event though the cueing signal is different. As pointed out above, vibration quality can change at around 100 Hz. as it does here, from a "rough and aperiodic" transient to a "smooth and regular" stable oscillation. Furthermore, the note onset itself consists of two distinct episodes before leading to stable, periodic Helmholtz (stick/slip) motion. Initial sporadic releases are followed by an interval of very unstable Helmholtz motion with a flat pitch.

The sporadic release episode shows a surprising difference between the two recorded channels: Where the bridge sees only isolated releases, the fingertip feels a plucked periodic vibration at 123 Hz.(**B2**) one whole tone higher than the actual pitch. The phenomonen results from the bow hair sticking to the string immediately after a quasi-

pizzicato, causing the string to be split into two portions. The felt pitch corresponds to the string length between fingertip and bow. Confirmation was made by studying the resulting pitch at different bow contact positions. Figure 3 shows the first episode.

During the second episode, the tone exhibits flattened, aperiodic Helmholtz motion. The fingertip waveform is more complex because of competing, incommensurate oscillations. In Figure 4, Helmholtz motion (pitch **A2**) takes over from the earlier plucked string motion (pitch **B2**) and brings in lower components.

Both tones confirm that fingertip vibration (or lack of vibration) can be used to gauge the time and length of articulation. Depending on the note played (pitch, note fingering, etc.) finger motion was found to provide cues through amplitude and spectral content. The player interprets cues in relation to the specific note. For example, the same message concerning oscillation stability will be received as presence/absence or smoothness/roughness cues, depending on pitch height.

IV. Physical Models

With the advent of physical models for synthesis, the world of electronic sound generation has a new class of "unpredictable" instruments. The same unpredictability is found to some degree in most traditional musical instruments and is easily summarized as the "french horn problem." Unruly overblown notes on the horn are an extreme example of an oscillation going one way when the performer wishes to go another. The family of real time physical models developed at CCRMA exhibits this independence in all its members – it is an article of faith in the theory of oscillating nonlinear systems that this is "a feature, not a bug." Incorporating vibrotactile feedback addresses specifically the problem of performing on instruments that are not purely deterministic.

Controllers that have been attached to CCRMA's physical models include MIDI modulation wheels, MIDI keyboard aftertouch, mouse-controlled computer panels and homemade gear such as Cook's WhirlWind instrument [5]. The hand controlling a synthesis parameter locates a particular value either be ear or combined with a coarse sense of position (which may depend also on the eye watching a cursor). Position itself is relatively coarse compared to the model's sensitivity to some parameters. Worse yet, the models often do not respond identically to a precisely repeated parameter value, since system state interacts with reponse in the physical modeling world. The models exhibit multiple possible regimes of oscillation for a given set of parameter values.

The *electronic* french horn problem is presently much worse than the natural one. The *lip tension* parameter of CCRMA's nearest model, *HosePlayer*, determines in part which overblown note will sound. Using the controllers listed above (with a hand controlling lip tension) we have yet to hear anyone play Taps (a bugle call) without a mistake. The only possible feedback is crossmodal (ear/finger). Normally, when playing a brass instrument, *lip tension control* and the *lip reed* producing the sound would be intimately associated. Effort injected into the oscillator would be metered directly by vibrotactile sensation at the point of excitation. Instead, an electronic controller is employed which is either "dead" in this sense or imparts vibration and resistance of its own kind, and which are not derived from the oscillating system.

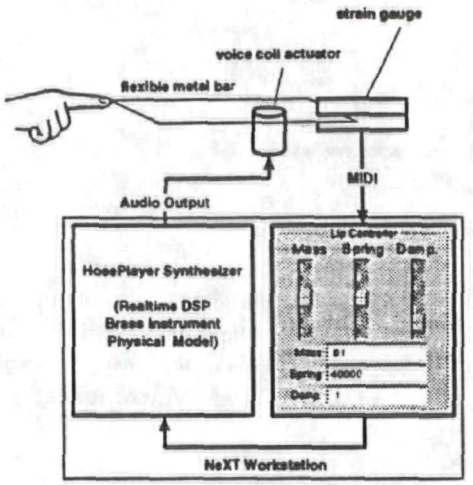


Figure 5: Tactile Audio Feedback is used to control the lip tension parameter of the brass physical model *HosePlayer*.

V. Tactile Audio Feedback

An inital test has been performed to see if the situation improves with addition of vibrotactile feedback by creating a direct control loop at the finger tip. The setup is diagrammed in Figure 5. Depressing a flexible metal bar corresponds to a change in lip tension. Audio output of the model is fed back to a voice coil actuator that vibrates the metal bar. With the finger depressing the bar and feeling the output of the oscillation, adept maneuvers of lip tension are possible (Taps is much more playable). Turning off audio feedback to the actuator removes vibrotactile feedback and causes the situation to revert back to imprecision. Most of the pitches are above frequency cutoff for the vibrotactile sense.

Figure 6 shows a spectogram of a portion of a

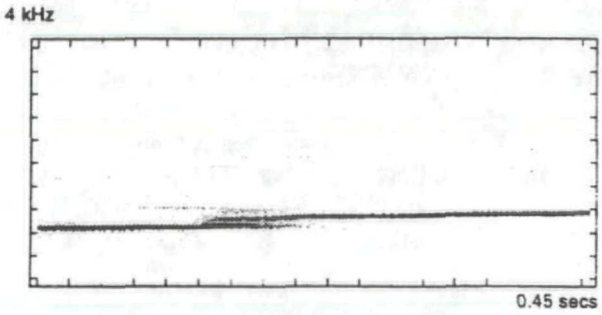


Figure 6: A spectogram of a lip tension glissando shows subharmonic components when the pitch shifts to the next higher harmonic.

harmonic glissando produced by changing lip tension, as synthesized with Cook's TBone program [4]. Brief bursts of energy support subharmonics that lie below cutoff at note transitions and harmonics of subharmonics are visible. The finger on the controller experiences these moments as "buzzes" or "bumps" when the overblown harmonic changes but feels nothing of the sustained tones between. Feedback to the performer consists of the same cue as Tone 1 in the cello analysis above.

VII. Conclusion

Two vibrotactile cues have been explored. Certainly the number of cues is larger when taking into account the full range of an instrument's sonic possiblities. Feedback concerning oscillation timing and quality has been found. The experiment incorporating vibrotactile feedback in the controller for a realtime physical model of a brass instrument can be extended simply in more sophisticated controllers: Feed the audio output of the synthesis back to the controlling device so that the musician feels the oscillation. The result will improve a player's perception of when the oscillator speaks and how it speaks. Controllers that communicate to the sense of touch can also incorporate kinesthetic forces [6] [7]. Good tools lend themselves to skillful operation – future work aims at affording better control to performers of synthetic electronic musical oscillators.

References

[1] R. Verrillo. "Vibration Sensation in Humans." Music Perception, Vol. 9, No. 3, pp. 281-302: 1992

[2] C. Sherrick and R. Cholewiak. "Cutaneous Sensitivity," in Handbook of Perception and Human Performance, Vol I, pp. 12-1 12-58: 1986

[3] J. Bolanowski, et al. "Mechanical aspects of touch," JASA, Vol. 88, No. 5, pp. 1680-1694: 1988

[4] P. Cook. "TBone: An Interactive WaveGuide Brass Instrument Synthesis Workbench for the NeXT Machine," Proceedings ICMC 91, Montreal, pp. 297-299: 1991

[5] P. Cook. "A meta-wind-instrument physical model, and a meta-controller for real time performance control." Proceedings ICMC 92, San Jose, pp. 273-276: 1992

[6] B. Gillespie. "The Touchback Keyboard," Proceedings ICMC 92, San Jose, pp. 447-448: 1992

[7] C. Cadoz, L. Liswoski, and J-L. Florens. "Modular Feedback Keyboard." Proceedings ICMC 90, Glasgow, pp. 379-382: 1990

IGDIS (Instrument for Greek DIction and Singing): A Modern Greek Text to Speech/Singing Program for the SPASM/Singer Instrument

Perry Cook (prc@ccrma.stanford.edu) Stanford CCRMA
Dimitris Kamarotos, Taxiarchis Diamantopoulos, Giorgos Philippis Athens CCMR

Abstract

A Greek language text reading program has been constructed which generates control files for the SPASM/Singer physical model of the human singing voice. A simple rule system for Modern Greek determines pronunciation. The Greek text is processed in stream fashion, yielding a text file which controls the Singer instrument. Note names, frequencies, or functions specifying pitch are imbedded directly into the text using <> brackets. Note durations are extended with dashes. Tuning systems based on a large number of Greek microtonal modes have been specified. A number of commonly used Greek vocal ornaments have been specified as C functions, and can be passed to the Singer instrument using {} braces. The system allows for easy creation of new ornaments, figures, words, and tuning systems.

1 Introduction

IGDIS "Instrument for Greek DIction and Singing" is based on the LECTOR Ecclesiastical Latin to Speech and Singing Project [Cook, 1991]. The goal of the Greek Singer project is to provide simple means for creating control files for a physical model of the human singing voice, with emphasis on the Greek language and Greek vocal music.

Greek language has a historical development of 3000 years, and is still an actively used and evolving language, therefore the task is much more complex than that for Ecclesiastical Latin. Greek has met various influences from other languages, but has kept some very specific characteristics. Among those characteristics are a large number of primal roots and complex patterns of pitch and amplitude that resemble the complexity of the singing voice. A good Greek language synthesizer should be very close to singing!

The IGDIS project provides a user-friendly tool that can be used for research in Greek folk and Byzantine Ecclesiastical music as well as for an experimental approach of the prosodic and rhythmic characteristics of ancient Greek language and music. Some features very frequently encountered in all forms of Greek music are the mode-like "scales", with non symmetrical intervals that differ from "well tempered" western semitones and tones, the very frequent monodic environment, and the formalized ornamental richness. These "ornaments" are not of a decorative nature as most commonly used in western music, but they construct complex melodic lines called "melisma" [Karas, 1988].

A big part of this musical context has been analyzed and categorized ,so that it is possible to produce special routines that simulate these characteristics of vocal behavior [Baud-Bovy, 1984].

2 Text to Speech

The IGDIS project provides simple means of data input for the SPASM/Singer articulatory-controlled voice synthesis systems. The SPASM and Singer projects are described in [Cook, 1989, 1990, 1991]. The necessary modern Greek language rules for the purposes of Greek speech are included, as well as those for Greek singing. Moreover a set of Greek folk music Singer subroutines are also included to model most of the commonly encountered vocal ornaments.

The program enables the user to enter the desired text for speech/singing in ASCII text. The text is then processed in stream fashion, yielding a text file which controls the Singer instrument. Similar to common vocal notation practice, durations can be extended with dashes. All phonemes and dashes yield a default synthesis duration of 0.2*vtRef or 0.2*ctRef seconds. The default for vtRef and ctRef are 0.5 seconds, yielding a 100 millisecond phoneme frame. As an example, the following yields a soundfile 2.2 seconds long:

κα——λι——με——ρα——

3 Pitch Information

Frequencies (440.0) or note names, both as in western (c4) and Byzantine music (bou1) notation, are imbedded directly in the text using <> brackets before the phoneme associated with the target pitch:

<c4>ka--<bou1>--l<440.0>i--m<120.7*sin(1.5)>e----ra--

Enclosing a message in curly braces places it verbatim into the Singer instrument control stream. This could be used for changing the time references, or globally controlling the amplitude of consonants:

{ vtRef = 0.15; ctRef = 0.25; nAmp = 0.75;}

The Greek input text is then processed by the Greek parser program. This program serves not only as a parser for the input text file, but also applies modern Greek pronunciation and phonetic rules, and finally converts the input code into the corresponding Singer control code. The major principles of modern Greek pronunciation have been applied. In a large number of cases the pronunciation of a single character depends on the characters immediately before or after the current one. Other cases include double consonants, diphthongs etc. The use of "transitional" diphones between two successive phonemes is one of the main characteristics of Greek speech. This means that the total number of the original phonemes as well as the transitional diphones is very large, so a practical Greek vocal synthesizer must make use of smoothing routines rather than trying to include all possible phoneme combinations. The Singer instrument automatically provides means for interpolating between target positions in phoneme space, where the control space is that of vocal tract shape (articulatory space). Smooth and continuous transitions in this space inherently provide more natural synthesis, because unlike transitions in many other speech representation spaces, motions in articulatory space are always smooth and continuous.

This IGDIS input code:

```
<c4>ka---<bou1>---li<440.0>--me<120.7*sin(1.5)>------ra---
```

yields this Singer code:

```
#define Syn synthesize
#define LG lastGlot()
#define LP lastPitch()
#define LV lastVibrAmt()
#define CR ctRef
#define VR vtRef
#define GA gAmp
#define NA nAmp

    setPerfPitch(c4);
Syn(0.05*CR,"kkk",LG,LP,0.00*GA,0.00*NA,LV,fd);
Syn(0.05*CR,"kkk",LG,LP,0.00*GA,0.00*NA,LV,fd);
Syn(0.02*CR,"kk+",LG,LP,0.00*GA,0.10*NA,LV,fd);
Syn(0.08*CR,"kk+",LG,LP,0.00*GA,0.00*NA,LV,fd);
Syn(0.10*VR,"alpha",LG,LP,0.80*GA,0.00*NA,LV,fd);
Syn(0.10*VR,"alpha",LG,LP,0.80*GA,0.00*NA,LV,fd);
Syn(0.40*VR,"alpha",LG,LP,0.80*GA,0.00*NA,LV,fd);
    setPerfPitch(bou1);
Syn(0.10*VR,"alpha",LG,LP,0.80*GA,0.00*NA,LV,fd);
Syn(0.10*VR,"alpha",LG,LP,0.80*GA,0.00*NA,LV,fd);
Syn(0.40*VR,"alpha",LG,LP,0.80*GA,0.00*NA,LV,fd);
Syn(0.05*CR,"lamda",LG,LP,0.80*GA,0.00*NA,LV,fd);
Syn(0.05*CR,"lamda",LG,LP,0.80*GA,0.00*NA,LV,fd);
Syn(0.10*VR,"iota",LG,LP,0.80*GA,0.00*NA,LV,fd);
Syn(0.10*VR,"iota",LG,LP,0.80*GA,0.00*NA,LV,fd);
    setPerfPitch(440.0);
Syn(0.10*VR,"iota",LG,LP,0.80*GA,0.00*NA,LV,fd);
Syn(0.10*VR,"iota",LG,LP,0.80*GA,0.00*NA,LV,fd);
Syn(0.20*VR,"iota",LG,LP,0.80*GA,0.00*NA,LV,fd);
Syn(0.05*CR,"mi",LG,LP,0.80*GA,0.00*NA,LV,fd);
Syn(0.15*CR,"mi",LG,LP,0.80*GA,0.00*NA,LV,fd);
Syn(0.10*VR,"epsilon",LG,LP,0.80*GA,0.00*NA,LV,fd)
Syn(0.10*VR,"epsilon",LG,LP,0.80*GA,0.00*NA,LV,fd);
    setPerfPitch(120.7*sin(1.5));
Syn(0.10*VR,"epsilon",LG,LP,0.80*GA,0.00*NA,LV,fd)
Syn(0.10*VR,"epsilon",LG,LP,0.80*GA,0.00*NA,LV,fd)
Syn(0.80*VR,"epsilon",LG,LP,0.80*GA,0.00*NA,LV,fd);
  rollr(0.15*CR,LG,LP,0.5,LV,fd);
Syn(0.10*VR,"alpha",LG,LP,0.80*GA,0.00*NA,LV,fd);
Syn(0.10*VR,"alpha",LG,LP,0.80*GA,0.00*NA,LV,fd);
Syn(0.60*VR,"alpha",LG,LP,0.80*GA,0.00*NA,LV,fd);
```

As can be seen, the Singer control file is a time-ordered list of vocal tract shapes, glottal input files, pitches, amplitudes etc. The vocal tract shapes and glottal input files reside in a library and have been generated by the SPASM synthesis program. The vocal tract shape files resemble as much as possible the Greek phonetic alphabet, and the glottal input files reflect different voice qualities and ranges. The Singer voice synthesizer drives smoothly from one shape and set of parameters to the next according to the control script, synthesizing the spoken/sung text.

4 Synthesis of Greek Singing

As for the Greek singing part of the IGDIS project, the main emphasis has been given to the scales, the various ornaments, and the prosodic features that are used both by Greek folk and Byzantine Ecclesiastical music [Karas, 1982]. Well tempered scales are also included. The scale in use must be declared in the beginning of the input text as an **#include** definition. Ornament routines have been added to the program and can be entered in the stream of the text, linearly as they are to be performed in time. These Singer ornament subroutines include both western-like ornaments that are similar to those used in Greek tradition, as well as ornaments which are unique to music of Greece and the surrounding areas.

The routine "melisma"- a Greek word for musical ornament, actually performs parallel control over a pitch and an amplitude envelope throughout the same time period. Because most of the Greek folk music ornaments deal a lot with parallel, but at the same time independent, linear control over pitch and amplitude, this routine can be used as a model and many other useful functions can be derived from it.

Here is a typical example of an input text making use of the routines "strepton1" (strepton) and "petasti1" (pt), tuned in "first mode" (firstMode):

```
#include "tuning/firstMode.h"

// The "strepton1" array is used to hold continuous
// gliss breakpoints, in the form of:
// (timeN,centsN) <--- Nth point
```

```
{   double strArray[4][2];
    double basis = a3;

    vtRef=0.6;     ctRef=0.6;

    strArray[0][0] = 0.11;
    strArray[0][1] = -120.0;
    strArray[1][0] = 0.12;
    strArray[1][1]= 120.0;
    strArray[2][0] = 0.12;
    strArray[2][1] = 200.0;
    strArray[3][0]= 0.12;
    strArray[3][1]= -200.0;

set_file_path("/Users/dimitris/LibrarySPASM/SPASM/");
    setGlot("Greekdefault");
    setPerfVibrAmt(0.001);
    setPerfRndAmt(0.005);
}

 <c4>ka-{strepton(4,strArray,fd);}-<bou1>---li<440.0>--
{pt(fd);}me<120.7*sin(1.5)>-----ra---
```

where "tuning/firstMode.h" is:

```
#include <math.h>

#define zv0a basis*pow(2.0,-20.0/72.0)
#define nh0 basis*pow(2.0,-12.0/72.0)
#define pa1 basis*pow(2.0,0.0/72.0)
#define bou1a basis*pow(2.0,10.0/72.0)
#define bou1d basis*pow(2.0,9.0/72.0)
#define ga1 basis*pow(2.0,18.0/72.0)
#define di1 basis*pow(2.0,30.0/72.0)
#define ke1 basis*pow(2.0,42.0/72.0)
#define zv1f basis*pow(2.0,48.0/72.0)
#define zv1a basis*pow(2.0,52.0/72.0)
#define zv1d basis*pow(2.0,51.0/72.0)
#define nh1 basis*pow(2.0,60.0/72.0)
#define pa2 pa1*2.0
#define bou2a bou1a*2.0
#define bou2d bou1d*2.0
#define ga2 ga1*2.0
#define di2 di1*2.0
#define ke2 ke1*2.0
#define zv2f zv1f*2.0
#define zv2a zv1a*2.0
#define zv2d zv1d*2.0
#define nh3 nh2*2.0
```

As can be seen in the above header file, there is often more than one definition for certain steps of the mode. This happens because of the characteristic of some steps to use a different pitch depending whether they are in ascending or descending movement, or represent stable or unstable tones in the particular melodic context [Merlier, 1935].

5 Conclusions and Future Directions

The IGDIS tools allow voice to be synthesized from input text. The Singer C code can be edited and fine tuned, and improvements fed back into the Singer subroutines. Work remaining to be done includes:

- Further development of the Greek phoneme library.
- Formalization of speech phenomena characterizing the transitional phonemes in modern Greek language.
- Expansion of the rules of parsing including more cases of modern Greek pronunciation.
- Construction of more complex Singer subroutines.
- Friendlier user interface for data entry.

6 Acknowledgements

The Greek Singer project was sponsored by the Greek Ministry of Culture and CCMR in Athens. Thanks to Thanassis Rikakis for providing the opportunity for the project, and his help with the project.

References

[Baud-Bovy 1984] Samuel Baud-Bovy, Etude sur la Chanson Populaire Greque, Nafplio 1984.

[Cook, 1989] Perry R. Cook, Synthesis of the Singing Voice Using a Physically Parametrized Model of the Human Vocal Tract, Proc. of the International Computer Music Conference, pp 69-72, Columbus OH, 1989.

[Cook, 1990] Perry R. Cook, SPASM : A Real-Time Vocal Tract Physical Model Editor/Controller and Singer: The Companion Software Synthesis System, Colloque les Modeles Physiques Dans L'Analyse, la Production et la Creation Sonore, ACROE, Grenoble, 1990, Published in Computer Music Journal 17(1), pp. 30-44, 1993.

[Cook, 1991a] Perry R. Cook. LECTOR: An Ecclesiastical Latin Control Language for the SPASM/Singer Instrument, Proc. of the International Computer Music Conference, pp 319-321, Montreal, 1991.

[Cook, 1991b] Perry R. Cook, Identification and Control of Parameters in an Articulatory Vocal Tract Model, With Applications to the Synthesis of Singing, Ph.D. dissertation, Dept of Electrical Engineering, Stanford Univ. 1991.

[Karas, 1988] Simonos I. Karas, Armonika, Proc. of Delphi Musicological Conference, Delphi 1988

[Karas, 1982] Simonos I. Karas, Methodos tis Ellinikis Mousikis-Theoritikon, Athens 1982.

[Merlier, 1935] Melpo Merlier Etudes de Musique Byzantine. Le Premier Mode et son Plagal, Athenes 1935.

A MIDI Control and Performance System for Brass Instruments

Perry Cook (prc@ccrma.stanford.edu) Stanford CCRMA
Dexter Morrill (dexter@cs.colgate.edu) Colgate University
Julius O. Smith (jos@ccrma.stanford.edu) Stanford CCRMA

Abstract

Pitch and frequency detection schemes which do not take into account the unique spectral and acoustical properties of a particular instrument family usually generate errors of three types: 1) octave and harmonic detection errors, 2) half step errors which jitter rapidly about the true estimate, and 3) latency of detection. A pitch detection and live MIDI control system has been constructed for the trumpet, which significantly reduces detection errors and latency. By limiting the search range to the natural playing range of the trumpet, sampling rate and computation can be optimized, reducing latency in the pitch estimates. By measuring and utilizing valve position in the pitch detection algorithm, the half-step jitter problem is completely eliminated, and latency can be further reduced. Schemes for reducing harmonic detection errors will be presented. The trumpet system is quantitatively compared to two popularly available pitch-to-MIDI systems. Performance features are integrated in the trumpet MIDI control system, such as MIDI file and sound file playback controlled by triggers mounted on the instrument itself.

1 Introduction: Pitch/Period Detection

Pitch detection is of interest whenever a single quasi-periodic sound source is to be studied or modeled, specifically in speech and music [Hess, 1983]. Pitch detection algorithms can be divided into methods which operate in the time-domain, frequency-domain, or both. One group of pitch detection methods relies on the detection of some set of features in the time-domain. Other time-domain methods use autocorrelation functions or difference norms to detect similarity between the waveform and a time lagged version of itself. Such methods are essentially period detectors, and not truly pitch (a perceptual measure) detectors. However, many signals such as those encountered in music are highly periodic, and thus systems which detect frequency or periodicity are often called pitch detectors. Another family of methods operates in the frequency-domain, locating sinusoidal peaks in the frequency transform of the input signal. Other methods use combinations of time and frequency-domain techniques to detect pitch.

Frequency-domain methods call for the signal to be frequency-transformed, then the frequency-domain representation is inspected for the first harmonic, the greatest common divisor of all harmonics, or other such indications of the period. Windowing of the signal is recommended to smooth the effects at frame edges, and a minimum number of periods of the signal must be analyzed to enable accurate location of harmonic peaks. Various linear pre-processing steps can be be used to make the process of locating frequency-domain features easier, such as performing linear prediction on the signal and using the residual signal for pitch detection. Performing non-linear operations such as peak limiting also simplifies the location of harmonics.

In a time-domain feature detection method the signal is usually pre-processed to accentuate some time-domain feature, then the time between occurrences of that feature is calculated as the period of the signal.

A typical time-domain feature detector might low-pass filter the signal, then detect peaks or zero crossings. Since the time between occurrences of a particular feature is used as the period estimate, feature detection schemes usually do not use all of the data available. Selection of a different feature yields a different set of pitch estimates [Deem *et. al.*, 1989]. Since estimates of the period are often defined at the instant when features are detected, the frequency samples yielded are non-uniform in time. To avoid the problem of non-uniform time sampling, a window of fixed size is moved through the signal, and a number of detected periods within each window are averaged to obtain the period estimate. For reliable and smooth estimation, the window must be at least a few periods long. Often the signal must be interpolated between samples in order to locate the feature occurrence time as accurately as needed.

2 A New Period Estimation Algorithm

A method of pitch detection which uses the phase delay of a periodic predictor to form the pitch estimate will be briefly presented in this section. For more detail on the algorithm, implementation, and applications, see [Cook, 1991]. This pitch detector accurately tracks a quasi-periodic signal, and will be called the Periodic Predictor Pitch Tracker (PPPT). The PPPT provides a method of automatically and adaptively determining the optimum continuous-time period, and also provides an estimate of the reliability of the period estimate. The PPPT system as initially described is not a complete pitch detector, in that it relies on some other scheme for an initial estimate of the period. Once the detector locks onto the correct period, the method provides accurate estimates of the instantaneous period using all samples of the input signal, provides an estimate of the periodicity of the signal, and provides controls which affect the dynamics and accuracy of the pitch detector.
Given a quasi-periodic signal x(n), and an integer estimate P of the initial period, periodic prediction is implemented by:

$$\hat{x}(k) = \sum_{i=-M}^{M} x(k-P+i)\, c(i) \qquad [1]$$

where M is some appropriately chosen small number

and c(l) are the predictor coefficients. Backward prediction is implemented by replacing P with -P in Equation 1. Figure 1 shows a block diagram of an FIR periodic predictor.

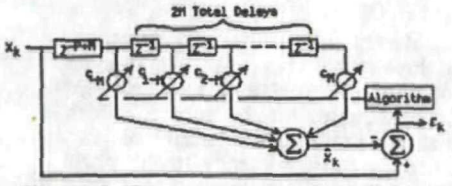


Figure 1: Linear FIR period predictor

The phase (relative to the Pth delayed sample) of the FIR filter implemented by the predictor coefficients is computed by:

$$\theta = \tan^{-1} \frac{\sum_{l=-M}^{M} (c(l) - c(-l)) * \sin(\omega l)}{c(0) + \sum_{l=-M}^{M} (c(l) + c(-l)) * \cos(\omega l)} \quad [2]$$

The frequency ω is the frequency at which the phase delay of the filter is calculated. The relation between the pitch estimate and ω is:

$$\omega = 2\pi / T_0 = 2\pi / T_s \quad [3]$$

where T_s is the sampling period in seconds and T_0 is the period estimate.

For further computational savings, sine, cosine, and $\tan^{-1}$ values can be calculated by interpolated table lookup. The phase delay of the filter is computed by:

$$\text{Phase Delay} = \theta / \omega \quad [4]$$

By adding the computed time delay to the time delay of the P length delay line, the net time delay of the predictor is computed. This total delay is then used to compute a period and frequency estimate:

$$\text{Period} = T_0 = \text{Phase Delay} + (P / \text{Sampling Rate}) \quad [5]$$

$$\text{Frequency} = F_0 = 1 / T_0 \quad [6]$$

2.1 FIR Periodic Prediction Algorithms

There are many known methods of implementing the adaptive FIR predictor used in the PPPT, among them the Covariance Least Squares (CLS), Recursive Least Squares Adaptive (RLS), and Least Mean Squares Adaptive (LMS) algorithms. All of these methods minimize the Mean Square Error (MSE):

$$MSE = 1 / N \sum_{i=M}^{M} \varepsilon_i^2 \quad [7]$$

where the instantaneous error ε_k is defined as the difference between the signal sample and the predicted sample at time k:

$$\varepsilon_k = x(k) - x\hat{}(k) \quad [8]$$

The Least Mean Squares (LMS) adaptive [Widrow and Stearns, 1985] algorithm is a gradient steepest descent algorithm using the instantaneous error to estimate the gradient of the error surface. It is preferred by the authors because of its performance and efficiency of implementation. Each predictor coefficient is adjusted at each time sample by an amount proportional to the instantaneous error and the signal value which is associated with the coefficient being adjusted. The LMS update equation is:

$$C_{k+1} = C_k + 2\mu X_k \varepsilon_k \quad [9]$$

where C_k is the vector of predictor coefficients, and X_k is the corresponding delayed input sample vector. The adaptation constant 2μ comes from the Newton's method derivation (the 2 comes from taking the multi-dimensional derivative of the error function), and controls the dynamics (and stability) of adaptation. Stability is ensured if:

$$\mu < (2M+1)^{-1} \overline{x^2}^{-1} \quad [10]$$

where $\overline{x^2}$ is the signal power. The adaptation parameter μ can be adapted dynamically, yielding the Normalized LMS algorithm:

$$C_{k+1} = C_k + \alpha X_k \varepsilon_k / (2(M+1)\overline{x^2}) \quad [11]$$

where the signal power is computed over at least the last 2M+1 samples. α is any positive number < 1.

2.2 Adapting the Delay Parameter P

The integer period estimate P is variable, and there are new issues of filter dynamics in the LMS and RLS systems caused by on-line adaptation of the delay-line length. Ideally, the filter should experience no transients because of the adaptive modification of P. Various methods have been developed for adapting P, and are described in detail in [Cook, 1991].

2.3 The AMDF

The PPPT can be viewed as a refinement of the Average Magnitude Difference Function (AMDF) detectors. Methods of this type have also been called comb-filter methods. The AMDF measures the difference between the waveform and a lagged version of itself. The generalized AMDF is:

$$AMDF(P) = \sum_{i=q}^{q+N-1} |x(i) - x(i+P)|^m \quad [12]$$

The quantity m is set to 1 for average magnitude difference, and other values for other related methods. The zero lag P=0 position of the AMDF is identically zero, and the next significant null is a likely estimate of the period. Other nulls will occur at integer multiples of the period. If the error signal from the optimally adjusted PPPT were rectified and integrated, the output would be close to that of the optimum lag bin of the AMDF. The difference is that the PPPT can adjust to fractional sample periods, can even adjust to signals which are not purely harmonic (periodic), and does not require the block processing that the AMDF imposes.

3 A System for Brass Instrument MIDI Control

A MIDI (Musical Instrument Digital Interface)

performance control system has been constructed which integrates a valve-guided AMDF fundamental pitch estimator for coarse estimates, one PPPT adaptive period predictor for fine estimates, and other controls for musical performance. Figure 2 shows the system control screen, which runs on a NeXT computer.

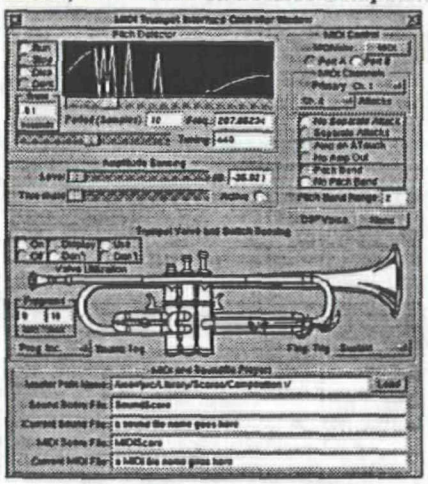


Figure 2: NeXT control window for Trumpet MIDI controller program

The trumpet audio is sensed using a mouthpiece-mounted microphone, routed to the CODEC port on the NeXT, and sampled at 8012 Hz. The valve states are sensed using two optical switches per valve, resulting in four position accuracy for each valve. Two additional switches mounted on the trumpet can be actuated by the player's finger and thumb, providing programmable control of performance functions such as MIDI and sound file playback, and synthesizer controls such as program change and sustain. The eight valve and control switches are encoded into a single serial byte and routed to the NeXT machine via one of the serial ports running at 19,200 baud. There is one internal synthesizer voice provided (an FM trumpet) synthesized on the NeXT DSP 56001 digital signal processing chip, primarily for tuning the control instrument and pitch detectors. A standard MIDI interface connects to the other NeXT serial port, and can be used to connect the system to any MIDI compatible synthesizer.

Modes are provided to incorporate valve information into the AMDF fundamental pitch estimate, by limiting the AMDF calculations to 'legal' harmonics based on the current valve state. A single PPPT adaptive periodic predictor is set to the integer period yielded by the AMDF detector, and the coefficients of the PPPT are used to calculate fine estimates of fundamental pitch. Pitch bend can be used to continuously update the MIDI synthesizer based on the fine pitch estimate of the PPPT.

The system was tested and compared to two popular 'pitch to MIDI' devices available in the music market. The two pitch to MIDI detectors will be called OTHER-A (circa 1989, list price about $900) and OTHER-B (circa 1991, list price about $300). Our device will be called OURS (list price about $7000, because of custom-machined hardware for the trumpet, and the host NeXT machine). The three systems were tested for latency, defined as the total time delay from first trumpet sound to first synthesizer sound. All three devices were presented with the same audio and controlled the same synthesizer voice. The test passage consisted of a chromatic scale up and down, separately articulated (tongued). The systems were also tested for accuracy, defined by the number of erroneous MIDI messages sent during the articulated chromatic test passage, and another chromatic passage which was performed slurred (notes connected with no space between and no rearticulation). Two of each passage were presented, and the best performance of each system was used to calculate accuracy and average latency. Figures 3 and 4 show the results of the three detectors on the two chromatic test passages. The graphs of the three device outputs are offset vertically by 10 for plotting clarity. Errors are circled.

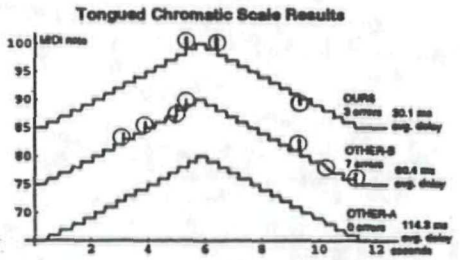


Figure 3: Detection results for tongued chromatic scale.

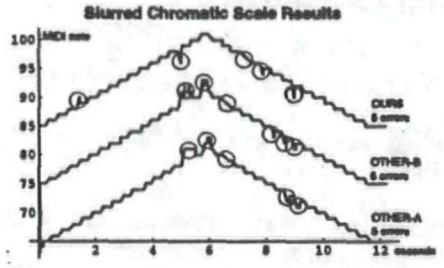


Figure 4: Detection results for slurred chromatic scale.

OTHER-A exhibited an average latency of 114.3 ms. with a standard deviation of 66 ms., zero errors in the articulated chromatic passage, and five errors in the non-articulated chromatic passage. OTHER-B exhibited an average latency of 60.4 ms. with a standard deviation of 21.7 ms., seven errors in the articulated chromatic passage, and six errors in the non-articulated chromatic passage. OURS exhibited an average latency of 30.1 ms. with a standard deviation of 8.1 ms., three errors in the articulated chromatic passage, and five errors in the non-articulated chromatic passage. These results point out a general tradeoff between accuracy and latency, where 'waiting around' pays off in increased confidence in the estimate. The OTHER-A device exhibited the highest accuracy, but at the price of objectionably high latency. Most of the errors in OURS were caused by the valves moving slightly before the pitch changes, or intermediate valve positions between two stable positions (in moving from all valves up to all valves down for example, two valves may arrive earlier

than the third), thus causing a spurious message to be sent. It is conjectured that performance can be improved by delaying the valve information slightly and making more use of fractional valve positions to determine valve trajectories.

One further test was performed which integrates both aspects of latency and accuracy, based on the system's ability to accurately track a valve trill which increases in speed. As the trill speed increases, at some point each device should fail based on its inability to reliably detect pitch and send an output message. Of course, with valves active, the theoretical limit for a system using valve information to track a valve trill is 1/2 the valve sampling rate (1000 Hz in the OURS case). Figure 5 shows the results of a half step trill (all valves up alternating with valve two down). The OTHER-A device fails at a trill rate of about 7 Hz. (consistent with the 114.4 ms. average latency). The OTHER-B device began to produce errors at around 6 Hz., but did not fail to track changes in pitch at the maximum trill rate of 8 Hz. The OURS system produced no errors and tracked the trill up to the maximum rate of 8 Hz.

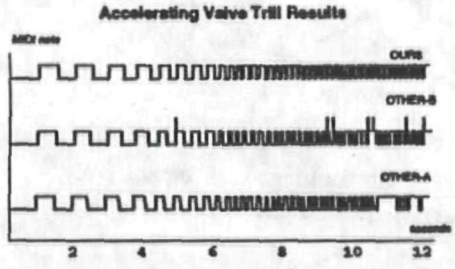


Figure 5: Detection results for accelerating valve trill.

4 Composing Music for Trumpet Performers Using Pitch Detectors

As a single voice, sustaining instrument with a full potential for a fast, somewhat percussive attack, the trumpet requires a reasonably fast and accurate pitch detector in computer music systems. The trumpet often plays long notes because of its rich melodic potential, and the amplitude contour may vary considerably. Both the steady state amplitude variation and small variations in pitch, including a periodic vibrato, make it essential for the system to constantly "follow" the sound. Players can feel the response of the computer music system and find it difficult to play naturally if the pitch detection/MIDI delay is greater than .035 seconds. For the performer, the most annoying problems are (1) missed pitch, especially in fast passages, (2) MIDI NOTE ON delays of more than .035 seconds, (3) false MIDI NOTE ON messages caused by valve noise or small breath impulses, and (4) a lack of pitch bend (small pitch variations) and aftertouch (amplitude variations) causing the synthesis to 'untrack' from the brass sound source on long notes.

From the standpoint of composition, the matter of pitch detection accuracy is quite simple. If the computer music system is not very accurate (i.e., near 100%), then the composer must find a way to alleviate the problem for the performer. In conventional music, especially most music found in the western music repertoire, pitch accuracy is perhaps of the highest order of importance, and performers can sometimes quickly correct mistakes. Fast scale passages are particularly well known and easily followed by listeners, because of their common usage and high degree of predictability. Composers can compensate for slow pitch detection in computer music systems by not allowing the listener to know what the 'right' pitch is, or have a low expectation for pitches. In situations where the pitch is missed by the player, some tolerance is demanded by the composition. Improvisation is ideal, allowing the player some freedom in the choice and rhythmic placement of notes. It is best to avoid conventional arpeggios or scales, and an obvious melodic doubling of parts played by the music system.

To some extent, the performer using these systems may enter the domain of accompaniment, where a rich texture or mass of sound results. Most players want to expand the limits of their own playing techniques, not to reduce the role of their playing, or to have a machine do most of the work for them. For this reason, there has been an odd circumstance where composers and performers sometimes have very different goals; the player reaching for a fantastic new technique and the composer attempting to find a new sound and musical expression, leaving the player with new, but unrewarding tasks.

The concept of our design is to leave the natural trumpet alone and offer the player an extension to his or her instrument, reaching beyond to the world of synthesizers and processors with hand and embouchure control that is already learned. It is unrealistic to think that this extension will be without some cost or disadvantages, or that it will not require the player to learn some new techniques. Wind players are very sensitive to tiny changes in mouthpiece size, for example, so it is reasonable to expect that such a formidable expansion in the domain of sound will require adjustments and some time for exploitation.

References

[Cook, 1991] Perry R. Cook, Identification and Control of Parameters in an Articulatory Vocal Tract Model, With Applications to the Synthesis of Singing, Ph.D. dissertation, Dept of Electrical Engineering, Stanford Univ. 1991

[Deem *et al*, 1989] J. F. Deem, W. H. Manning, J. V. Knack and J. S. Matesich, The Automatic Extraction of Pitch Perturbation Using Microcomputers: Some Methodological Considerations, Journal of Speech and Hearing Research, vol. 32, pp. 689-697, 1989.

[Hess, 1983] Wolfgang Hess, *Pitch Determination of Speech Signals*, Berlin: Springer Verlag, 1983.

[Widrow and Sterns, 1985] Bernard Widrow and S. D. Stearns, *Adaptive Signal Processing*, New Jersey: Prentice-Hall, 1985.

Efficient Synthesis of Stringed Musical Instruments

Julius O. Smith III
jos@ccrma.stanford.edu

Center for Computer Research in Music and Acoustics (CCRMA)
Stanford University Stanford, CA 94305

Abstract

Techniques are described for reducing complexity in stringed instrument simulation for purposes of digital synthesis. These include commuting losses and dispersion to consolidate them into a single filter, replacing body resonators by look-up tables, simplified bow-string interaction, and single-filter, multiply-free coupled strings implementation.

1 Digital Waveguide Theory

This section summarizes the *digital waveguide* model for vibrating strings. Further details can be found in [Smith, 1992].

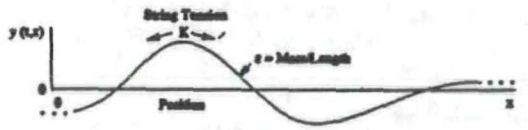


Figure 1: The ideal vibrating string.

The *wave equation* for the ideal (lossless, linear, flexible) vibrating string, depicted in Fig. 1, is given by

$$Ky'' = \epsilon\ddot{y}$$

where

K	$\triangleq$	string tension	y	$\triangleq$	$y(t,x)$
ϵ	$\triangleq$	linear mass density	$\dot{y}$	$\triangleq$	$\frac{\partial}{\partial t}y(t,x)$
y	$\triangleq$	string displacement	y'	$\triangleq$	$\frac{\partial}{\partial x}y(t,x)$

The same wave equation applies to any displacement along one dimension in any perfectly elastic medium. We refer to the general class of such media as *one-dimensional waveguides*. Extension to two and more dimensions is described elsewhere in this proceedings [Van Duyne and Smith, 1993].

It can be readily checked that the wave equation is solved by any string shape which travels to the left or right with speed $c = \sqrt{K/\epsilon}$. (But note that the derivation of the wave equation assumes the string slope is much less than 1 at all times and positions.) If we denote right-going traveling waves by $y_r(x-ct)$ and left-going traveling waves by $y_l(x+ct)$, where y_r and y_l are arbitrary twice-differentiable functions, then the general class of solutions to the lossless, one-dimensional, second-order wave equation can be expressed as

$$y(x,t) = y_r(x-ct) + y_l(x+ct)$$

Sampling the traveling-waves gives

$$\begin{aligned} y(t_n, x_m) &= y_r(t_n - x_m/c) + y_l(t_n + x_m/c) \\ &= y_r(nT - mX/c) + y_l(nT + mX/c) \\ &= y_r[(n-m)T] + y_l[(n+m)T] \end{aligned}$$

Since T multiplies all arguments, we suppress it by defining

$$y^+(n) \triangleq y_r(nT) \qquad y^-(n) \triangleq y_l(nT)$$

The "+" superscript denotes a traveling-wave component propagating to the right, and "−" denotes propagation to the left. Finally, the left- and right-going traveling waves must be summed to produce a physical output according to the formula

$$y(t_n, x_m) = y^+(n-m) + y^-(n+m)$$

The appendix shows a linear wave equation with constant coefficients, of any order, admits a decaying, dispersive, traveling-wave solution. Even-order time derivatives give rise to dispersion and odd-order time derivatives correspond to losses; higher order spatial derivatives give rise to multiple solutions of the same kind. The corresponding digital simulation of an arbitrarily long (undriven and unobserved) section of medium can be simplified via commutativity to at most two pure delays and at most two linear, time-invariant filters. In dimensions higher than one, these remarks apply to any given direction of traveling-wave propagation.

Since every linear, time-invariant filter can be expressed as a zero-phase filter in cascade with an

allpass filter, we may factor the filter into its lossy part and its dispersive part. The zero-phase factor implements frequency-dependent gain (damping in a digital waveguide), and the allpass part gives frequency-dependent delay, (dispersion in a digital waveguide). A digital simulation diagram appears in Fig. 2.

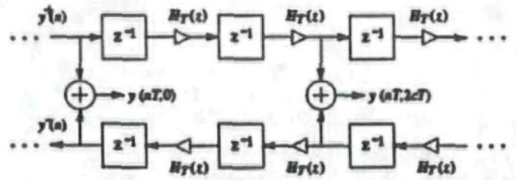


Figure 2: Discrete simulation of the ideal, linear, lossy, dispersive, digital waveguide. Between each delay element is an arbitrary linear filter $H_T(z)$ whose amplitude response implements frequency-dependent *attenuation* and whose phase response implements frequency-dependent *dispersion* in one sampling period.

The simulation of the traveling-waves is exact, in principle, at the sampling positions and instants, even though losses and dispersion are admitted in the wave equation. Note also that the losses which are *distributed* in the continuous solution have been consolidated, or *lumped*, at discrete intervals of cT meters in the simulation. The filter $H_T(z)$ *summarizes* the distributed filtering incurred in one sampling interval. The lumping of distributed filtering does not introduce an approximation error at the sampling points. Furthermore, bandlimited interpolation can yield arbitrarily accurate reconstruction between samples [Smith and Gossett, 1984]. The main restriction is that all initial conditions and excitations be bandlimited to half the sampling rate.

It is usually possible to realize vast computational savings in waveguide simulation by *commuting losses out of unobserved and undriven sections of the medium and consolidating them at a minimum number of points*. Because the digital simulation is linear and time invariant (given constant medium parameters), and because linear, time-invariant elements commute, the diagram in Fig. 3 is exactly equivalent (to within numerical precision) to the previous diagram in Fig. 2. Each per-sample filter $H_T(z)$ is commuted with delay elements and combined with other filters until an input or output is encountered which inhibits further migration. Filters can also be pushed through nodes in the diagram to achieve further simplifications in some cases.

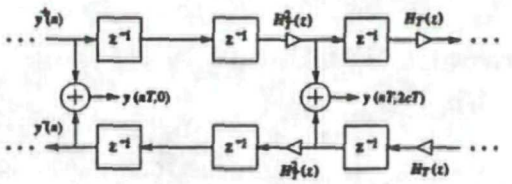


Figure 3: General linear digital waveguide with commuted loss/dispersion filters.

2 The Terminated String

Using the above simplification principles, it is possible to commute the elements of a rigidly terminated, dispersive, lossy string into the form shown in Fig. 4, provided that the string is to be excited by initial conditions and the output signal is taken to be a traveling-wave component. In this case, the losses and dispersion are lumped at a single point in the round-trip travel along the string. When the loop filter is a two-point average $(1+z^{-1})/2$, and when the initial conditions used to "pluck" the string are taken to be random numbers, the well known Karplus-Strong algorithm for string and drum sounds is obtained [Karplus and Strong, 1983; Jaffe and Smith, 1983].

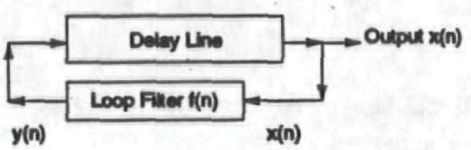


Figure 4: The rigidly terminated, linear string.

In a general physical model, the loop filter is determined by the cascade of (1) the filtering experienced by a traveling wave in traversing the string twice, and (2) the reflection transfer functions of the two terminations.

If the wave impedance of the string is $R = \sqrt{K\epsilon}$, and the bridge driving-point impedance is $R_b(z)$, then the reflection transfer function at the bridge is given by

$$S_b(z) = \frac{F^-(z)}{F^+(z)} = \frac{R_b(z) - R}{R_b(z) + R}$$

for force waves, and $-S_b(z)$ for velocity waves. Because the bridge is passive, $R_b(z)$ is *positive real*, [Van Valkenburg, 1960], i.e.,

(1) $R_b(z)$ is real when z is real.

(2) $|z| \geq 1 \Rightarrow \mathrm{re}\{R_b(z)\} \geq 0$.

This implies $S_b(z)$ is a *Schur function*, i.e., $S_b(z) \leq 1$ for $|z| \geq 1$. Reflection filters associated with passive, finite-order impedances always have an equal number of poles and zeros, as can be seen

from the above expression. If the bridge termination is lossless, its impedance $R_b(z)$ is purely reactive and the reflection filter $S_b(z)$ becomes allpass. Typically, the reflection filter has gain less than but close to 1 at all frequencies, and the gain is smallest at frequencies where there is strong coupling with a bridge or body resonance.

3 Simplified Body Filters

In a complete stringed musical instrument, such as a guitar, the string couples via the bridge into a resonating "body" which is needed for coupling to the surrounding air, and which imposes a frequency response of its own on the radiated sound. In addition, spectral characteristics of the string excitation affect the radiated sound. Thus, we have the components shown in Fig. 5.

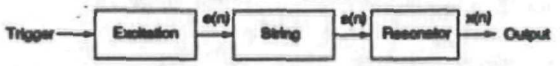


Figure 5: Schematic diagram of a stringed musical instrument.

Because the string and body are approximately linear and time-invariant, we may *commute the string and resonator*, as shown in Fig. 6.

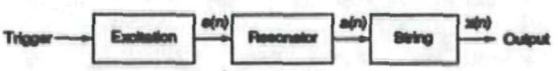


Figure 6: Equivalent diagram in the linear, time-invariant case.

The excitation can now be *convolved* with the resonator impulse response to provide a single, aggregate, excitation table, as depicted in Fig. 7.

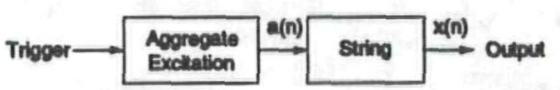


Figure 7: Use of an aggregate excitation given by the convolution of original excitation with the resonator impulse response.

In the simplest case, the string is "plucked" using the (half-windowed) impulse response of the body.

An example of an excitation is the force applied by a pick or a finger at some point, or set of points, along the string. The input force per sample at each point divided by $4R$ gives the velocity to inject additively at that point in both traveling-wave directions. (The factor of 4 comes from splitting the injected velocity into two traveling-wave components, and from the fact that two string end-points are being driven.) Equal injection in the left- and right-going directions corresponds to a excitation force which is stationary with respect to the string.

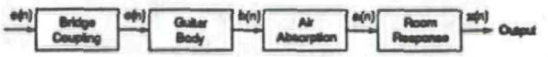


Figure 8: Possible components of a guitar resonator.

In a practical instrument, the "resonator" is determined by the choice of output signal in the physical scenario, and it generally includes filtering *downstream of the body itself*, as shown in Fig. 8. A typical example for the guitar or violin would be to choose the output signal at a point a few feet away from the top plate of the body. In practice, such a signal can be measured using a microphone held at the desired output point and recording the response at that point to the striking of the bridge with a force hammer. It is useful to record simultaneously the output of an accelerometer mounted on the bridge in order to also obtain experimentally the driving-point impedance at the bridge. In general, it is desirable to choose the output close to the instrument so as to keep the resonator response as short as possible. The resonator components need to be linear and time invariant, so they will be commutative with the string and combinable with the string excitation signal via convolution.

The string should also be linear and time invariant in order to be able to commute it with the generalized resonator. However, the string is actually the least linear element of most stringed musical instruments, with the main effect of nonlinearity being a slight increase of the fundamental vibration frequency with amplitude. A secondary effect is to introduce coupling between the two polarizations of vibration along the length of the string. In practice, however, the string can be considered sufficiently close to linear to permit commuting with the body. The string is also time varying in the presence of vibrato, but this too can be neglected in practice. While commuting a live string and resonator may not identical mathematically, the sound is substantially the same.

There are various options when combining the excitation and resonator into an aggregate excitation, as shown in Fig. 7. For example, a wavetable can be prepared which contains the convolution of a particular point excitation with a particular choice of resonator. Perhaps the simplest choice of excitation is impulse signal. Physically, this would be natural when the wave variables in the string are taken to be acceleration waves for a plucked string; in this case, an ideal pluck gives

rise to an impulse of acceleration input to the left and right in the string at the pluck point. If loss of perceived pick position is unimportant, the impulse injection need only be in a single direction. (The comb filtering which gives rise to the pick-position illusion can be restored by injecting a second, negated impulse at a delay equal to the travel time to and from the bridge.) In this simple case of a single impulse to pluck the string, the aggregate excitation is simply the impulse *response* of the resonator. Many excitation and resonator variations can be simulated using a collection of aggregate excitation tables. It is useful to provide for interpolation of excitation tables so as to provide intermediate points along a parameter dimension. In fact, all the issues normally associated with sampling synthesis arise in the context of the string excitation table. A disadvantage of combining excitation and resonator is the loss of multiple output signals from the body simulation, but the timbral effects arising from the mixing together of multiple body outputs can be obtained via a mixing of corresponding excitation tables.

If the aggregate excitation is too long, it may be shortened by a variety of techniques. It is good to first convert the final excitation $a(n)$ in Fig. 7 to *minimum phase* so as to provide the maximum shortening consistent with the original magnitude spectrum. Secondly, $a(n)$ can be "half-windowed" using the right wing any window function typically used in spectrum analysis. An interesting choice is the exponential window, since it has the interpretation of increasing the resonator damping in a uniform manner, i.e., all the poles and zeros of the resonator are contracted radially in the z plane by the same factor.

4 Simplified Bowed Strings

The method of the previous section can be extended to bowed strings in an efficient way. The "leaning sawtooth" waveforms observed by Helmholtz for steady state bowed strings can be obtained by periodically "plucking" the string in only one direction along the string. In principle, a traveling impulsive excitation is introduced into the string in the right-going direction each period for a "down bow" and in the left-going direction for an "up bow." This simplified bowing simulation works best for smooth bowing styles in which the notes have slow attacks. More varied types of attack can be achieved using the more physically accurate McIntyre-Woodhouse theory [Smith, 1987].

Commuting the string and resonator means that the string is now plucked by a *periodically repeated resonator impulse response*. A nice simplified vibrato implementation is available by varying the impulse-response retriggering period, i.e., the vibrato is implemented in the excitation oscillator and not in the delay loop. The string loop delay need not be modulated at all. While this departs from being a physical model, the vibrato quality is satisfying and qualitatively similar to that obtained by a rigorous physical model. Figure 9 illustrates the overall block diagram of the simplified bowed string and its commuted and response-excited versions.

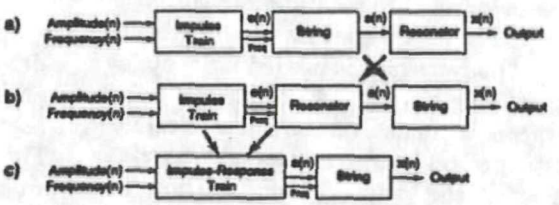


Figure 9: a) The simplified bowed string, including amplitude, pitch, and vibrato controls. The frequency control is also used by the string. b) Equivalent diagram with resonator and string commuted. c) Equivalent diagram in which the resonator impulse response is played into the string each pitch period.

In current technology, it is reasonable to store one recording of the resonator impulse response in digital memory as one of many possible *string excitation tables*. The excitation can contribute to many aspects of the tone to be synthesized, such as whether it is a violin or a cello, the force of the bow, and where the bow is playing on the string. Also, graphical equalization and other time-invariant filtering can be provided in the form of alternate excitation-table choices.

During the synthesis of a single bowed-string tone, the excitation signal is played into the string quasi-periodically. Since the excitation signal is typically longer than one period of the tone, it is necessary to either (1) interrupt the excitation playback to replay it from the beginning, or (2) start a new playback which overlaps with the playback in progress. Variant (2) requires a separate incrementing pointer and addition for each instance of the excitation playback; thus it is more expensive, but it is preferred from a quality standpoint.

Of course, ordinary wavetable synthesis or any other type of synthesis can also be used as an excitation signal in which case the string loop behaves as a pitch-synchronous comb filter following the wavetable oscillator. Interesting effects can be obtained by slightly detuning the wavetable oscillator and delay loop; tuning the wavetable oscillator to a harmonic of the delay loop can also produce an ethereal effect.

The externally excited, filtered delay loop can be used also to simulate wind and other musical

instruments. In fact, *any quasi-periodic tone* can be approximated using an appropriate excitation signal (which may be varied over time) together with some loop filter (which also may be varied over time). The fact that the delay line is approximately one period in length restricts application of this type of structure to quasi-periodic tones. However, aperiodic tones which can be well approximated by a superposition of a few quasi-periodic tones can be synthesized using multiple delay loops added together in parallel and excited by common or separate excitations. Thus, piano, marimba, and glockenspiel can be approximated, for example. For wind instruments, a filtered, enveloped noise excitation is needed. In summary, the externally excited, filtered delay loop can be viewed as an efficient compression technique for arbitrary quasi-periodic signals with musically desirable parameters.

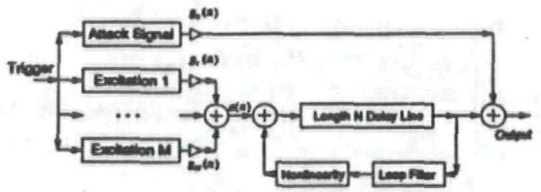


Figure 10: Generalized filtered delay loop synthesis.

Figure 10 illustrates a more general version of the table-excited, filtered delay loop synthesis system. The generalizations help to obtain a wider class of timbres. The multiple excitations summed together through time-varying gains provide for timbral evolution of the tone. For example, a violin can transform smoothly into a cello, or the bow can move smoothly toward the bridge by interpolating among two or more tables. Alternatively, the tables may contain "principal components" which can be scaled and added together to approximate a wider variety of excitation timbres. An excellent review of multiple wavetable synthesis appears in [Horner et al., 1993]. The nonlinearity is useful for obtaining distortion guitar sounds and other interesting evolving timbres.

Finally, the "attack signal" path around the string has been found to be useful for reducing the cost of implementation: the highest frequency components of a struck string, say, tend to emanate immediately from the string to the resonator with very little reflection back into the string (or pipe, in the case of wind instrument simulation). Injecting them into the delay loop increases the burden on the loop filter to quickly filter them out. Bypassing the delay loop altogether alleviates requirements on the loop filter and even allows the filtered delay loop to operate at a lower sampling rate; in this case, a signal interpolator would appear between the string output and the summer which adds in the scaled attack signal in Fig. 10. For example, it was found that the low E of an electric guitar (Gibson Les Paul) can be synthesized quite well using a filtered delay loop running at a sampling rate of 3 kHz. (The pickups do not pick up much energy above 1.5 kHz.) Similar savings can be obtained for any instrument having a high-frequency content which decays much more quickly than its low-frequency content.

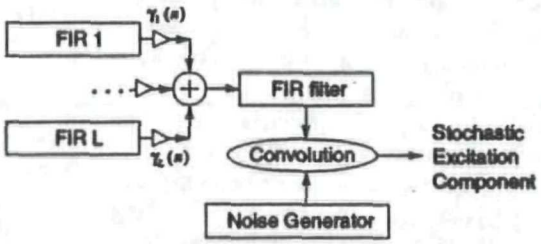


Figure 11: Example of a filtered noise excitation implementation.

For good generality, at least one of the excitation signals should be a filtered noise signal. An example implementation is shown in Fig. 11. In this example, there is a free running bandlimited noise generator which is filtered by a finite impulse response (FIR) digital filter. The filter coefficients are computed in real time as a linear combination of a set of fixed FIR coefficient sets stored in ROM. A recursive filter may also be used, in which case ladder/lattice forms can be used so that the coefficients can be interpolated without stability problems. In a simple implementation, only two gains might be used, allowing simple interpolation from one filter to the next, and providing an overall amplitude control for the noise component of the excitation signal.

5 Coupled Strings

In stringed musical instruments, coupling phenomena cannot be ignored. Coupling effects include amplitude modulation of partial amplitude envelopes due to "beating" between two or more coupled modes, two-stage decay (a fast decay followed by a slower decay), or "aftersound" [Weinreich, 1979]. Physically, significant coupled-string phenomena result from inter-string coupling, coupling between the horizontal and vertical polarizations of vibration on one string, and between the string and body resonances.

The simplest simulation of coupled strings is obtained by simply summing two or more slightly detuned strings. More realistic string coupling involves actual signal flow from each coupled string to all others.

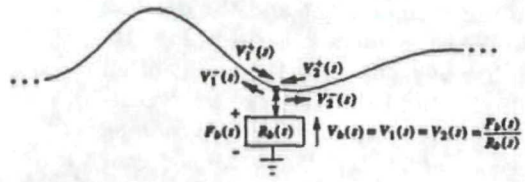


Figure 12: Two strings terminated at a common bridge impedance.

A diagram for the two-string case is shown in Fig. 12. This situation is a special case of the *loaded waveguide junction* [Smith, 1987], with the number of waveguides being $N = 2$, and the junction load being the transverse driving-point impedance $R_b(s)$ where the string drives the bridge. For a direct derivation, we need only observe that (1) the string velocities of each string endpoint must each be equal to the velocity of the bridge, or $v_1 = v_2 = v_b$, and (2) the sum of forces of both strings equals the force applied to the bridge: $f_b = f_1 + f_2$. The bridge impedance relates the force and velocity of the bridge via $F_b(s) = R_b(s)V_b(s)$. Expanding into traveling wave components in the Laplace domain, we have

$$\begin{aligned} R_b(s)V_b(s) &= F_b(s) = F_1(s) + F_2(s) \\ &= [F_1^+(s) + F_1^-(s)] + [F_2^+(s) + F_2^-(s)] \\ &= R_1\{V_1^+(s) - [V_b(s) - V_1^+(s)]\} \\ &\quad + R_2\{V_2^+(s) - [V_b(s) - V_2^+(s)]\} \end{aligned}$$

or

$$V_b(s) = H_b(s)[R_1V_1^+(s) + R_2V_2^+(s)]$$

where R_i is the wave impedance of string i, and

$$H_b(s) \triangleq \frac{2}{R_b(s) + R_1 + R_2}$$

Thus, in the time domain, the incoming velocity waves are scaled by their respective wave impedances, summed together, and filtered according to the transfer function $H_b(s) = 2/[R_b(s) + R_1 + R_2]$ to obtain the velocity of the bridge $v_b(t)$.

Given the filter output $v_b(t)$, the outgoing traveling velocity waves are given by

$$\begin{aligned} v_1^-(t) &= v_b(t) - v_1^+(t) \\ v_2^-(t) &= v_b(t) - v_2^+(t) \end{aligned}$$

Thus, the incoming waves are subtracted from the bridge velocity to get the outgoing waves.

Since $V_2^-(s) = H_b(s)R_1V_1^+(s) = H_b(s)F_1^+(s)$ when $V_2^+(s) = 0$, and vice versa exchanging strings 1 and 2, H_b may be interpreted as the *transmission admittance filter* associated with the bridge coupling. It can also be interpreted as the bridge admittance transfer function from every string, since its output is the bridge velocity resulting from the sum of incident traveling force waves.

A general coupling matrix contains a filter transfer function in each entry of the matrix. For N strings, each conveying a single type of wave (e.g., horizontally polarized), the general linear coupling matrix would have N^2 transfer-function entries. In the present formulation, only one transmission filter is needed, and it is shared by all the strings meeting at the bridge.

The above sequence of operations is formally similar to the *one multiply scattering junction* frequently used in digital lattice filters [Markle and Gray, 1976]. In this context, it would be better termed the "one-filter scattering termination."

When the two strings are identical (as would be appropriate in a model for coupled piano strings), the computation of bridge velocity simplifies to

$$V_b(s) = H_b(s)[V_1^+(s) + V_2^+(s)]$$

where $H_b(s) \triangleq 2/[2 + R_b(s)/R]$ is the *velocity* transmission filter. In this case, the incoming velocities are simply summed and fed to the transmission filter which produces the bridge velocity at its output. A commuted simulation diagram appears in Fig. 13.

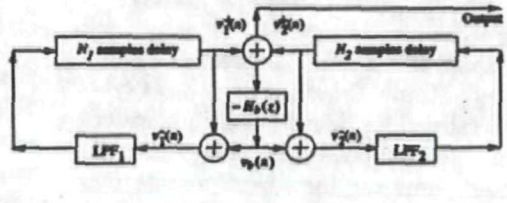


Figure 13: General linear coupling of two equal-impedance strings using a common bridge filter.

Since $R_b(z)$ is positive real, it is readily verified that

$$|2H_b(e^{j\omega T}) - 1| \leq 1$$

which restricts the set of coupling filters to those having frequency response values in the circle of radius 1/2 centered at $z = 1/2$ in the complex plane. If the two coupled strings are taken to be lossless (e.g., two pure delay loops), then this constraint becomes the stability condition for the overall system. If the amplitude and phase response of the filter are denoted $G(\omega)$ and $\theta(\omega)$, respectively, the passivity constraint may be written in the form

$$\cos[\theta(\omega)] \geq G(\omega)$$

Thus, the gain may approach unity only at frequencies where the phase approaches zero. In no case may the absolute value of the phase exceed 90 degrees, nor may the gain exceed 1 at any frequency. If the phase does approach plus or minus

90 degrees, the gain must approach zero also. The real part of the frequency response is always positive, and it may approach zero only if the imaginary part (hence gain) also approaches zero.

If the transmission filter H_t is taken to be a real, frequency-independent gain G, corresponding to a "resistive bridge termination," the passivity constraint becomes simply

$$0 \leq G \leq 1$$

Such a set of resistive bridge couplings may be realized *without multiplies* by using gain values of the form

$$G = 2^{-K}, \quad K = 0,1,2,\ldots$$

The case $G = 1$ corresponds to a zero bridge impedance which means the two strings simply fuse into one long ideal string. The case $G = 0$ corresponds to an infinitely rigid bridge, in which case the two strings are isolated from one another. Since realistic bridges are close to rigid, we desire many settings in the vicinity of $G = 0$, and the "right-shift" $G = 2^{-K}$ has this property.

Another passive, multiply-free, transmission filter is any filter having a transfer function of the form

$$H_t(z) = 2^{-K}(1 + z^{-1}), \quad K = 1,2,\ldots$$

Thus, the right-shifter is augmented by a unit-sample delay and a summer. In this case, the bridge appears more rigid at high frequencies, behaving like a mass. Spring-like bridges can be implemented using a transmission filter of the form $H_t(z) = 2^{-K}(1 - z^{-1}), \quad K = 1,2,\ldots$. These are one-zero filters. Corresponding multiply-free one-pole versions are $H_t(z) = 2^{-K}/(1 - z^{-1})$ for a mass-like bridge and $H_t(z) = 2^{-K}/(1 + z^{-1})$ for a spring-like bridge.

Any passive transmission filter can be cascaded with any resistive loss. Also, one mass-like and one spring-like transmission filter as defined above can be cascaded. However, instability can result if two mass or two spring filters are used in cascade. For higher orders, it is necessary to go to second-order sections whose poles and zeros interlace near the unit circle so as to obey the phase constraint. (Note that even the simple filter z^{-1}, corresponding to a unit sample delay, reaches phase π at half the sampling rate and is therefore not a passive transmission filter.) Physically, pole-zero interlacing corresponds to the fact that a bridge impedance "looks like a spring" at frequencies from 0 to the first resonance frequency, then it looks like a mass up to the next resonance, then like a spring again, and so on, up to half the sampling rate. These are the classical "stiffness controlled" and "mass controlled" frequency regions of a lightly damped impedance. Right on a resonance frequency, the phase goes to 0 and the impedance "looks like a dashpot" in that the impedance is real.

Note that a yielding bridge introduces losses into all attached strings. Therefore, in a maximally simplified implementation, all string loop filters may be eliminated, resulting in only one filter—the transmission filter—serving to provide all losses in a coupled-string simulation. If that transmission filter is multiply free, then so is the entire multi-string simulation.

6 Summary

Techniques applicable to efficient synthesis of stringed musical instruments were presented, along with some further extensions. Specific techniques included lumping of distributed losses and dispersion, convolving body resonators and string excitation signals into aggregate excitation look-up tables, bowed strings as periodically plucked strings, single-filter coupled strings implementation, and ways to eliminate multiplications. Since multiplies are intrinsically more expensive than additions in linear number systems (e.g., a 16 by 16 multiply requires 16 extended-precision additions), the number of voices possible in a VLSI implementation normally goes up as the number of multiplications goes down.

7 Appendix

To introduce losses into the wave equation, odd-order time derivatives such as $\dot{y}$, $\partial^3 y/\partial t^3$, and $\partial^5 y/\partial t^5$ are introduced. To introduce dispersion, e.g., for stiff strings and bars, a fourth-order term proportional to y'''' is added in. A general, linear, time-invariant, differential equation which covers all of these cases is

$$\sum_{k=0}^{\infty} \alpha_k \frac{\partial^k y(t,x)}{\partial t^k} = \sum_{l=0}^{\infty} \beta_l \frac{\partial^l y(t,x)}{\partial x^l}$$

On setting $y(t,x) = e^{st+vx}$, (or taking the 2D Laplace transform with zero initial conditions), we obtain the algebraic equation,

$$\sum_{k=0}^{\infty} \alpha_k s^k = \sum_{l=0}^{\infty} \beta_l v^l$$

Solving for v in terms of s is straightforward in the case of simple losses and stiff strings, and doing so yields the filtering needed to simulate simple losses and dispersion [Smith, 1992]. More general cases are not solvable in closed form, but are solvable numerically. For example, note that starting at

$s = 0$, we normally also have $v = 0$ (corresponding to the absence of static deformation in the medium). Stepping s forward by a small differential $j\Delta\omega$, the left-hand side can be approximated by $\alpha_0 + \alpha_1 \Delta\omega$. Requiring the generalized wave velocity $s/v(s)$ to be continuous, a physically reasonable assumption, the right-hand side can be approximated by $\beta_0 + \beta_1 \Delta v$, and the solution is easy. As s steps forward, higher order terms become important one by one on both sides of the equation. Each new term in v spawns a new solution for v in terms of s, since the order of the polynomial in v is incremented. It appears possible that *homotopy continuation methods* [Morgan, 1987] can be used to keep track of the branching solutions of v as a function of s. For each solution $v(s)$, let $v_r(\omega)$ denote the real part of $v(j\omega)$ and let $v_i(\omega)$ denote the imaginary part. Then the eigensolution family can be seen in the form $\exp\{j\omega t \pm v(j\omega)x\} = \exp\{\pm v_r(\omega)x\} \cdot \exp\{j\omega(t \pm v_i(\omega)x/\omega)\}$. Defining $c(\omega) \triangleq \omega/v_i(\omega)$, and sampling according to $x \to x_m \triangleq mX$ and $t \to t_n \triangleq nT(\omega)$, with $X \triangleq c(\omega)T(\omega)$ as before, (the spatial sampling period is taken to be frequency invariant, while the temporal sampling interval is modulated versus frequency using allpass filters), the left- and right-going sampled eigensolutions become

$$\begin{aligned} e^{j\omega t_n \pm v(j\omega)x_m} &= e^{\pm v_r(\omega)x_m} \cdot e^{j\omega(t_n \pm x_m/c(\omega))} \\ &= G^m(\omega) \cdot e^{j\omega(n \pm m)T(\omega)} \end{aligned}$$

where $G(\omega) \triangleq e^{\pm v_r(\omega)X}$. Thus, a completely general map of v versus s, corresponding to a partial differential equation of any order, can be translated, in principle, into an accurate, local, linear, time-invariant, discrete-time simulation. The boundary conditions and intitial state determine the initial mixture of the various solution branches.

References

[Cook, 1990] P. R. Cook, "Identification of Control Parameters in an Articulatory Vocal Tract Model, with Applications to the Synthesis of Singing," Ph.D. Dissertation, Elec. Eng. Dept., Stanford University, Dec. 1990.

[Horner et al., 1993] Andrew Horner, James Beauchamp, and Lippold Haken. "Methods for Multiple Wavetable Synthesis of Musical Instrument Tones," *J. Audio Eng. Soc.*, vol. 41, no. 5, pp. 336–356, May 1993.

[Jaffe and Smith, 1983] David Jaffe and Julius Smith. "Extensions of the Karplus-Strong Plucked String Algorithm," *Computer Music J.*, vol. 7, no. 2, pp. 56–69, 1983. Reprinted in the *The Music Machine*, Roads, C., ed., MIT Press, 1989.

[Karjalainen et al., 1993] Matti Karjalainen, Juha Backman, and Jyrki Pölkki, "Analysis, Modeling, and Real-Time Sound Synthesis of the Kantele, A Traditional Finnish String Instrument," *Proc. IEEE ICASSP-93*, (Int. Conf. Acoustics, Speech, and Signal Processing).

[Karplus and Strong, 1983] Kevin Karplus and Alex Strong. "Digital Synthesis of Plucked String and Drum Timbres," *Computer Music J.*, vol. 7, no. 2, pp. 43–55, 1983. Reprinted in the *The Music Machine*, Roads, C., ed., MIT Press, 1989.

[Markle and Gray, 1976] John D. Markel and Augustine H. Gray, *Linear Prediction of Speech*, Springer-Verlag, New York, 1976.

[McIntyre et al., 1983] Michael E. McIntyre, Robert T. Schumacher, and James Woodhouse, *On the Oscillations of Musical Instruments, J. Acoust. Soc. Amer.*, vol. 74, no. 5, pp. 1325–1345, Nov. 1983.

[Morgan, 1987] A. Morgan, *Solving Polynomial Systems Using Continuation for Engineering and Scientific Computations*, Prentice-Hall Inc., Englewood Cliffs, NJ, 1987.

[Morse and Ingard, 1968] Philip M. Morse and K. Uno Ingard. *Theoretical Acoustics*, McGraw-Hill, New York, 1968.

[Rabiner and Gold, 1975] Lawrence R. Rabiner and Bernard Gold. *Theory and Application of Digital Signal Processing*, Prentice-Hall Inc., Englewood Cliffs, NJ, 1975.

[Smith and Gossett, 1984] Julius Smith, and Phil Gossett. "A Flexible Sampling-Rate Conversion Method," *Proc. IEEE Conf. Acoust. Sp. and Sig. Proc.*, vol. 2, pp. 19.4.1-19.4.2, San Diego, March 1984. An expanded version of this paper is available via anonymous ftp to ccrma-ftp.stanford.edu., directory pub/DSP/.

[Smith, 1985] Julius O. Smith. "Introduction to Digital Filter Theory," In J. Strawn, ed., *Digital Audio Signal Processing: An Anthology*. William Kaufmann, Inc., Los Altos, California, 1985. A shortened version appears in *The Music Machine*, , Roads, C., ed., MIT Press, 1989.

[Smith, 1992] Julius O. Smith. "Physical Modeling Using Digital Waveguides," *Computer Music J.*, vol. 16, no. 4, pp. 74–91, Winter 1992.

[Smith, 1987] Julius O. Smith. "Music Applications of Digital Waveguides," (A compendium containing four related papers and presentations.) CCRMA Tech. Rep. STAN-M-67, Stanford University, 1987. CCRMA Technical Reports can be obtained by calling (415)723-4971, or by writing to the address on the first page. A master bibliography is also available.

[Vaidyanathan, 1993] P. P. Vaidyanathan, *Multirate Systems and Filter Banks*, Prentice-Hall Inc., Englewood Cliffs, NJ, 1993.

[Van Duyne and Smith, 1993] Scott A. Van Duyne and Julius O. Smith, "Physical Modeling with the 2-D Digital Waveguide Mesh," *Proc. International Computer Music Conference*, September, 1993, Tokyo.

[Van Valkenburg, 1960] M. E. Van Valkenburg, *Introduction to Modern Network Synthesis*, Wiley, New York, 1960.

[Weinreich, 1979] Gabriel Weinreich, "Coupled Piano Strings," *J. Acoust. Soc. Amer.*, vol. 62, no. 6, pp. 1474–1484, Dec. 1977. Also see *Scientific American*, vol. 240, p. 94, 1979.

Physical Modeling with the 2-D Digital Waveguide Mesh

Scott A. Van Duyne
savd@ccrma.stanford.edu

Julius O. Smith III
jos@ccrma.stanford.edu

Center for Computer Research in Music and Acoustics (CCRMA)
Stanford University, Stanford, CA 94305

Abstract

An extremely efficient method for modeling wave propagation in a membrane is provided by the multi-dimensional extension of the digital waveguide. The 2-D digital waveguide mesh is constructed out of bi-directional delay units and scattering junctions. We show that it coincides with the standard finite difference approximation scheme for the 2-D wave equation, and we derive the dispersion error. Applications may be found in physical models of drums, soundboards, cymbals, gongs, small-box reverberators, and other acoustic constructs where a one-dimensional model is less desirable.

1 Background Theory

There are many musical applications of the one-dimensional digital waveguide ranging from the generation of wind and string instrument tones, to flanging effects [Van Duyne and Smith, 1992], to reverberation [Smith, 1987]. We review the theoretical derivation of one-dimensional traveling waves as a basis for development of the two-dimensional digital waveguide mesh.

1.1 The 1-D Wave Equation

The one-dimensional wave equation for displacement of an ideal vibrating string may be written as follows,

$$u_{tt}(t,x) = c^2 u_{xx}(t,x),$$

where t is time, x is longitudinal position along the string, $u(x,t)$ is transverse displacement of the string as a function of time and position, u_{tt} is the second partial time derivative of u corresponding to the transverse acceleration of a point on the string, and u_{xx} is the second partial space derivative of u corresponding the "curvature" of the string at a point. The equation says that the force which accelerates a point on the string back toward its rest position is proportional to how tightly the string is curved at that point [Morse and Ingard, 1968].

It is easy to verify by substitution that this equation is solved by the sum of two arbitrary traveling waves,

$$u(t,x) = g^+(x - ct) + g^-(x + ct),$$

where $g^+(x - ct)$ represents an arbitrary fixed wave shape traveling to the right, and $g^-(x + ct)$ represents an arbitrary fixed wave shape traveling to the left. To see that these waves travel at speed c, note that, as t is increased by 1 in the expression, $g^+(x - ct)$, x must be increased by c for the function argument to remain unchanged. The wave speed is given by $c = (T/\varepsilon)^{0.5}$, where T is the constant tension on the string and ε is the mass per unit length. Intuitively, we may check that increased tension should speed up wave travel and increased mass should slow it down.

1.2 The Digital Waveguide

The traveling wave solution to the one-dimensional wave equation may be implemented digitally with a pair of bi-directional delay lines as shown in Figure 1. The upper rail contains a signal traveling to the right and the lower rail contains a signal traveling to the left. This structure is known as the digital waveguide. Two arbitrary traveling waves propagate independently in their respective left and right directions, while the physical wave amplitude at any point may be obtained by summing the left- and right-going waves.

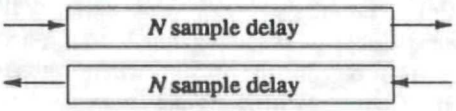


Figure 1. The Digital Waveguide

1.3 Force, Velocity, and Impedance

We need not choose displacement as our wave variable. By taking the time derivative of displacement waves, we may obtain velocity waves. In this case, the physical transverse velocity at a point on the string is the sum of its two traveling components, $v = v^+ + v^-$. The transverse force component on a string is proportional to the slope of the string at a given point for small displacements. Therefore, force waves may be obtained by properly scaling the position derivative of the displacement waves to obtain, $f = f^+ + f^-$. See [Smith, 1992] and [Smith, 1993] for a full derivation of all these wave relationships and their digital implementation.

Force and velocity are a convenient choice of wave variables as there is a well understood impedance relation between force and velocity in mechanical systems which is analogous to the impedance relation between voltage and current in electrical systems. There is also a *wave* impedance relation between force and velocity waves on strings which is analogous to the *wave* impedance relation between voltage and current waves on electrical transmission lines. The acoustical system of the vibrating air column is also mathematically equivalent to both the vibrating string and the electrical transmission line.

The wave impedance relationship between the traveling components of force and velocity for the string can be written,

$$f^+ = Rv^+,$$
$$f^- = -Rv^-,$$

where $R = (T\varepsilon)^{0.5}$. Intuitively, when a force is applied transversely to a string, the resultant transverse velocity should be slower for greater string mass and also slower for greater string tension.

1.4 The Lossless Scattering Junction

It is useful to be able to interconnect waveguides of possibly varying wave impedance at junctions which may be lossless, or which may be loaded with impedances of their own, or be driven by external forces. For example, driving a violin string with a pulsed noise signal representing the bow requires a scattering junction on the string where the bow divides it [Chafe, 1990]. Tone holes in wind instrument models may take advantage of scattering junctions. Strings may be coupled together at a bridge via scattering junctions [Smith, 1993]. Scattering junctions may be used to build up acoustic tubes of varying diameter by joining segments of cylindrical tubes [Cook, 1990]. Julius Smith [1987] has developed a reverberation algorithm which depends on interconnecting any number of varying length and varying impedance waveguides into an arbitrarily elaborate network. The membrane model presented in this paper may be viewed as a canonical form of this reverberation structure.

When several strings, say N of them, intersect at a single point, or junction, without loss of energy, we have a "series" junction and require two conditions: (1) that the velocities of all the strings at the junction be equal since they are all moving together at that point,

$$v_1 = v_2 = \ldots = v_N,$$

and (2) that the forces exerted by all the strings must balance each other at that point, i.e., they must sum to zero,

$$f_1 + f_2 + \ldots + f_N = 0.$$

Note that the acoustic tube junction is "parallel" and has the dual constraints, i.e., that the *pressures* must all be equal and the *flows* must sum to zero. Figure 2 shows a schematic representation of waveguides intersecting in a lossless scattering junction. The line segments with opposing arrows on them represent the bi-directional delay lines of the digital waveguide shown in Figure 1, with their associated wave impedance, R_i. The circumscribed S represents the junction.

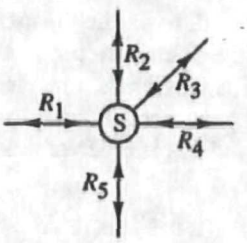


Figure 2. Scattering Junction for N=5 case.

Combining the two series junction constraints with the wave variable definitions, $v_i = v^+_i + v^-_i$ and $f_i = f^+_i + f^-_i$, and with the wave impedance relations, $f^+_i = R_i v^+_i$ and $f^-_i = -R_i v^-_i$, we can derive the lossless scattering equations for the interconnection of several strings,

$$v_J = \left(2\sum_i R_i v^+_i\right) / \sum_i R_i$$
$$v^-_i = v_J - v^+_i,$$

where v_J represents the junction velocity, the v^+_i are the incoming waves at the junction, and the v^-_i are the outgoing waves. These equations say that, as a wave is coming into a junction along a string, some portion of the wave reflects off the junction and travels back where it came from, while the rest of it travels into the junction and is divided among the outgoing waves along the other strings. The relative proportions of this scattering effect is dependent only on the relative impedances of the strings and not on their length.

2 The Two-Dimensional Case

2.1 The 2-D Wave Equation

The two-dimensional wave equation for displacement of an ideal membrane may be written as follows,

$$u_{tt}(t,x,y) = c^2\,[u_{xx}(t,x,y) + u_{yy}(t,x,y)],$$

where t is time, x and y are spatial coordinates on the membrane, $u(t,x,y)$ is transverse displacement of the

membrane as a function of time and spatial position [Morse and Ingard, 1968].

In the one-dimensional string case, we could solve and implement the wave equation as two bi-directional traveling waves. In the 2-D membrane case, the traveling wave solution involves the integral sum of an infinite number of arbitrary plane waves traveling in all directions,

$$u(t,x,y) = \int g_\alpha(x\cos\alpha + y\sin\alpha - ct)\,d\alpha$$

Since assigning one waveguide to each of the infinite plane waves is not feasible, we need an alternative approach.

2.2 The 2-D Digital Waveguide Mesh

Proposed in this paper is a formulation of the 2-dimensional wave equation in terms of a network of bi-directional delay elements and 4-port scattering junctions. This structure can be viewed as a layer of parallel vertical waveguides superimposed on a layer of parallel horizontal waveguides intersecting each other at 4-port scattering junctions between each bi-directional delay unit. Figure 3 shows such a mesh. In the canonical case, the scattering junctions are taken to be equal impedance lossless junctions and the interconnecting waveguides are of unit length.

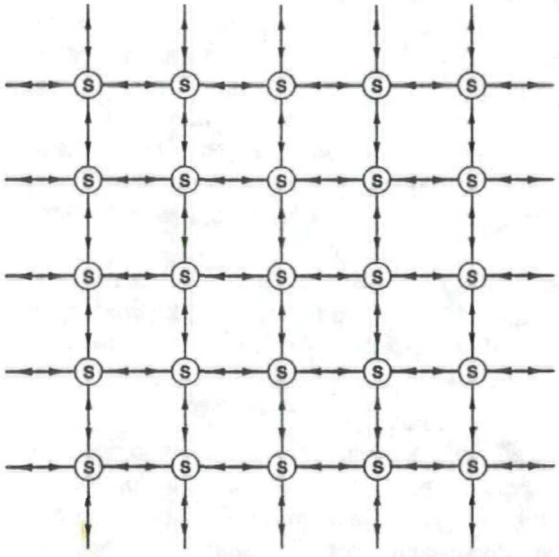


Figure 3. The 2-D Digital Waveguide Mesh

If we view the mesh as a lattice of interconnected vibrating strings, the displacement velocities at the four ports of each junction must be equal, and the forces at each junction must sum to zero; in this case, we have *series* scattering junctions with force or velocity waves traveling in the two-port, bi-directional, delay units. On the other hand, if we view the mesh as a lattice of interconnected acoustic tubes, the pressures at each junction must be equal, and the flows into each junction must sum to zero; in this case, we have *parallel* scattering junctions with pressure or volume velocity waves traveling through the delay units.

3 Empirical Analysis

It is evident that, given an initial excitation at some point on the digital waveguide mesh, that energy from that excitation will tend to spread out from the excitation point more and more as the traveling waves scatter through the junctions. It is not, however, easy to see that the wave propagation on the mesh converges to that on the ideal membrane.

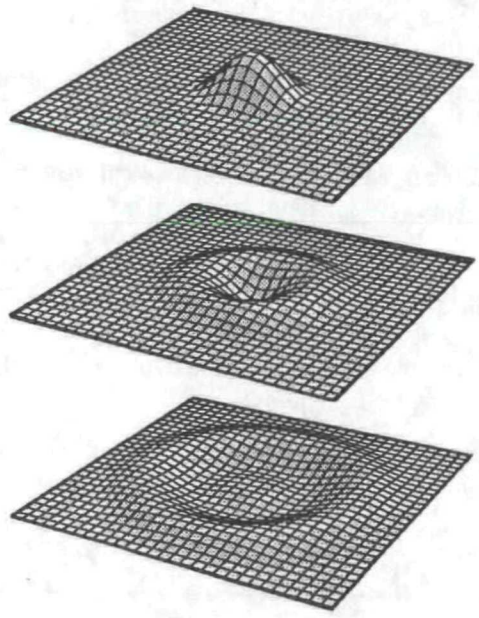


Figure 4. Wave Propagation on the Mesh

3.1 Animation of the Mesh

A visual verification of the waveguide mesh algorithm can be seen in Figure 4, which shows three separate time frames of an animation computed directly from the algorithm. The top frame shows the initial deflection loaded into the mesh. Each intersecting grid point represents a scattering junction. The next two frames show the circular propagation outward of the initial excitation in a way consistent wave propagation on the ideal membrane.

3.2 Sounds from the Mesh

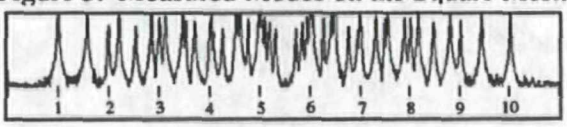
As another check of the mesh algorithm, we can compare the expected modal frequencies on an ideal membrane with those generated from the mesh model. The allowed frequencies in a theoretical ideal square membrane with clamped edges are proportional to $(m^2 + n^2)^{0.5}$, for m = 1,2,..., and n = 1,2,...[Morse and Ingard, 1968]. These modes may be labeled (m,n) for any given m and n. In Table 1 is computed a list of the normalized frequencies of the first few of these modes given as multiples of the lowest allowed frequency.

Table 1. Modes on Ideal Square Membrane

(1,1) → 1.00	(1,5) → 3.60
(1,2) → 1.58	(2,5) → 3.80
(2,2) → 2.00	(4,4) → 4.00
(1,3) → 2.24	(3,5) → 4.12
(2,3) → 2.55	(1,6) → 4.30
(1,4) → 2.92	(2,6) → 4.47
(3,3) → 3.00	(4,5) → 4.50
(2,4) → 3.16	(3,6) → 4.74
(3,4) → 3.54	(5,5) → 5.00

Figure 5 is a spectral analysis of a sound generated by a square 10 junction by 10 junction digital waveguide mesh reflectively terminated at the boundaries. A careful inspection of the plot will reveal that the theoretical modal frequencies listed in Table 1 are all present and accounted for in the sound generated by the model. This would indicate that the mesh is doing the right thing.

Figure 5. Measured Modes on the Square Mesh



Frequencies up to half the sampling rate are shown in Figure 5. However, notice that the spectrum mirrors around one quarter of the sampling rate. This symmetry, which also occurs in the one dimensional waveguide case, is a result of the fact that, when the waveguide or waveguide mesh is reflectively terminated, all the unit delays on the upper rails can be commuted down to the lower rails making the system a function of z^{-2}, in effect, over-sampling the system by a factor of two.

4 Mathematical Analysis

Given the the function $f(x)$, one may approximate its first derivative by the difference,

$$f'(x) \approx \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

By applying this expression to itself we may arrive at the standard difference scheme approximation for the second derivative,

$$f''(x) \approx \frac{f(x+\Delta x) - 2f(x) + f(x-\Delta x)}{\Delta x^2}$$

Computable and numerically stable difference schemes can be found for many partial differential equations by substituting approximations of this kind into the equation. The digital waveguide mesh algorithm, in fact, may be interpreted as a difference scheme for computation of the two-dimensional wave equation.

4.1 The Mesh as a Difference Scheme

In the digital waveguide mesh, we require that the impedances in all directions be equal, as we would desire for the isotropic membrane case. Setting the wave impedances of the four unit waveguides attached to each junction point in the mesh equal, i.e., $R_1 = R_2 = R_3 = R_4$, the scattering equations then reduce to,

$$v_{J\,l,m}(n) = \frac{v^+_{1\,l,m}(n) + v^+_{2\,l,m}(n) + v^+_{3\,l,m}(n) + v^+_{4\,l,m}(n)}{2},$$

$$v^-_{i\,l,m}(n) = v_{J\,l,m}(n) - v^+_{i\,l,m}(n),$$

where the l,m indices represent the spatial position of the junction in the mesh and the n index represents the current time sample. $v_{J\,l,m}(n)$ represents the velocity of the junction at position l,m at time n; $v^+_{i\,l,m}(n)$ and $v^-_{i\,l,m}(n)$ represent the four input and output waves to that junction, respectively. As an implementation note, observe that this junction computation may be performed with 7 adds (or subtracts), 1 shift (to divide by 2), and *no multiplies.*

In addition to these scattering equations, we may also note that the sum of the inputs to a junction equals the sum of the outputs,

$$\sum_i v^+_{i\,l,m}(n) = \sum_i v^-_{i\,l,m}(n),$$

and that, since the junctions are interconnected by unit delay elements, the input at one port of one junction is equal to the output at the opposing port of the adjacent junction *at the previous time sample,*

$$v^+_{\text{opposite}}(n) = v^-_{\text{opposite}}(n-1).$$

With a little perseverance, one may manipulate all these relations algebraically into the following difference equation,

$$\begin{aligned} v_{J\,l,m}(n) - 2v_{J\,l,m}(n-1) + v_{J\,l,m}(n-2) = \\ 0.5[v_{J\,l,m+1}(n-1) - 2v_{J\,l,m}(n-1) + v_{J\,l,m-1}(n-1)] \\ + 0.5[v_{J\,l+1,m}(n-1) - 2v_{J\,l,m}(n-1) + v_{J\,l-1,m}(n-1)], \end{aligned}$$

A comparison of this difference scheme with the two-dimensional wave equation,

$$u_{tt}(t,x,y) = c^2(u_{xx}(t,x,y) + u_{yy}(t,x,y)),$$

reveals that it is the standard second-order difference scheme for the hyperbolic partial differential wave equation for the ideal membrane, with wave propagation speed $c = 2^{-0.5} \approx 0.7$, and the time and spatial sampling intervals ($X=\Delta x$, $Y=\Delta y$, $T=\Delta t$) taken to be equal to each other. The mesh implements a wave propagation speed of one-half unit diagonal distance per time sample. Intuitively, when we superimposed the perpendicular layer of parallel waveguide strings to form the mesh, we doubled the mass density per unit area, thereby reducing the wave speed by one over the square root of two.

Defining, $\lambda = TX^{-1} = 1$, we observe that the Courant-Friedrichs-Lewy stability condition, $|c\lambda| \leq 2^{-0.5}$, is satisfied by this difference scheme [Strikwerda, 1989]. This condition says that for a difference scheme to track the solution of a hyperbolic equation with two space dimensions, the *cone* of dependence for each point of the continuous solution must lie within the *pyramid* of dependence for each point of the difference scheme solution. Since the condition is satisfied in the equality, the lowest possible dissipation and dispersion error for this particular scheme is obtained.

The numerical approximation schemes for initial value problems involving second-order hyperbolic partial difference equations usually require a multi-step time scheme which retains values for at least two previous time frames. This is to cope with the second partial time derivative in the equation. The waveguide mesh reduces this structure to a *one-step* time scheme where each new time frame may be computed wholly from the previous time frame. This is made possible by the use of traveling wave components in place of physical wave variables.

4.2 Von Neumann Error Analysis

Von Neumann analysis of finite difference scheme approximations of partial differential equations uses Fourier transform theory to compare the evolution over time of the *spatial spectrum* in the continuous time solution to that in the discrete time approximation [Strikwerda, 1989]. Recall that to solve an ordinary linear differential equation, we may reduce the problem to a polynomial in s by taking the Laplace transform and replacing orders of derivatives with powers of s. Similarly, we can take a *spatial transform* of a *partial* differential equation with independent time and space variables to obtain an *ordinary* differential equation describing the evolution of *spatial spectra* over time. From here, we can check how the spatial spectrum evolves after a time delay of T seconds. The ratio of the spatial spectrum at time $t + T$ to the spatial spectrum at time t is known as the *spectral amplification factor.*

Recall that in the discrete case, to solve a time-indexed difference equation, we may reduce the problem to a polynomial in z^{-1} by taking the Z-transform and replacing samples of time delay in the index with powers of z^{-1}. If we have a time- and space-indexed difference scheme approximation to compare with our partial differential equation, we may perform a similar spectral evolution analysis on the difference equation using *discrete* Fourier transforms to obtain a *discrete spectral amplification factor.*

In the case of the ideal membrane equation, we already know the solution is an integral sum of plane waves all moving at constant speed c. The speed of wave travel is independent of spatial frequency, i.e., smooth low spatial frequency waves travel at the same speed as jagged or ripply-looking high spatial frequency waves. This means that there is no frequency *dispersion* in the ideal membrane. After T seconds, the position of a plane wave of spatial frequency ξ, traveling in direction α, would have moved forward a distance of cT, corresponding to a spatial phase shift of $-c\xi$T. The *spectral amplification factor* would then be $e^{-jc\xi T}$.

In the difference scheme derived in Section 4.1, computation of a *discrete spectral amplification factor* is a little messier. Unfortunately, the speed of plane wave travel on the digital waveguide mesh is dependent on both the direction of travel, α, and on the spatial frequency of the wave, ξ. The *discrete spectral amplification factor* may be written in the form $e^{-jc'(\alpha,\xi)\xi T}$, where $c'(\alpha,\xi)$ represents the direction and frequency dependent speed of plane wave travel on the waveguide mesh.

When making two-dimensional spatial transforms, we take $x \leftrightarrow \xi_1$ and $y \leftrightarrow \xi_2$. The coordinate frequencies, ξ_1 and ξ_2, are hard to understand conceptually, but viewing the transform in polar coordinates, we see that the point (ξ_2,ξ_1) in the two-dimensional frequency space is referring to a plane wave of spatial frequency $\xi = (\xi_1^2 + \xi_2^2)^{0.5}$, oriented in the radial direction, $\alpha = \tan^{-1} \xi_2/\xi_1$.

Using the procedure outlined above, a closed-form

expression for the normalized speed of plane wave travel in the waveguide mesh may be found,

$$c'(\xi_1,\xi_2) / c = 2^{0.5} (\xi T)^{-1} \tan^{-1} (4 - b^2)^{0.5} b^{-1},$$

where $b = \cos \xi_1 T + \cos \xi_2 T$.

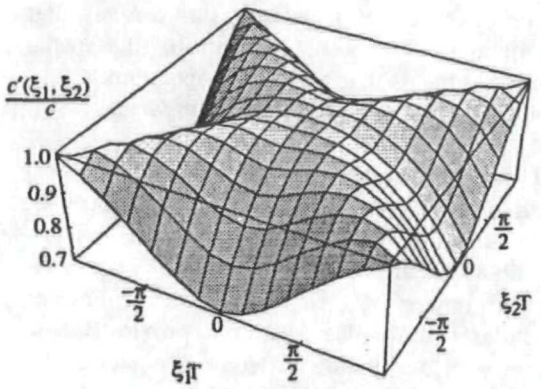


Figure 6. Wave Travel Speed vs. Frequency

Figure 6 shows a plot of the normalized wave travel speed on the mesh versus spatial frequency. The center region of the plot corresponds to low spatial frequencies; the outer regions of the plot correspond to higher spatial frequencies. The angular position on the plot, as seen from the frequency plane origin, corresponds exactly to the direction of plane wave travel on the mesh. Notice that near the center of the plot, corresponding to smooth, low frequency plane waves, the c'/c ratio is fairly close to 1. Also, $c'/c = 1$ *exactly* along the diagonals, corresponding to *no dispersion at any spatial frequencies when traveling in a direction diagonal to the mesh coordinate system.* In waves traveling along the coordinate axes of the mesh, we see a fall off in travel speed in the higher spatial frequencies.

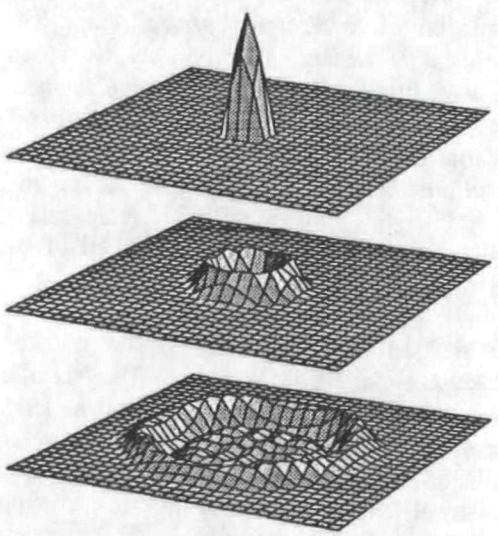


Figure 7. Wavefront Dispersion on Mesh

Figure 7 shows three time frames of the mesh initialized with a deflection containing high spatial frequencies. Notice how the wavefront smooths out along the mesh coordinate directions, corresponding to high frequency dispersion, while it remains sharp along the diagonal directions, corresponding to *no* dispersion. In a bounded mesh, speed distortion results in a mistuning of resonant modes. This distortion can be reduced by allpass filtering and/or warping of the membrane boundary in a compensating manner. Oversampling the mesh and low-passing can eliminate the effect to arbitrary accuracy. We note that the high frequency modes of a membrane become so dense that, in audio contexts, this error may not be important.

5 Implementation Features

The digital waveguide mesh may be computed in parallel, and without multiplies. In addition numerical round off loss may be redistributed back into the mesh to create a zero-loss system.

5.1 Two Pass Parallel Computation

The network elements in the 2-D digital waveguide mesh are of two types: 4-port scattering junctions and 2-port bi-directional unit delays. If the unit delays are double buffered, so that each delay has its own input and output buffers, the computation of all the elements in the mesh can be segregated and computed in any arbitrary order or in parallel, according to the following two pass computation scheme: (1) The scattering junction outputs are computed from their known inputs and placed at the junction outputs. This constitutes the scattering pass. (2) The outputs from each scattering junction are placed at the inputs of the adjacent scattering junctions, thereby implementing the bi-directional delay units. This constitutes the delay pass.

Due to the possibility of arbitrary ordering of the scattering computations, implementation on a parallel computing architecture with local four-sided connectivity between processors is ideal for the mesh algorithm. In this implementation, the junction equations are computed in the processors; and then the data transfer cycle is used to transfer data from the outputs of each processor to the appropriate inputs of the adjacent processors.

Since the equal impedance 4-port lossless scattering junction is multiply-free, as pointed out in Section

4.2, a VLSI implementation may be constructed with a handful of gates with no need for hardware multipliers. Since the junctions may be computed in parallel, the whole mesh may be computed in the time it takes to do 7 adds (i.e., 3 adds and 4 subtracts) and one shift. In fact, the four subtracts may be performed in parallel.

5.2 Energy Preserving Junctions

When performing the multiply-free junction computation, one divide by two is required. If a simple sign-preserving right shift and truncation is used for this operation, the junction value is rounded toward zero in the case of positive numbers and rounded away from zero in the case of negative numbers. This is a round down in both cases, which could introduce a negative offset into the values of the mesh which may eventually lead to numerical instability or reduced dynamic range, if there is no loss in the system somewhere else.

The usual solution to such a problem would be to make a conservative rounding toward zero in both the positive and negative cases. This way no energy and no DC drift will be introduced into the system. This method is known to work quite well in one-dimensional feedback loops. Unfortunately, in the two-dimensional mesh case, there are so many junctions that the cumulative losses in all the junctions add up to a noticeable amount.

An energy preserving method of junction computation may be constructed as follows. When shifting a binary number to the right, there are exactly two cases: (1) the low-order bit which is shifted off the end of the word is zero, and the computation is exact, or (2) the low-order bit is one, and the error is exactly 0.5. When the junction inputs are subtracted off v_J in computing the scattering junction outputs, this error is magnified by 4 and the 0.5 error propagates into the four output signals equally. Note that the error is in the same direction in all four cases so the total error is $4 \cdot 0.5 = 2$ full bits. To preserve energy in the mesh, round two output signals down (i.e., just truncate) and round the other two output signals up (i.e., add the low-order bit back in after truncating). This re-distribution of the error produces a numerically exact lossless scattering junction. In effect, the slight numerical error has been converted into a slight scattering dispersion error. Whereas the numerical error was problematic, the dispersion adjustment is vanishingly small.

6 Modeling with the Mesh

To build a model of a drum membrane, we need to clamp down the boundary of the mesh, corresponding to terminating the mesh reflectively, inverting the traveling waves at the edges. Since the 2-D digital waveguide mesh is just a big linear system, filters representing loss in the system are easily interconnected, and may be consolidated as desired around the rim due to commutativity of linear systems.

Modeling a stiff plate might be accomplished by letting the waves reflect off the boundary with out inversion (for an unclamped plate). To help out with the greater spacing of higher modes in the plate caused by stiffness, some appropriate allpass filtering might be introduced.

Figure 8 suggests the possibility of modeling a harpsichord by connecting an array of waveguide plucked string models to a waveguide mesh representing the sounding board via 5-port junctions. The soundboard mesh would have appropriate boundary filters with low pass characteristics to represent loss and allpass characteristics to represent stiffness effects in the board. In this model the mesh is used as a resonant coupling connection which both reverberates the string outputs and scatters energy into strings which have not been plucked, thereby inducing sympathetic vibrations.

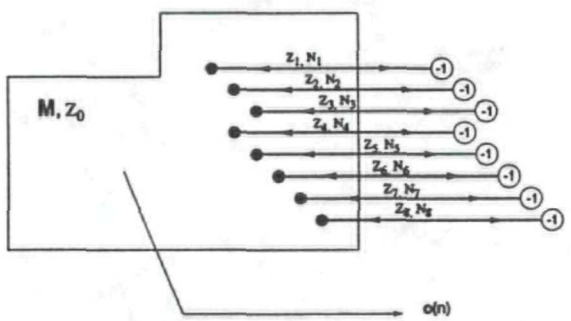


Figure 8. The 8-String Harpsichord.

6.1 Extensions of the Mesh

The mesh algorithm assumes nothing about its boundary conditions. There is nothing to stop one from connecting one edge of the mesh to the opposite edge to produce a cylindrical topology, as shown in Figure 9. Furthermore, it is straight forward to extend the algorithm into three dimensions by layering several 2-D meshes above each other, replacing all the 4-port junctions with 6-ports and

connecting up the layers. This topology is shown in Figure 10.

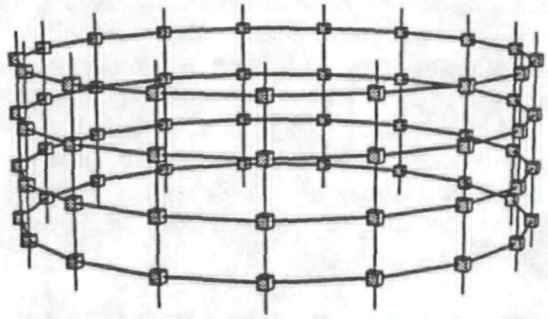


Figure 9. The Cylindrical Mesh

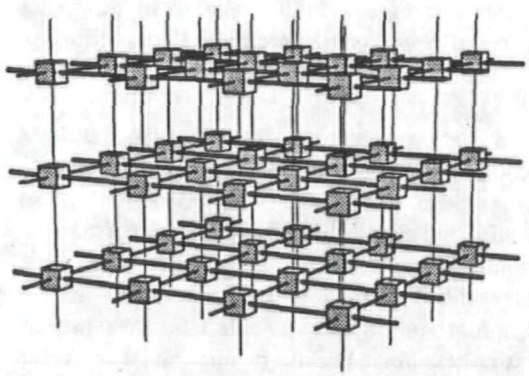


Figure 10. The 3-D Digital Waveguide Mesh

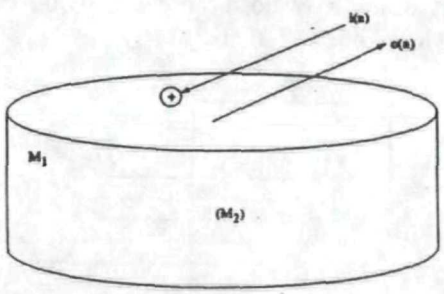


Figure 11. The 3-D Drum Model.

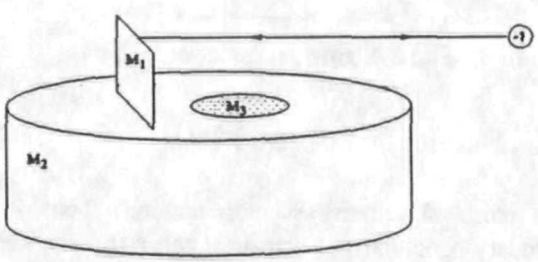


Figure 12. The 3-D, One-String Guitar.

With constructs such as these, fully physical models of musical instruments can me made. For example, Figure 11 shows the 3-D drum model with a mesh modeling the drum head, connected to a stiff cylindrical mesh modeling the sides of the drum and 3-D mesh inside the drum modeling the air cushion. Figure 12 shows a guitar model with 2-D mesh material modeling the bridge and body shell and 3-D mesh material modeling the resonant body cavity.

7 Summary

Although finite element and difference scheme approximation methods are known which can help with the numerical solution of the 2-D wave equation, these methods have two drawbacks: (1) their heavy computational time is orders of magnitude beyond reach of real time, and (2) traditional problem formulations fit poorly into the physical model arena of linear systems, filters, and network interactions. On the other hand, the 2-D digital waveguide mesh formulation proposed in this paper, while corresponding exactly with the standard difference scheme approximation, may be implemented in a fully parallel, multiply-free formulation; the energy preserving, digitally exact round-off method eliminates numerical problems; the mesh extends simply to 3- or N-dimensions; and, finally, the algorithm is a linear network which connects up easily to other physical models.

References

[Chafe, 1990] Chris Chafe. "Pulsed Noise in Self-Sustained Oscillations of Musical Instruments," *Proceedings IEEE International Conf. Acoust. Speech and Signal Proc.*, Albuquerque, NM.

[Cook, 1990] Perry Cook. *Identification of Control Parameters in an Articulatory Vocal Tract Model, with Applications to the Synthesis of Singing.* Ph.D. Diss., Elec. Eng. Dept., Stanford University.

[Morse and Ingard, 1968] P. M. Morse and K. U. Ingard. *Theoretical Acoustics.* McGraw-Hill, New York.

[Smith, 1993] Julius O. Smith. "Efficient Synthesis of Stringed Musical Instruments," *Proceedings ICMC*, Tokyo.

[Smith, 1992] Julius O. Smith. "Physical Modeling Using Digital Waveguides," *Computer Music Journal*, 16:4.

[Smith, 1987] Julius O. Smith. *Music Applications of Digital Waveguides.* CCRMA, Stanford Univ., Stanford, CA, Tech. Rep. STAN-M-39.

[Strikwerda, 1989] J. Strikwerda. *Finite Difference Schemes and Partial Differential Equations.* Wadsworth & Brooks, Pacific Grove, CA.

[Van Duyne and Smith, 1992] S. A. Van Duyne and J. O. Smith. "Implementation of a Variable Pick-Up Point on a Waveguide String Model with FM/AM Applications," *Proceedings ICMC*, San Jose, CA.