An Improved Digital Waveguide Model of a Flute - Implementation Issues

Vesa Välimäki[1,2], Rami Hänninen[3], Matti Karjalainen[2]

1. Helsinki University of Technology, Laboratory of Acoustics and Audio Signal Processing, Espoo, Finland

2. University of Westminster, School of Electronic and Manufacturing Systems Engineering, London, UK

3. Helsinki University of Technology, Telecommunications and Multimedia Laboratory, Espoo, Finland

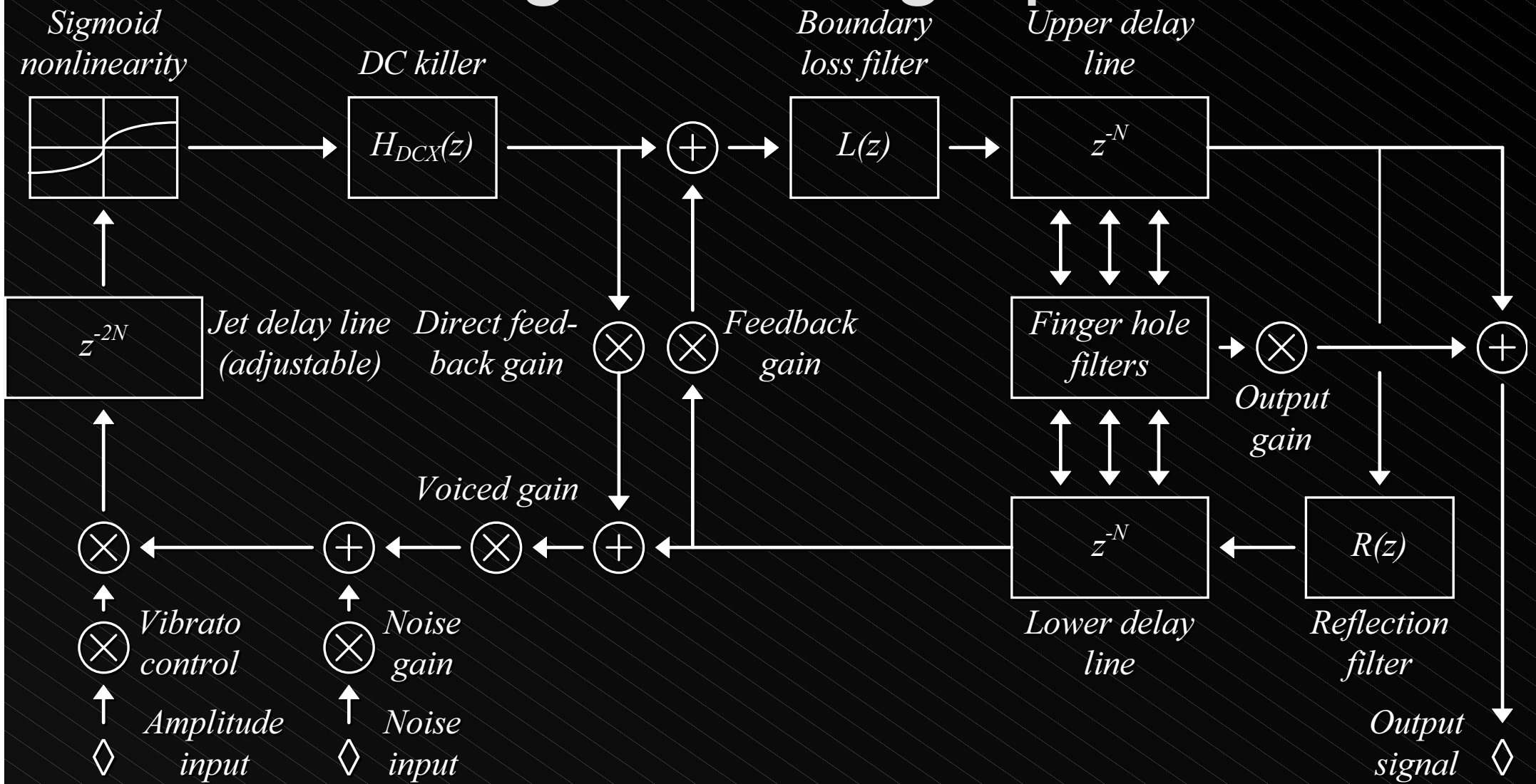
Introduction

- Physical modeling of musical instruments has become a popular approach to digital sound synthesis
- Digital waveguide modeling is one of the most important methods of the many modeling techniques
- Basic theory of digital waveguide modeling developed by J. O. Smith (CCRMA, Stanford Univ.) in the 1980's
- The first real-time waveguide flute model at the Acoustics Lab of HUT was built in 1991-1992 (Karjalainen et al., 1991; Välimäki et al., 1992)
- Recently, physical modeling of the flute has been a target of active research (Coltman, 1992; Chafe 1995; Cook, 1995; Verge, 1995; Magalotti et al., 1995)

Features of the New Flute Model

- Digital waveguide model
- Full set of finger holes (also introduced by Cook (ICMC'95), using a different approach)
- Fractionally delayed finger hole filters using FIR interpolation and deinterpolation
- Overblowing, vibrato, real flute finger tables
- Smooth note transition
- Real-time C++ implementation on a fast Unix-workstation (16 kHz)

Signal Flowgraph



Implementation

Matlab

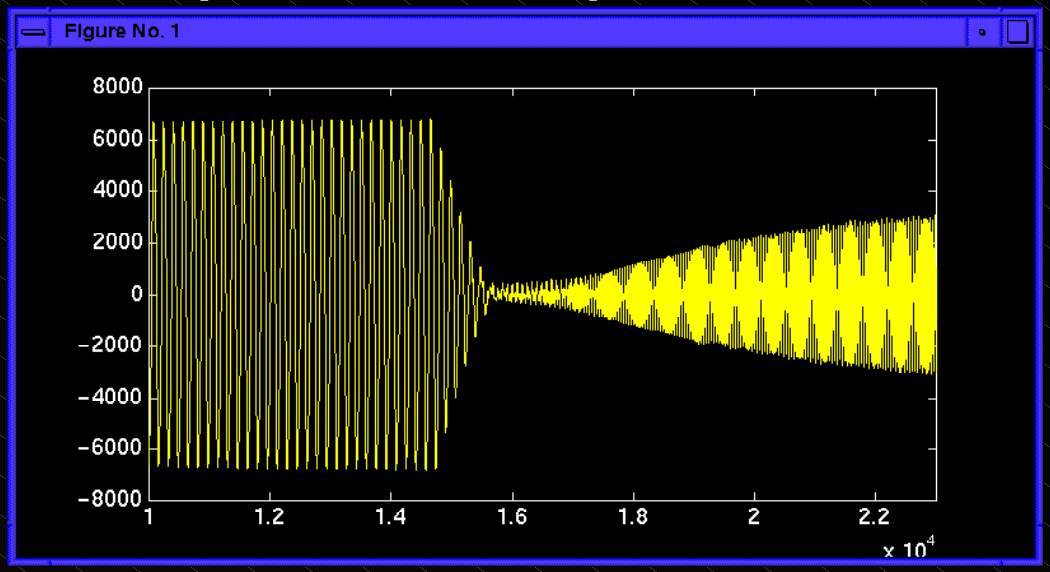
- Linear vectorized delay line
- Finger hole filtering vectorized with matrix operations
- Easy to analyze
- Portable

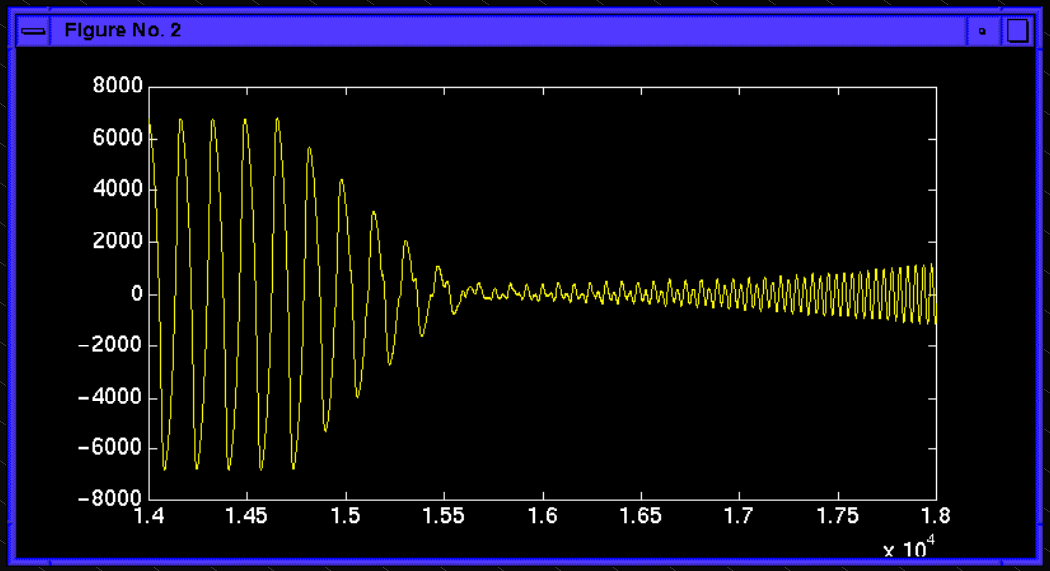
C++

- Traditional ring-buffered delay line
- Each finger hole filter is an C++ object instance
- Fast
- Portable

Simulation (Matlab)

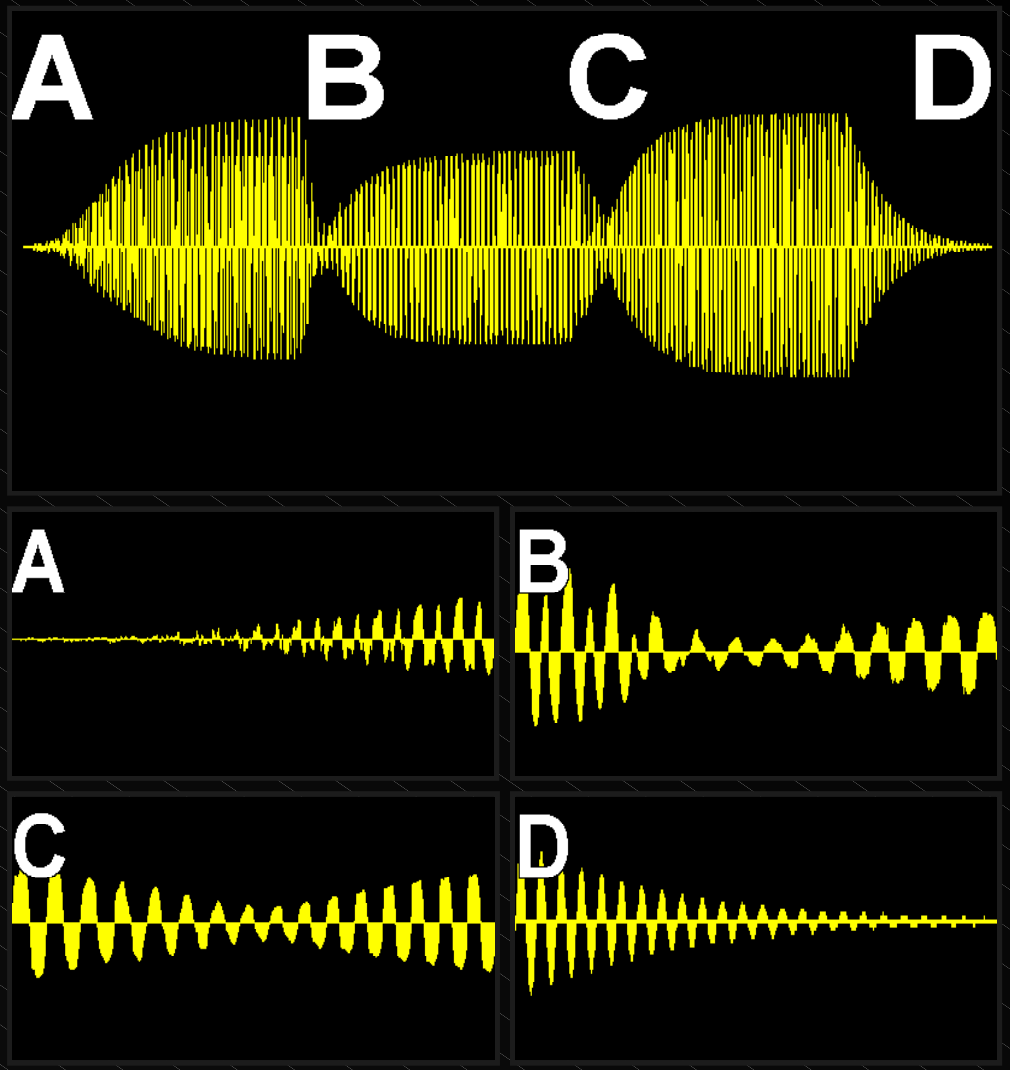
- Simulated on a 60 MHz HP 9000/712
- Matlab 4.2a
- Sample rate 44.1 kHz
- With 1 finger hole, computes about 230 samples/s
- With 10 finger holes, computes about 200 samples/s
- Tools to experiment with different finger hole configurations





Simulation (C++)

- Simulated on a 200 MHz SGI IP19/Mips
- Compiled with gcc 2.7.2
- Sample rate 44.1 kHz
- Computes about 18.000 samples/s
- Implemented as an easy-to-use C++ object
- Tools include a MIDI-player and manual flute parameter adjuster



Observed Behaviour

- The exact parameter values to reach a certain overblow level depend on the current state of the flute model
- With constant input signal amplitude, the output amplitude depends on the current overblow level and finger hole configuration
- A real flutist must learn how to compensate these effects. A well-implemented flute model MIDI controller should do this, too.

Future Work

- A good parametrization method is required to fine-tune the numerous (>100) parameters:
  - geometrical measurements of the locations and sizes of finger holes
  - measurement and estimation using sophisticated DSP methods (e.g., Karjalainen et al.,1995)
  - automatic learning system, such as a genetic algorithm (e.g.,Vuori and Välimäki, 1993)
- Many parameters depend on sample rate, currently only 44.1 kHz is supported
- Air turbulence in input air flow should be incorporated into the model