Virtual Piano Lab

Julius O. Smith III and Nelson Lee,

RealSimple Project*
Center for Computer Research in Music and Acoustics (CCRMA)
Department of Music, Stanford University
Stanford, California 94305

November 27, 2008

Abstract

In this lab, you will write Matlab code to find parameters for a loop filter from a recorded piano note. You will also modify an existing STK instrument. Starter code and soundfiles can be found here.[1]

1 Loop-filter Design

(25 pts) Design a loop loss filter from a recorded piano tone `D4.wav`[2] using the technique[3] described in the text.

1.1 The Overtones and Decay Times

Measure the frequency f_k and decay time τ_k for each overtone k. (Try to avoid the longitudinal modes.) Frequencies can be estimated from a zero-padded spectrum using parabolic interpolation and the corresponding decay times can be estimated by fitting a line to the Energy Decay Relief at STFT bins corresponding most closely to f_k, but you may use any method you like.

1.2 Corresponding Loop Gains

Calculate the desired loop gain $g_k$at each frequency f_k.

1.3 Minimum Phase Filter Design

Create a minimum phase desired spectrum using `mps.m`[4]. Simple linear interpolation works well to fill in values between the desired gain data points. The gain at DC should be set to the desired

*Work supported by the Wallenberg Global Learning Network

[1] http://ccrma.stanford.edu/realsimple/piano/piano-starter-code.tar.gz

[2] http://ccrma.stanford.edu/realsimple/piano/D4.wav

[3] http://ccrma.stanford.edu/~jos/pasp/Loop_Filter_Identification.html

[4] http://ccrma.stanford.edu/~jos/filters/Matlab_listing_mps_m_test.html

gain of the fundamental frequency (g_1), and the gain at $f_s/2$ can be arbitrarily set to a low value in order to force greater decay rates at higher frequencies. If you have outliers in your τ_k estimates, you may want to weed them out of the desired spectrum specifications so that the overall trend is monotonic.

1.4 invfreqz

Use `invfreqz` to design a loop filter based on your desired frequency response. What is the filter order, and what are its coefficients?

1.5 stmcb

Use `stmcb` to design a lower-order filter from your previous result. What is the filter order, and what are its coefficients?

1.6 Comparing Methods

Plot your target frequency response and overlay it with those of the two filters you designed. Also include a close-up of the low frequency region below 3 or 4 kHz

1.7 Proper Phase

Optional: Work out the proper phase for the stretched overtones, and design a filter that implements both damping and stretching.

2 The Commuted Piano

The STK instrument `CommutedPiano`[5] excites multiple strings with a filtered soundboard response. Multiple strings, coupled together at the bridge and tuned slightly out of unison, produce a strong attack, long decay, and beating effects. The soundboard response is filtered by a hammer force pulse, which can be approximated by a hanning window

$$h[n] = \frac{1}{2} + \frac{1}{2}\cos\left(\frac{2\pi n}{M}\right),$$

where M is the length of the window in samples and $n = -M/2 \ldots M/2$. The same hammer force filter is used for all the strings.

Since the hammer acts as a nonlinear spring, the shape and time duration of the pulse varies with amplitude. A soft hit produces a wider pulse, whereas a strong hit produces a taller, narrow pulse. The duration of the pulse is also dependent on the hammer mass, so that lighter hammers used in the high registers have a shorter contact duration with the string and heavier hammers used in the lower registers have a longer contact time. The contact durations are approximately 0.5 *(ff)* to 1.2 *(pp)* ms at higher registers and 2.0 *(ff)* to 4.0 *(pp)* ms at lower registers. The frequency of the piano note determines the range of durations for the force pulse, and the amplitude value is used to linearly interpolate over this range.

[5] http://ccrma.stanford.edu/realsimple/piano/piano-starter-code.tar.gz

Modify the instrument to add allpass filtering, in order to create stretched overtones in the strings. Implement the allpass filter cascade you designed in the previous problem.