Model of Electric Bass Using Extended Karplus-Strong Plucked-String Algorithm

- MUMT 618 Final Project
Louis Deng – 260656496
boyu.deng@mail.mcgill.ca

McGill University
2020

0 - Introduction

The Karplus-Strong algorithm (Fig. 1) was developed to synthesize plucked string and drum sounds with exceptionally realistic timbre and the characteristically harmonic decay (Karplus & Strong, 1983).

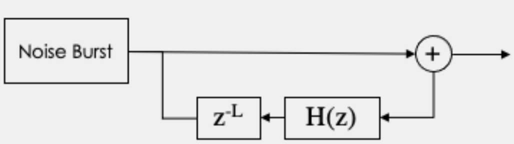

Figure 1. The original Karplus-Strong algorithm (Karplus & Strong, 1983).

In the original model, input of a short noise burst is taken and fed back with a delay commuted with a low-pass filter. This generates a faux-periodic signal (Jaffe & Smith, 1983) with harmonics that decay over time.

This model from Karplus and Strong, over the decade after it was first published, has received modifications and extensions in several other studies (Jaffe & Smith, 1983; Karjalainen, Valimaki, & Tolonen, 1998; Välimäki, Huopaniemi, Karjalainen, & Jánosy, 1995). The simplicity of this algorithm despite its quality sound, makes real-time implementation computationally viable. In a later model (Välimäki et al., 1995), the low-pass of the loop filter was modified to adapt to various timbres of instruments. Among other works to make this model sound even more realistic, input noise burst before going into the loop filter, is passed through a comb filter. The comb filter efficiently introduces zeros in the input spectrum, the process of which resembles the plucking position effect that is prevalent in all plucked string instruments (Jaffe & Smith, 1983). Additionally, body responses have been extensively studies in the form of a cascade of filters; the application of which to the Karplus-Strong algorithm-based instrumental output can yield persuasive results that are extremely close to the real counterpart (Fig. 2).

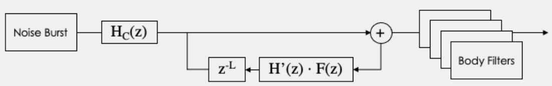

Figure 2. The extended Karplus-Strong algorithm. H_C is the comb filter that modifies the input noise burst to create the plucking-point effect.

However, the use of cascading filters can introduce a certain amount of delay in the system. Since this entire model is a linear system, the researchers have come up a way to bypass this costly computation by commuting the body filters with the input signal (Fig. 3). Hence, instead of recording the body response and try to model filters to fit the response, we can take the recording of excitation signals with the string muted. This process saves resources in the processing loop.

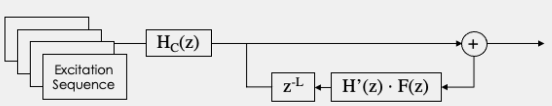

Figure 3. The commuted synthesis model based on extended Karplus-Strong algorithm.

An alternative approach to commuted body response synthesis is to use inverse filtering (Välimäki et al., 1995). If we can accurately represent the harmonic decay of the signal with the loop filter. The application of the inverse of the entire loop filter onto the recording of a plucked-string sound can yield an excitation signal that is similar to that of the commuted pluck-body response.

Due to restraint on the accessibility of proper equipment to record instrumental body response, the inverse filtering approach is used in this project, the methodology is mostly based on the paper by (Välimäki et al., 1995).

1 - Analysis of the Electric Bass Signal

The electric bass, compared to other plucked string instruments, has stiffer strings. Most basses have strictly steel strings, some strings have coiled structures. A recording of electric bass is acquired by the direct input/injection method via a DI box. The exert is a recording of a note with normal fingering velocity on 5^{th} fret on D-string. The recording is taken with a sample rate of 44100Hz with 16bit precision. [exert-1]

Using this recording, a short-time Fourier transform is performed with the FFT size of 4096 samples, hop size of 2048 samples and an overlap of 2048 samples. The 3-D graph of magnitude in dB scale over frequency and time is shown in (Fig. 4).

Upon initial observation, we can tell that the first 5 harmonics decay rather linearly at a very slow rate over the recorded 8 seconds, but from harmonics 6^{th} and up, the harmonic peaks decay at a higher rate, and there is some non-linearity in the form of energy transform between modes. Using this information, we can extract the bin numbers of the harmonic peaks along the spectra (Fig. 5) and isolate the amplitude values at the selected frequencies.

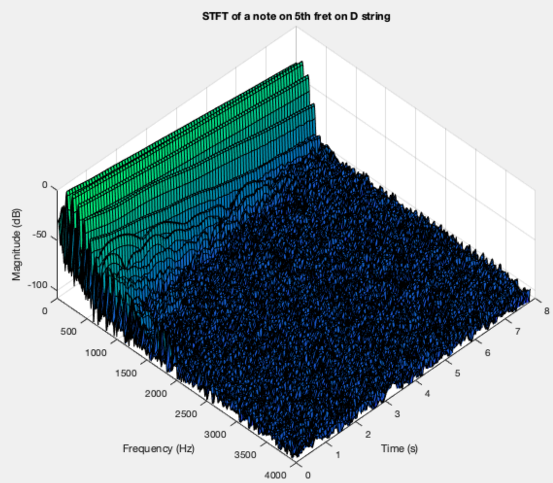

Figure 4. STFT of the note G2 by pressing on 5th fret on D string and finger-plucked with normal velocity. Frequency axis is zoomed in from 0 to 4000Hz.

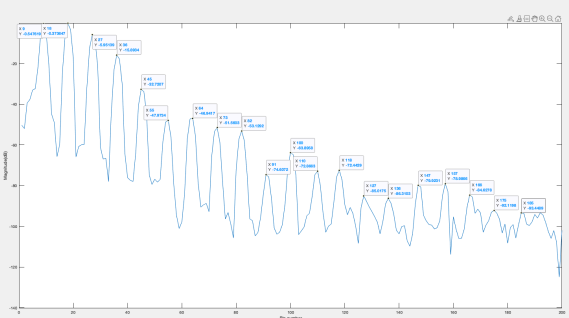

Figure 5. Snapshot of STFT at the time start of the signal (t = 3H, H = 2048 samples).

With the information we have of the STFT analysis, we can extract the first 20 harmonics (Fig. 6). Upon initial inspection, we can already see the plucking point effect at the start of the signal that resembles the magnitude response of a comb filter.

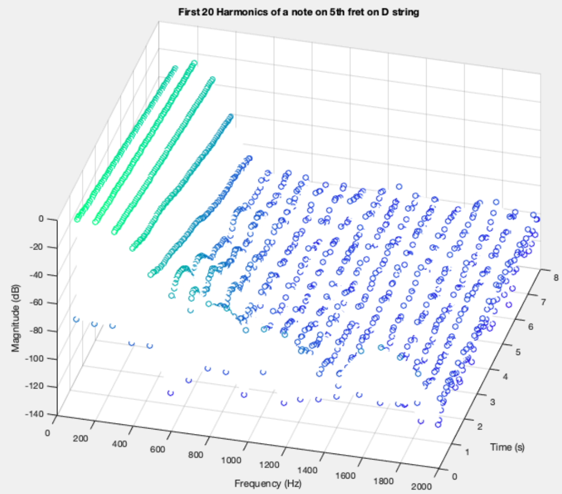

Figure 6. First 20 harmonics of the G2 note. Frequency axis is zoomed in from 0 to 2000Hz.

2 - Loop Filter Design

The loop filter is the core of this algorithm, it is the low-pass filter that provides frequency-dependent damping. The loop filter I used in this project is in the form of a minimal-phase all-pole low-pass filter (Välimäki et al., 1995), denoted by:

$$H_P(z) = g\frac{1 + a_1}{1 + a_1 z^{-1}}$$

with:

$$g \in (0,1), a_1 \in (-1,0)$$

To approximate the value of g and a_1, we need the value of the order-1 (linear) slope (decay rate) at each harmonic peak. Since the information we have is the STFT analysis of an actual recording, the slope will have to be linear regressed based on these values. In Matlab, these slope values, along with the y-intercept (order-0, initial gain, G_i) are approximated using the `polyfit()` function.

The filter gain (g) can be reliably determined by taking the average from the slopes of the first three harmonics:

$$g = 0.9967$$

To determine the best of a_1, we can use a weighted least-squared error function, as suggested in (Välimäki et al., 1995):

$$E = \sum_{k=1}^{20} W(G_k)\,[\,|H_P(\omega_k)| - G_k\,]^2$$

where,

$$W(G_k) = \frac{1}{1 - G_k}$$

However, the use of this weight function $W(G_k)$ did not yield promising result for a_1, possibly because of the paper only mentioned nylon and steel string guitar but not electric bass, or because of the hugely non-linear upper harmonics that are present in the recording. Hence, I decided to modify the weight function to:

$$E = \sum_{k=1}^{20} W'\,[\,|H_P(\omega_k)| - G_k\,]^2$$

where,

$$W' = G_i$$

This error function is now weighted towards the lower harmonics, since they are less affected by non-linearity and plucking point effect.

The finding of a_1 which results in the least error is carried out by an iteration loop: within each iteration, the value of a_1 is changed slightly, and the error function is reevaluated. The starting value is $a_1 = 0$. (Fig. 7) shows the evolution of the error that subtly decreases over 100 iterations and stabilizes eventually. The results of the filter coefficient a_1 is:

$$a_1 = -0.0063$$

The magnitude response of this filter is shown in (Fig. 8).

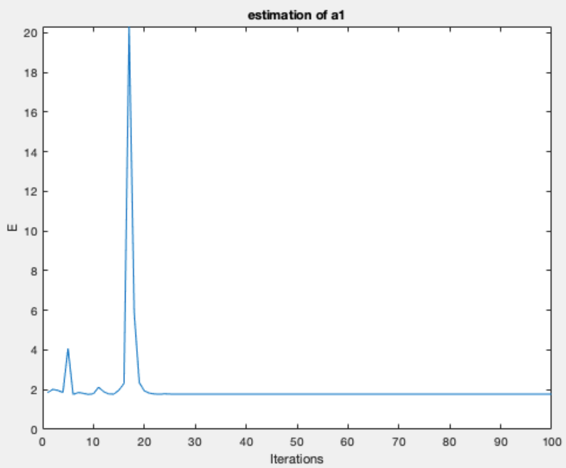

Figure 7. Estimation of a_1. Error function evaluation over 100 iterations.

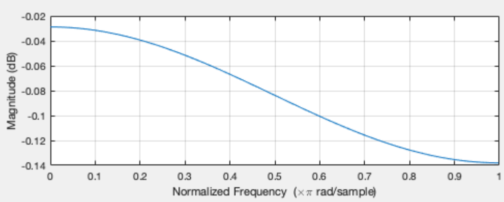

Figure 8. Magnitude response of the loop filter $H_P(z)$, with $g = 0.9967$, $a_1 = -0.0063$

The value of a_1 is extremely low, so the loop filter is a mild low pass filter, but the absence of it is definitely noticeable.

Beside the low-pass filter in the feedback loop, we also require a delay that has the value of:

$$L = \frac{f_s}{f_0}$$

The length of the delay we can implement with a circular buffer, without interpolation can only be an integer, as this becomes problematic, we can divide the delay length to two parts:

$$L = floor\left(\frac{f_s}{f_0}\right) + P_c$$

An integer part, which will be implemented using a buffer, and a fractional part (P_c) in the form of an all-pass filter:

$$H_F(z) = \frac{C + z^{-1}}{1 + C\, z^{-1}}$$

where the all-pass coefficient is calculated as,

$$C = \frac{1 - P_c}{1 + P_c}$$

At this point, we have finalized all the parts of the feedback loop (Fig. 9).

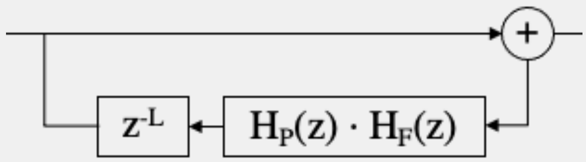

Figure 9. Finalized form of the feedback loop for the Karplus-Strong electric bass model

3 - Inverse Filtering

As mentioned earlier in this report, the inverse filtering technique is implemented to generate the excitation sequence as the input of this algorithm (Välimäki et al., 1995).

Since the system is linear, we can take an inverse of what represents the entire feedback loop:

$$S(z) = \frac{1}{1 - z^{-L} H_P(z) H_F(z)}$$

to:

$$S^{-1}(z) = \frac{1 + a_1 z^{-1} - g(1 + a_1 z^{-1}) z^{-L} H_F(z)}{1 + a_1 z^{-1}}$$

This inverse filter has relatively few co-efficients in play, but it has a high order that depends on the fundamental frequency of the recording of the real instrument. In our case, the order is 457. This means that we cannot use this in real-time synthesis, nor did I intend to. For real-time algorithmic implementation of the bass model, the in-verse filtering step is performed offline, to generate a small-size audio that is truncated to milliseconds, which is used as the input of the model.

For the recording of our one bass note only, the inverse filter is applied, and the result is truncated before the periodicity kicks in. [exert-2]

4 - Resynthesis

Before diving into resynthesizing the bass tone, we can apply a plucking-point comb filter to the input signal, which we extracted using the method of the previous section. The effect of plucking position has been thoroughly studied in several papers, including (Karjalainen et al., 1998).

In our implementation, we combine a delay with a low-pass filter for timbral control in a feedforward fashion, in the form shown in (Fig. 10).

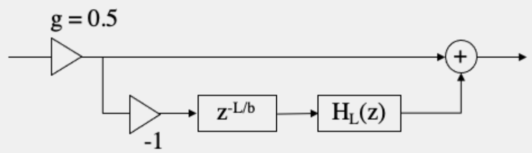

Figure 10. H_C Feedforward comb filter for plucking position. The delay in the feedforward part is L/b, where L is the length of the Karplus-Strong loop delay length in samples, and b is the distance from the plucking position to the bridge from 0 to 1.

This application of the comb filter can adequately simulate the plucking position, and the low-pass filter can polish the input timbre additionally, it has the transfer function of:

$$H_L = \frac{0.01 + 0.01z^{-1}}{1 - 0.98z^{-1}}$$

The magnitude response of this comb filter is shown in (Fig. 11).

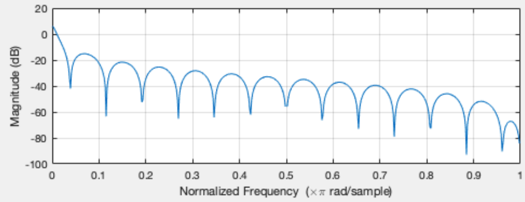

Figure 11. H_C Feedforward comb filter magnitude response.

The same low-pass filter is used at the end of this model, as a further varnish to the output timbre.

The finalized form of the model is shown in (Fig. 12).

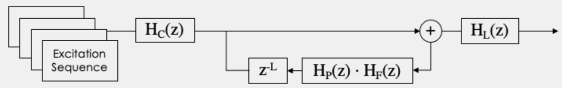

Figure 12. Finalized form of the extended Karplus-Strong algorithm implemented in this project.

The resynthesis is carried out exactly to what is described in the previous figure, the note-off effect is achieved by changing the filter gain of the loop filter suddenly to a low value right before the simulation ends. The result is pretty close to the original recording audibly(Fig. 13). [exert-3] However, there is a uniform buzz to the signal at higher fre-quency. My suspicion is that the truncation of the body response is flawed, and it might have some non-zero samples when it is wrapped around in the circular buffer.

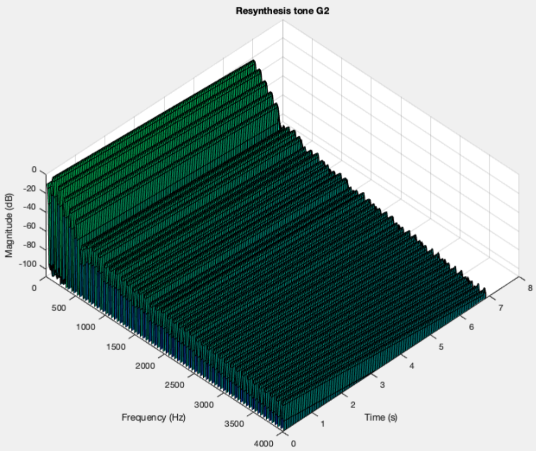

Figure 13. Resynthesized G2 electric bass tone.

After further experimentation, specifically using this one body response to generate tones that are of different pitch than the original recording, [exert-4] the high-frequency buzz becomes more audible. Additionally, the further the pitch is from the designated spot, the less natural the bass timbre is. This range of "sweet spot", outside which we can perceptually tell from an algorithm to a re-cording, is potentially due to the effect of the thumb of fingering hand pressing down on the neck. In future implementations, this problem can be solved by taking a lexicon of body responses at different finger positions (Fig. 14) at each string. We can construct a table exci-tation sequences that are made of these body responses. A string allocation program can ensure that the correct responses are used in case of real-time performance.

Figure 14. A scheme of taking finger position zones for a closer-to-natural timbre throughout the entire range of the instrument.

In future development and expansion to this project, I will probably look into implementation in the C++ Stk class, with support of real-time performance using RtMidi and RtAudio.

Appendix:

<u>Audio exerts:</u>

Exert-1: D_5.wav

Exert-2: body_response.wav

Exert-3: resynthesized_G2.wav

Exert-4: mario_kart_lick.wav

<u>Matlab Scripts:</u>

Script file name	Related sections
HDE_1.m	Analysis of the electric bass signal Loop filter design
KSLF_2.m	Loop filter design Resynthesis
INVF_3.m	Inverse filtering
RS_4.m	Resynthesis

Reference:

Jaffe, D. A., & Smith, J. O. (1983). Extensions of the Karplus-Strong Plucked-String Algorithm. *Computer Music Journal, 7*(2), 56-69. doi:Doi 10.2307/3680063

Karjalainen, M., Valimaki, V., & Tolonen, T. (1998). Plucked-string models: From the Karplus-Strong algorithm to digital waveguides and beyond. *Computer Music Journal, 22*(3), 17-32. doi:Doi 10.2307/3681155

Karplus, K., & Strong, A. (1983). Digital synthesis of plucked-string and drum timbres. *Computer Music Journal, 7*(2), 43-55.

Välimäki, V., Huopaniemi, J., Karjalainen, M., & Jánosy, Z. (1995). *Physical modeling of plucked string instruments with application to real-time sound synthesis.* Paper presented at the Audio Engineering Society Convention 98.