DSPFuzz
A Guitar Distortion Pedal Using The Stanford DSP Sheild

Woody Herman
Center For Computer Research In Music And Acoustics
Stanford University
Stanford, CA
Email: w.q.herman@gmail.com

***Abstract*—This project uses the Stanford DSP Sheild to implement a real time guitar distortion effects pedal. Presented here is a summary of the methods used, DSP concepts applied, and an analysis of what effects a distortion effect has on a signal.**

I. Introduction

In live performance and in recording sessions, guitarists frequently alter the tone of their instrument using effects pedals, commonly referred to as stompboxes since the user engages the pedal with a foot swtich. One of the most widely used effects is the distortion, or overdrive, effect. This clips the signal of the guitar creating a unique distorted sound that can be heard on nearly every rock recording made since the 1950s. In its earliest from, distortion was implemented by driving the transistor or vacuum tube amplifiers used into cutoff. With the new prominence of digital recording and effects, many of the classic analog stompboxes have been implemented digitally with great effect. Distortion, however, remains firmly rooted in the analog domain. Because of the many nonlinearities present in these distortion effects, it has been hard to create pleasent sounding digital alternatives. Despite these issues, and because of the problems inherent in maintaining and designing analog equipment, developing quality digital versions would certainly be of great value. This project implements a simple and commonly used method for recreating distortion effects digitally.

II. Overview of The Method

A simple, commonly used technique for replicating the sound created by analog distortion pedals is to apply a pre-distortion filter, for controlling the input tone, follwed by up-sampling, applying the non-linearity, down sampling and finally another filter to control output tone. This technique is widley is used in industry and, if the tone filters are tuned properly and if the non-linearity is carefully chosen, can yeild sonically pleasing results. This method is primarily chosen for its compuational simplicity, many of the nonlinear functions (often called waveshaping functions) are polynomials relating input amplitude to output amplitude, and flexiblity to create a variety of sounds.

A. Waveshaping Functions

Choosing the waveshaping function that will clip the signal is a very subjective art, although there are several that have become very popular throughout the years. Generally, they all follow the same format. That is, they are non linear functions that relate input amplitude x to output amplitude y. Equations (1) and (2) show two popular waveshaping functions.

$$y = \frac{x}{1 + |x|} \tag{1}$$

$$y = x + \frac{x^3}{3} \tag{2}$$

The functions above are written as they would be used for implementation in floating point systems. As will be shown later, fixed point arithmetic introdcues several new problems that make directly implementing these functions difficult if not impossible.

B. Tone Shaping Filters

Guitarists, and musicians in general, constantly talk about the tone of their instrument. The tone of an instrument in due in large part to the frequency response of that instrument. Because an electric guitar turns the vibration of its strings into electricity, we can apply any filter, analog or digital, that we might use in other applications. By boosting high frequencies we can make the guitar sound brighter, by boosting lower frequencies we can make the guitar sound fuller, or perhaps darker. What constitutes a good tone is highly subjective, but regardless of the result it is desireable to have some sort of filtering of the guitar signal in distortion effects. By applying a filter to the input and output signal of the effect, we can create custom tones or attempt to match the tone of famous analog effects.

III. Relation to Course Topics

Although the idea behind distortion is relatively simple, performing the operations digitally requires the use of a number of common and important digital signal processing theories. These include upsampling, digital filter design, analysis of our signal in the frequency domain, and fixed point arithmetic.

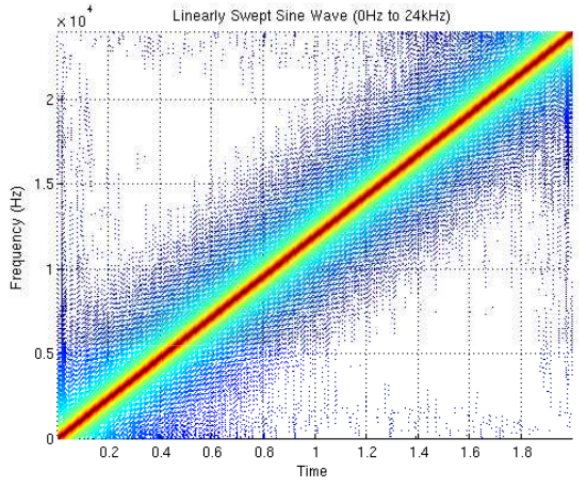

Fig. 1. Spectrogram of a chirp signal starting at dc and ending at $24kHz$

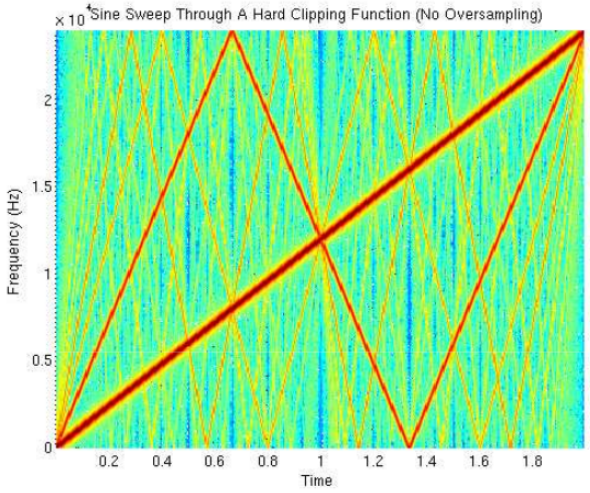

Fig. 2. Spectrogram of a chirp signal passed through a hard clipping function with no upsampling

A. *Upsampling*

As mentioned previously, implementing digital distortion requires the use of upsampling. This is because non linear functions, especially ones that clip the signal, introduce harmonics of the original frequency content. In distortion effects, when high gain is used, these harmonics can have audible amplitudes up to the 4th or 5th harmonic (that is 4 or 5 times the fundamental frequency). Because of this, we run the risk of our signal aliasing if a lot of distortion is applied. To counteract this aliasing, we upsample our signal to a much higher rate (8 and 16 are common factors used in industry) apply the non linearity and then down sample to the original rate. When used in conjunction with an interpolating and decimating filter, which can be the same filter since we are returning to the same sampling rate we started with, this blocks the aliased components while allowing for a natural expansion of harmonics in the audible range. This is important since the harmonics introduced by these clipping functions are an important part of the overall tone they give the guitar.

To illustrate this point, it is helpful to analyze the effects of clipping functions in the frequency domain using the DFT. Fig 1 shows a spectrogram of a chirp signal, a sine wave whose frequency increases linearly with time, that starts at dc and ends at the original nyquist frequency, $24kHz$.

If we pass this chrip signal through a clipping function, then the effects of aliasing become apparent. Below is a figure showing the same chirp signal passed through a hard clipping function, where the signal is abruptly flattened at the maximum amplitude.

Fig. 2 is an exaggerated version of what might actually occur because the frequecny response of an electric guitar tends to roll off at higher frequencies. Nevertheless, aliasing is still audible when upsampling is not used. If upsampling is used in conjunction with an interpolating and decimating filter then we can remove the aliased components while preserving the desired harmonics that occur naturally in analog distortion devices.

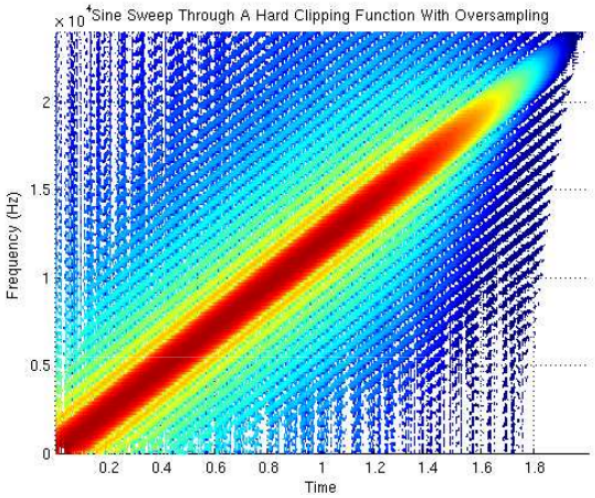

Fig. 3. Spectrogram of a chirp signal that has been upsampled by a factor of 8, filtered, clipped, filtered, and downsampled back to the original rate

Fig. 3 shows the same chirp signal passed through the same clipping function, but this time we have upsampled by a factor of 8 (using an interpolating filter whose design is described in subsequent sections), then applied the clipping function, then filtered the clipped signal with the same filter and downsampled back to the original rate. As can be seen in the plot, the aliased components have been removed but, as is desired, harmonics have been introduced to the signal. These harmonics are represented by the parallel bands in the spectrogram above and below the chirp signal. As mentioned before, these harmonics are very desireable since they help shape the tone of the disorted guitar signal. It should be noted that the magnitude of the chirp signal decreases as we approach nyquist. This, as will be shown later, is due to the fact that the filter used in upsampling and downsampling has a cutoff frequency below half the sampling rate.

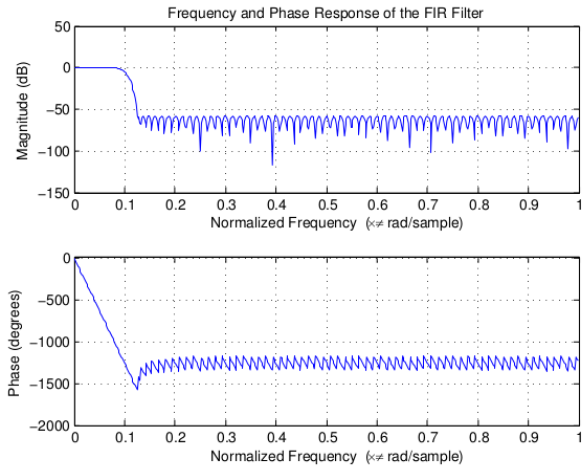

Fig. 4. Frequency response of the low pass filter used for up sampling and down sampling.

B. Filter Design

In order to properly implement the upsampling and downsampling required by digital distortion effects, we must design an interpolating filter that accomplishes several goals. First, as is always the case, it must remove the images of our spectrum created by sampling. Second, it must prevent aliasing caused by the non linear waveshaping function. Third, it must preserve as much of the original signal and added harmonics as possible. Lastly, it must be computationally efficient enough to be run in real time.

To accomplish the design criteria of the filter, the Parks-McLellan algorithm was used via MATLAB's $firpm()$ function. Although a similar inerpolation and decimation filter could be acheived with a far lower order IIR filter, an FIR option is attractive for several reasons. Because the PM designed FIR filter would be linear phase, we know that no phase distortion would occur. Furthermore, the DSP Sheild has built-in functions in the DSP library that efficiently up sample and down sample using FIR filters. The library implementation of these functions would almost certainly be faster than anything written.

Figure 4 shows the frequency response of a 141 order Parks-McLellan designed filter. Because electric guitars typically have frequency responses that begin to roll off at a little over $10kHz$ we are able to have the filter pass band end at $15kHz$ with no real audible effects. This allows us to obtain a stop band gain of $-60dB$ that begins at $24kHz$ which is $\frac{1}{8}pi$ radians at the increased sampling rate. $-60dB$ is a value commonly used in audio to represent when a signal has become inaudible. Thus, it is an appropriate stop band gain to acheive to successfully filter out any components that would alias at our original sampling rate.

The filter in fig. 4 is used in the program for both the upsampling and downsampling functions. This is possible because we are returning to the same sampling rate at which we began, thus the filter stop band must begin at the same frequency in both cases.

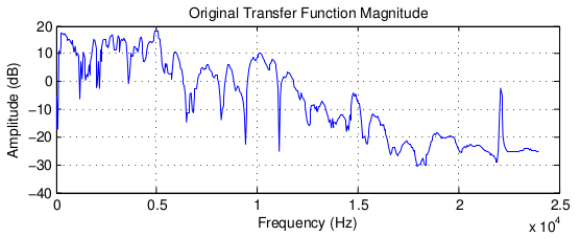

Fig. 5. Frequency response of a Vox ac30 guitar amplifier.

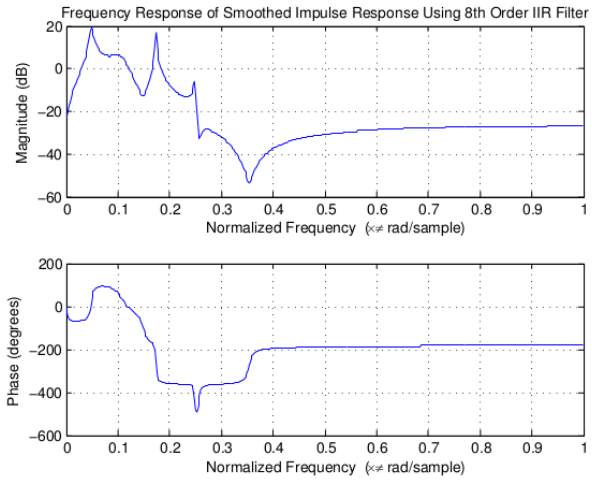

Fig. 6. 8th order approximation of the ac30 frequency response.

In addition to the filtering required by upsampling and downsampling, it is also common to add filters that will change the tone of the guitar. The design of these filters, much like the design of the waveshaping functions, is very subjective. One common trick is to use impulse response measurements to try and mimic the tone of popular, vintage guitar amplifiers. Here, we can make use of the DSP Sheild's hardware accelerated filters to alter the tone of the guitar without taking too much of a toll on the computational efficiency of the effect. Impulse responses of many of the most popular guitar amplifiers are freely available online. By taking the DFT of the impulse response of the Vox ac30 amplifier, a very popular vintage vaccuum tube amplifier, we can obtain the frequency response shown in fig. 5.

Using MATLAB's $invfreqz()$ function, we can model the frequency response with a desired filter order. Using an eigth order filter, we can get the frequency and phase response shown in fig. 6.

Although this may not be the most accurate model, it still encompasses some of the major features (blocking dc, boost in the lower-middle frequencies, and a tapering off in the higher frequenices), and should provide an interesting filter to shape the tone of the pedal. By breaking the 8th order filter into second order sections and quantizing the coefficients, we can implement this filter using the DSP Sheild's hardware

accelerated filters on the ADC and DAC.

C. Fixed Point Waveshaping Functions

Waveshaping functions are often used to implement digital distortion because of their simplicity and computational efficiency. This line of thought is very true in floating point systems. However, as we have seen throughout the quarter, fixed point architectures tend to make things a little more difficult. Two observations can be made by examining equations (1) and (2). We can note that (1) requires a division at every sample to compute. Division is an expensive operation, especially in a fixed point system where the division is happening at an oversampled rate. In fact, division is such an expensive operation to perform that running the program using the waveshaping equation in (1) is impossible.

Equation (2) presents its own challenges. As it is written, (2) clips a signal in the range $-1 \leq x \leq 1$. However, because we are operating in a fixed point system, we want to clip a signal in the range $-32768 \leq x \leq 32768$. To accomplish this we must re-arrange (2) to the following:

$$y = x + \frac{x}{32768^2 * 3} \tag{3}$$

Equation (3) also requires a division but, because it is a constant, it could be precomputed and used as a multiplication. However, because the number in the denominator is so large, the scaling factor required in the equation would be a number that is smaller than the smallest possible number that can be represented with 15 bits, and it is very close to the smallest number that can be represented with 31 bits. Therefore, implementing this with 15 bits as the base number of bits would be very difficult if not impossible. The way to get around this is to use a lookup table, a set of points sampled from the curve created by eq. (3) and then joined by together by linear interpolation. If enough points are used, then the lookup table can closely approximate the curve generated by eq. (3). The use of lookup tables to approximate nonlinear curves is common among digital distoriton effects, and it is the process that is used here.

IV. Proposed vs Original Timeline

The original proposed timeline was largely followed, although it was accelerated slightly. Some components, as per the feedback, were left until the end after more important components had been implemented.

A. Weeks One and Two

Weeks one and two, in the original proposal, were designated as basic set up weeks. They were intended to establish basic connections to the board for the guitar. This involved passing a guitar signal, via a $\frac{1}{4}$ inch. cable commonly used for guitars, to the DSP Sheild and then out to a guitar amplifier. Another connection that would be established in the first two weeks would be to connect the potentiometers that would control the volume and amount of distortion. Additionally, it was intended that a box to house the DSP Sheild would be constructed in the first two weeks.

As suggested by the comments, several of the goals of weeks one and two were shifted to the end of the project. Building a housing for the pedal, and attaching the pentiometers were determined to be additional components but not essential to the main objective of the project. Therefore, during week one, the guitar was successfully passed through the dsp sheild and out to an amplifier. Upsampling and downsampling, including the design of the filter used in each case, was also accomplished in the first two weeks. Towards the end of week two, work was begun to implement the non linear function, although this was not successfully implemented until the beginning of week 3.

B. Weeks Three and Four

Originally, weeks three and four would be used for designing tone filters, adding upsampling and implementing the non linearity. Because it was decided that these features were more important than some of the ones originally intended to be implemented in weeks one and two (potentiometers and case for the pedal); weeks three and four were left for the optional components.

At the beginning of week three, the non linearity was finished, thus completeing the bare essentials of the distortion effect. The tone filters were designed and implemented using the hardware accelerators in this week. This left the rest of week three and week four to implement the optional components, such as constructing the case and attatching the potentiometers.

V. Conclusions and Future Work

Digital audio effects are an ever groing presence in modern music making. In order to make commercially viable, portable products (ones that do not need to be run on a full pc), fixed point dsp processors will continue to be used for the forseeable future. Having an affordable, well documented, easily programmable learning tool to implement these effects is invaluable. In addition, the DSP Sheild can be used as a prototyping tool for novel effects. I plan on keeping the DSP Sheild to use as both a learning tool and as a tool to implement original effects in an easy to program format. Becuase I have already built the basic structure of a guitar pedal, it is a simple matter of replacing the code on the sim card of the DSP Sheild to test out and implement a variety of effects.