ECS7012 Music and Audio Programming

Final Project: Vintage Effect

xxxxxxx
May 2023

Abstract

This report represents a vintage tape recorder and environment digital effects implemented in the Bela platform using C++. This vintage effect is included a Noise Generator to add vinyl or other noise sounds, a wobble effect to make the audio tremble, three types of distortions, a sampler to emulate four types of old-school recorders and a reverb to add a sense of space.

1 Introduction

The vintage effect in digital audio processing has garnered significant interest due to its ability to recreate the nostalgic sound characteristics of analogue audio equipment. This effect aims to capture the essence of past eras by simulating the unique tonal qualities and imperfections associated with vintage gear. In recent years, the vintage effect has gained popularity in genres like lo-fi hip-hop and retro music. Lo-fi hip-hop, known for its low-fidelity aesthetics, heavily samples from vinyl records and incorporates vintage effect processing to create a nostalgic and laid-back atmosphere. Vinyl crackles, tape hiss, and warbled effects are used to transport listeners back in time and evoke a sense of nostalgia and authenticity.

As mentioned above the vintage effect is currently mostly used in the field of music production. Therefore, most of the previous research on the vintage effect has focused on industrial applications. There are so many websites to teach producers how to use many separate plug-ins for creating vintage sound effects. Such as LEA and iZotope, which always post many tutorials with software and digital signal processing (DSP) on their website. But these plugins are all separate, so my idea is to combine multiple independent effect algorithms into one software for vintage effect implementation. In addition to the usual processing effects that make the sound vintage, I also added Schroeder's ReverberatorsSchroeder and Logan (1961) to add a sense of space.

2 Design

2.1 Noise Generator

The addition of a noise generator in vintage effects enhances the authenticity and nostalgic appeal of the visuals or audio. It simulates the characteristics of old cameras or televisions, such as scan lines and grainy noise, providing a realistic touch and evoking a sense of bygone eras. This technique creates a vintage ambience, triggering nostalgia and adding a unique artistic element. Additionally, the noise generator can mask the modern feel of the original material and emphasize key elements.

The sequence of noise generators is designed to sum directly to the input audio, with Amplitude(a) representing the noise level(using "Volume" to represent in code):

$$y[n] = x[n] + aN[n]$$

In real-time, I used one of the three noise files as N[n], using a two-dimensional array to represent each sample, with the first dimension indicating the file's index and the second dimension indicating the sample to which the current pointer points. When the pointer reaches the end of the array and makes pointer=0.

2.2 Filters

There are four filters used in this project: the low-pass filter, the high-pass filter, the all-pass filter and the comb filter. The latter two are used in reverb, this section focuses on high-pass and low-pass filter. The low-pass and high-pass filter is two second-order filters. The sequence is expressed as:

$$y(n) = b_0x(n) + b_1x(n-1) + b_2x(n-2) - a1_y(n-1) - a2_y(n-2) \tag{1}$$

$$y(n) = b_0x(n) - b_1x(n-1) + b_2x(n-2) - a1_y(n-1) + a2_y(n-2) \tag{2}$$

the (1) is the low-pass sequence and the (2) is the high-pass sequence.

The two filter algorithms use two single-sided Linkwitz-Riley. The design equations are as follows Pirkle (2013)(the left is low-pass and right is high-pass):

$$\begin{aligned}
\theta_c &= \pi f_c/f_s & \theta_c &= \pi f_c/f_s \\
\Omega_c &= \pi f_c & \Omega_c &= \pi f_c \\
\kappa &= \frac{\Omega_c}{\tan(\theta_c)} & \kappa &= \frac{\Omega_c}{\tan(\theta_c)} \\
\delta &= \kappa^2 + \Omega_c^2 + 2\kappa\Omega_c & \delta &= \kappa^2 + \Omega_c^2 + 2\kappa\Omega_c \\
a_0 &= \frac{\Omega_c^2}{\delta} & a_0 &= \frac{\kappa^2}{\delta} \\
a_1 &= 2\frac{\Omega_c^2}{\delta} & a_1 &= \frac{-2\kappa^2}{\delta} \\
a_2 &= \frac{\Omega_c^2}{\delta} & a_2 &= \frac{\kappa^2}{\delta} \\
b_1 &= \frac{-2\kappa^2 + 2\Omega_c^2}{\delta} & b_1 &= \frac{-2\kappa^2 + 2\Omega_c^2}{\delta} \\
b_2 &= \frac{-2\kappa\Omega_c + \kappa^2 + \Omega_c^2}{\delta} & b_2 &= \frac{-2\kappa\Omega_c + \kappa^2 + \Omega_c^2}{\delta}
\end{aligned}$$

The use of a single-sided Linkwitz-Riley filter offers several advantages over a Butterworth filter. Firstly, the Linkwitz-Riley filter provides a steeper roll-off slope, resulting in a more precise and accurate frequency response. This characteristic ensures better separation between different frequency bands, minimizing unwanted interference and ensuring a cleaner signal. Secondly, the Linkwitz-Riley filter maintains a flat frequency response in the crossover region, eliminating any phase shifts or frequency anomalies that may occur with Butterworth filters. In the later reverb section, both high-pass and low-pass filters will be used. This helps to achieve a more seamless transition between the low and high-frequency components, ensuring a more natural and balanced sound output. Overall, the

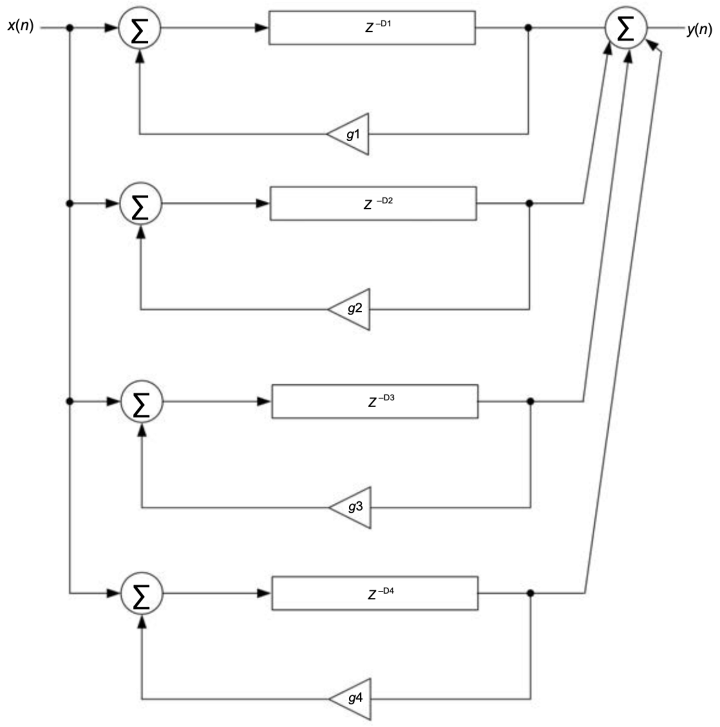


Figure 1: Four comb filters in parallel.

single-sided Linkwitz-Riley filter offers improved precision, better frequency response, and superior crossover performance compared to a Butterworth filter.

In the C++ implementation, I have packaged the two filters into classes, which have a similar structure. There are two important functions:

- float getFilteredSample(float in, float frequency): Processing of specified cut-off frequency samples with filters
- calculateCoefficients(float frequency, float Q): Calculate second-order Linkwitz–Riley's coefficients

2.3 Reverb

The reverb used in the project is Schroeder's Reverberator. There are two delaying filters has been combined to implement the reverb: The comb filter reverberator and the delaying all-pass filter reverberator.

2.3.1 Comb Filter Reverbrater

In Schroeder's Reverberator, there are four comb filters in parallel.Figure 1 shows this structure. Each has its own delay time (D1–D4) and gain (g1–g4)Pirkle (2013).

The difference equation and transfer function for the comb filter:

$$y(n) = x(n - D) + gy(n - D)$$

and

$$H(z) = \frac{z^{-D}}{1 - gz^{-D}}$$

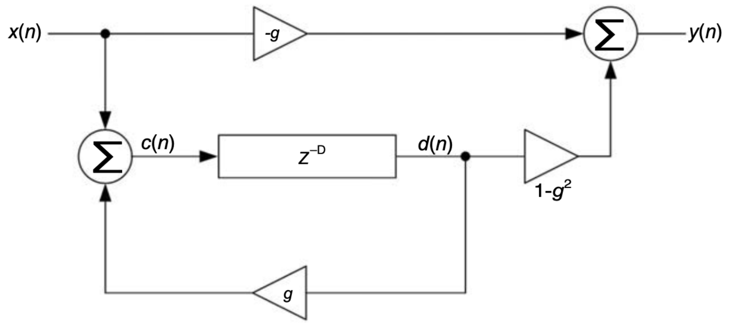


Figure 2: APF reverberator unit

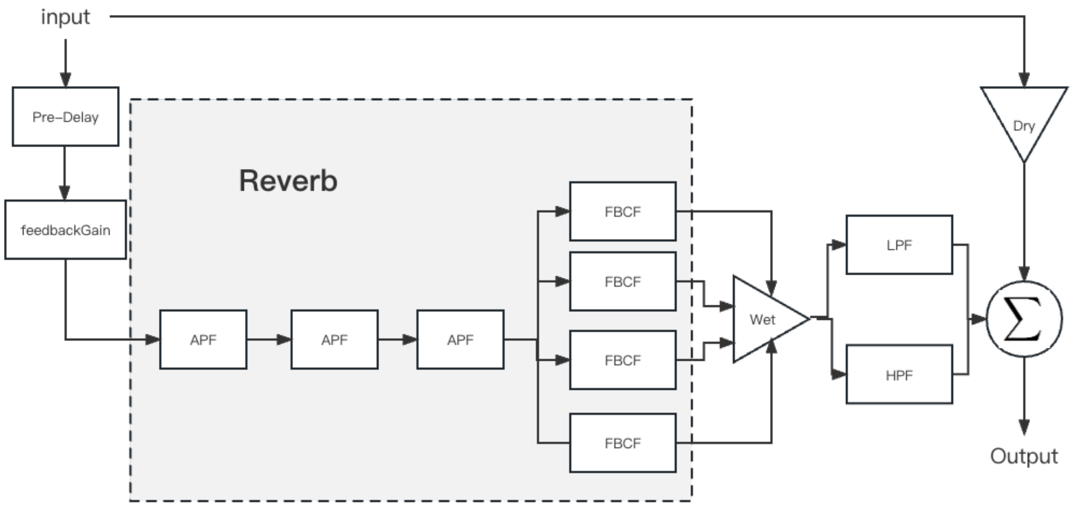


Figure 3: Model of the reverb algorithm

where

$$D = the\ delay\ length$$

2.3.2 The Delaying All-pass Filter Reverberator

Schroeder also proposed the delaying all-pass filter (APF) as a reverberator unit. The sequence function of the delaying all-pass filter can be obtained from the equation substitution of Pirkle (2013):

$$y[n] = -gx[n] + x(n-d) + gy(n-D);$$

2.3.3 Implementation

The reverberation algorithm diagram in this section can be represented as Figure(3): Firstly, we used the class *Reverb* to manage the whole reverb system and created the *process*() function to perform all the operations. From *render.cpp*, the whole system is a black box, just input the samples currently being processed and output the reverb added. Secondly, to obtain the accurate number of delay samples, I create a *delayLine* vector as a buffer to store samples before processing samples. Meanwhile, the size of *delayLine* depends on the pre-delay time. It can be calculated as:

$$delay\ line\ size = (int)(samplerate\ * pre - delay\ time\ in\ seconds)$$

Then $feedbackGain$ is multiplied by the sample after the delay process. The third step is the core of the reverb, refer to Smith (2010) parameters for setting g and z for all-pass filters and comb filters. This is usually the end of the reverb processing. I have added a low-pass and high-pass filter as mentioned in Section 2.2 to the back of the reverb-processed wet sample in order to make the output reverberate better.

2.4 Wobble

The addition of wobble effect in vintage effects provides several benefits. Firstly, it adds an element of instability and imperfection, mimicking the characteristics of vintage media formats. This can create a sense of nostalgia and authenticity, enhancing the overall vintage aesthetic. Secondly, the wobble effect can introduce subtle variations in pitch and modulation, adding warmth and character to the audio or visual content. It can recreate the subtle fluctuations and inconsistencies found in older analog recordings or film reels, evoking a sense of charm and uniqueness. Lastly, the wobble effect can serve as a creative tool to highlight certain elements or evoke specific emotions, as it can create an immersive and atmospheric experience.

The wobble effect is processed as follows:

- The original audio signal is through a certain low-pass filter to obtain a low-frequency signal.
- A sine wave is then superimposed on the low-frequency signal to achieve a dithering effect on the low-frequency signal.
- The low-frequency signal is mixed with the original audio signal to obtain the final Wobble effect audio signal.

In the wobble effect, there are two variables *depth* and *rate*. *depth* indicates the depth or amplitude of the wobble effect and *rate* indicates the oscillation frequency.

2.5 Distortion

There are three distortion algorithms referenced from Reiss and Mcpherson (2015). The first is a hard clipping distortion:

$$f(x) = \begin{cases} 1, & x > 1 \\ -1, & x < -1 \\ x, & else \end{cases}$$

The next are two soft clipping distortion:

$$f(x) = \begin{cases} 2x, & 0 \leq x < 1/3 \\ 1 - (2 - 3x)^2/3, & 1/3 \leq x < 2/3 \\ 1, & 2/3 \leq x \leq 1 \end{cases}$$

$$f(x) = \begin{cases} 1 - e^{-x}, & x > 0 \\ -1 + e^{x}, & x < 0 \end{cases}$$

In the vintage effect, the algorithm described above is used to emulate the distortion caused by poor head contact when recording on a tape machine. The playback of a vintage tape machine is often accompanied by a high-frequency decay, so a low-pass filter is used for this purpose. The parameter seen in the user interface is *tone*.

2.6 Emulate Sampler

Every digital audio system possesses a sample rate and a bit depth, which collectively establish the level of detail, often referred to as the "resolution"LEA. The Bela I/O bit depth is 16 bit and the sample rate is 44.1 kHz bel, which is too high a resolution for a vintage sampler. In this project, I have chosen several old-school samplers to emulate by reducing the bit depth and sample rate. Their specific data are as follows:

Sampler Name	Bit Depth & Sample Rate
Fairlight CMI	16bit, 24kHz
E-mu SP-1200	2bit, 26kHz
Akai's legendary MPC60	12bit, 4kKhz
Linn LM-1's	8bit, 28Hz

The bit reduction algorithms as follows:

$$y[n] = ceil(ampx[n]) * (1/amp)$$

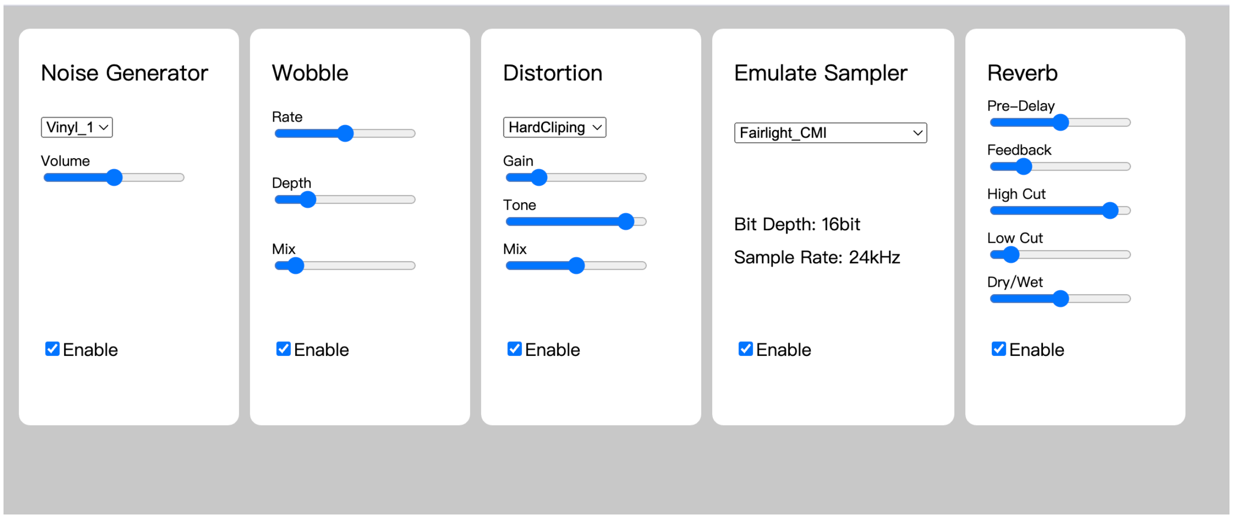
Downsampling is done by calculating the ratio of the sample rate before and after the change to obtain the ratio before and after the audio frame, thus calculating the number of interval samples missing after downsampling. In this algorithm, I used linear interpolation hair for the final non-integer finger to take the value. Also, the downsampling algorithm needs to take into account the aliasing. Although the aliasing sometimes produces good results in vintage effects, when testing I found that if I kept the aliasing it would make the audio sound bad, so I added a filter with a 10kHz cut-off frequency (less than the sample rate/2) to cut the aliasing before the downsampling algorithm.

2.7 User Interface

To use the checkbox and option boxes, I chose to use p5.js for the GUI creation, as shown in Figure(4). The values are passed using the protocol in Bela. The most important feature of the GUI is that the user can choose whether or not to use the effect via the checkbox.

3 Evaluation and Discussion

The audio quality tests, mainly through the ears. I used a 3.0mm to 6.35mm audio cable to plug into the sound interface and monitor headphones. This gave a more accurate effect feedback than using headphones directly into Bela's audio output. The evaluation was


Noise Generator
Vinyl_1
Volume
Enable

Wobble
Rate
Depth
Mix
Enable

Distortion
HardCliping
Gain
Tone
Mix
Enable

Emulate Sampler
Fairlight_CMI
Bit Depth: 16bit
Sample Rate: 24kHz
Enable

Reverb
Pre-Delay
Feedback
High Cut
Low Cut
Dry/Wet
Enable

Figure 4: GUI with default value

carried out using three types of audio: dry acapella vocals, dry instruments(mainly guitar) and pop songs (mixed vocals and instruments).

Based on the evaluation results, the vintage effect demonstrated satisfactory performance across all audio types. However, different modules exhibited varying levels of effectiveness in different scenarios. The noise generator seamlessly integrated with instrumental music, yielding favourable outcomes. Conversely, when applied to acapella vocals, the result consistently produced a poor mix of noise and vocals. The wobble effect showed a similar performance to the noise generator.

Distortion effects yielded good results in all three audio types, particularly excelling in pop music. Unfortunately, the vintage samplers did not perform well. Although the effect was noticeable when the *Enable* button was toggled in isolation, it weakened significantly when switching between different samplers. Furthermore, attempts to increase the volume of the sound card introduced unwanted white noise, which affected monitoring and rendered the endeavour unsuccessful. A comparison was conducted by rendering the audio using Reaper at the specified sampler bit depth and sample rate, revealing a noticeable difference in sound quality. This comparison underscored the subpar performance of the sampler module. A subsequent comparison could potentially involve applying a downsampling algorithm in the frequency domain for further evaluation.

Among all the effects, the reverb exhibited the best performance, but there is still substantial room for improvement. For instance, the current room size is insufficient, and the vintage effect on the listening experience is minimal. Comparatively, the industrial Valhalla VintageVerb reverb, which utilizes convolution reverb, stands out as one of the best options for achieving vintage effects. Convolution reverb enhances the realism of the effect in contrast to the physical modelling reverb used in this study. Exploring convolution reverb further is a worthwhile avenue to pursue.

Finally, in the GUI, my earlier idea was that the user could drag each module to change the order of the audio flow and thus the final output. However, in practice, I did not implement this feature in the end.

4 Conclusion

In conclusion, the digital vintage audio effect developed as part of this independent project has shown promising results. The effect includes a noise generator, wobble effect, three types of distortion, a sampler emulating old-school recorders, and a reverb effect.

Through evaluation and testing, the vintage effect demonstrated satisfactory performance across different audio types. The noise generator and wobble effect effectively added a nostalgic and authentic touch to instrumental music. However, their performance was less satisfactory when applied to acapella vocals. The distortion effects yielded good results in all audio types, particularly excelling in pop music. The vintage sampler module, however, did not meet expectations and showed noticeable differences in sound quality compared to actual vintage samplers.

The reverb effect stood out as the most successful module, offering immersive and atmospheric soundscapes. However, there is still room for improvement in terms of room size and overall vintage effect. Comparisons with industry-standard reverb algorithms, such as convolution reverb, could provide valuable insights for future enhancements.

In summary, this independent project has successfully implemented a digital vintage audio effect, offering new creative possibilities for audio production and music composition. While certain modules showed excellent performance, further refinement is needed for the sampler module and the reverb effect to enhance the overall vintage audio experience. Continued exploration and improvement in these areas can contribute to the advancement of digital audio technology.

References

Blog - loopcloud. URL `https://www.loopcloud.com/cloud/blog/`.

Bela hardware - the bela knowledge base. URL `https://learn.bela.io/using-bela/about-bela/bela-hardware/#sampling-rates-and-bit-depth`.

W. Pirkle. *Designing Audio Effect Plug-ins in C++ with Digital Audio Signal Processing Theory*. Focal Press, 2013. ISBN 9780240825151. URL `https://books.google.co.uk/books?id=v0ulUYdhgXYC`.

J. D. Reiss and A. P. Mcpherson. *Audio effects theory, implementation and application.* Boca Raton London New York Crc Press, Taylor Francis Group, 2015.

M. R. Schroeder and B. F. Logan. " colorless" artificial reverberation. *IRE Transactions on Audio*, (6):209–214, 1961.

J. Smith. Schroeder reverberators, 2010. URL `https://ccrma.stanford.edu/~jos/pasp/Schroeder_Reverberators.html`.