APPLYING AUDIO EFFECTS TO IMAGE

Hounsu Kim **Jongsoo Kim**
Graduate School of Culture Technology, KAIST , South Korea
{hanshounsu, jongsoo.kim}@kaist.ac.kr

ABSTRACT

Audio effects are applied to images in a near real-time manner by exploiting the JUCE framework and OpenCV library. GUI is constructed where the user can first choose an image from the local directory then apply various audio effects (including compressor, delay, reverb, bell-shaped filter) by manipulating the visual sliders.

1. INTRODUCTION

Utilizing a tool designed for one type of data (audio, for example) to another form of data (e.g. images) is an interesting field, both in aesthetic and engineering perspectives. This could expand to multimodality in the way we perceive art, where we can not only listen to music but simultaneously interplay with other senses such as our eyes, thus amplifying our experience. In this project we defined various methods in applying audio effects to images, by adapting the engineering principle of audio effects to the image domain. JUCE, an open-source cross-platform C++ application framework, is used mainly for GUI purposes, and OpenCV library is also used in order to manipulate the image data.

2. MAPPING AUDIO DOMAIN TO IMAGE DOMAIN

2.1 Compressor

In the case of compressor, image pixel domain is mapped to the time domain of audio, and intensity value of each image pixel is mapped to each of the audio sample's amplitude. Let $I_{x,y}$ and $I'_{x,y}$ be the intensity values of (x, y) of input image and output image, respectively, and I_t denote the threshold value. Then, we can represent the $I'_{x,y}$ as below:

$$I'_{x,y} = \begin{cases} I_t + \frac{I_{x,y} - I_t}{ratio} & \text{if } I_{x,y} > I_t \\ I_{x,y} & \text{if } I_{x,y} \le I_t \end{cases} \tag{1}$$

and $ratio$ is how much to compress when the $I_{x,y}$ is greater than the I_t.

2.2 2D Fourier Transform and Convolution

2-Dimensional Fourier transform and convolution are needed to implement filter, delay, and reverb in image domain. Unlike audio signals, image signals are 2-dimensional signals. Therefore, the Fourier transform $F[u, v]$ of the input signal $f[x, y]$ can be represented as follows:

$$F[u, v] = \mathcal{F}\{f[x, y]\} = \sum_{x=0}^{W-1} \sum_{y=0}^{H-1} f[x, y] e^{-j2\pi\left(\frac{ux}{W} + \frac{vy}{H}\right)} \tag{2}$$

And the convolution theorem that we studied can be also applied in image domain. The convolution theorem of 1-dimensional signal can be represented as below:

$$g[t] = \mathcal{F}^{-1}\{G[w]\} = \mathcal{F}^{-1}\{F[w] \cdot H[w]\} \tag{3}$$

Through the modification of (3), we can obtain the output image $g[x, y]$ with the transfer function $H[u, v]$ applied as follow:

$$g[x, y] = \mathcal{F}^{-1}\{G[u, v]\} = \mathcal{F}^{-1}\{F[u, v] \cdot H[u, v]\} \tag{4}$$

As Figure 1, the method to obtain $H[u, v]$ from audio domain proceeds in the following order. 1) Rotate $H[w]$ relative to the amplitude axis. 2) Obtain the frequency-shifted image from the rotated image.

2.3 Filter

For the filter, we chose the bell-shaped filter (digitized equalizer). Thus we manipulate the following 3 parameters: (f_c = cut-off frequency), (Q = Q value), and ($Gain$ = gain value). Let z denote e^{jw}. Then, in audio domain, we can represent the transfer function as below:

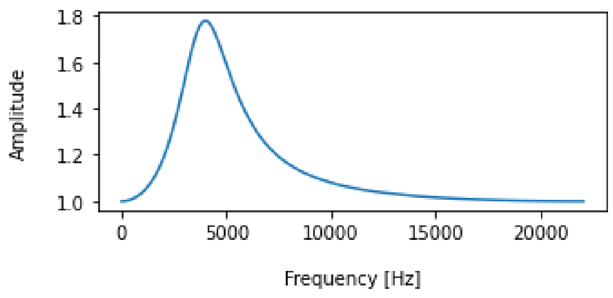
$$H(z) = \frac{(1 + \alpha \cdot A) - 2\cos\theta z^{-1} + (1 - \alpha \cdot A)z^{-2}}{(1 + \alpha/A) - 2\cos\theta z^{-1} + (1 - \alpha/A)z^{-2}} \tag{5}$$

where θ, α, and A are given as follows (the f_s is the sampling rate):

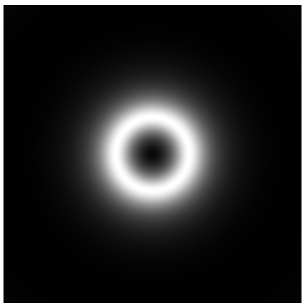
$$\theta = 2\pi \frac{f_c}{f_s} \tag{6}$$

$$\alpha = \frac{\sin\theta}{2Q} \tag{7}$$

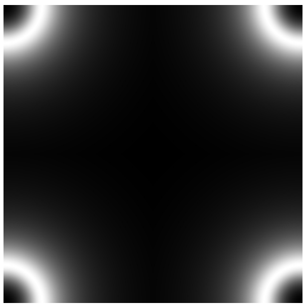
$$A = 10^{\left(\frac{Gain(dB)}{40}\right)} \tag{8}$$



(a) $H(w)$



(b) $H(u,v)$



(c) Shifted $H(u,v)$

Figure 1: (a) Example of $H(w)$. (b) Roation of (a) relative to the amplitude axis. (c) Shifted result image of (b).

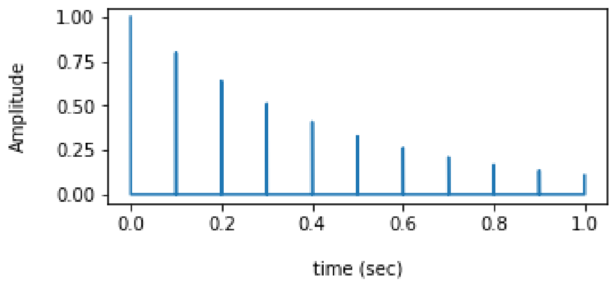


Figure 2: 1-Dimensional impulse response of delay (feedback value = 0.8, delay time = 0.1 sec)

2.4 Delay

We implemented simple delay using convolution with the impulse response. In the time domain of audio, the impulse response of delay can be obtained simply as Figure 2. There are 2 parameters for delay: feedback value and delay time. Impulse is generated by repeating at the delay time interval. A new impulse is generated at the value of the previous impulse multiplied by the feedback value. The initial impulse value is 1, and the impulse generation is repeated until the impulse value is less than 0.1. Therefore, the feedback value must be less than 1. Let k and t_d denote the feedback value and the delay time, respectively, and m denote the number of iterations. Then, we can represent the transfer function of delay impulse response as below:

$$h[t] = \sum_{i=0}^{m} k^i \cdot \delta[i \cdot t_d] \tag{9}$$

Using the Fourier transform, we can obtain the $H[u,v]$.

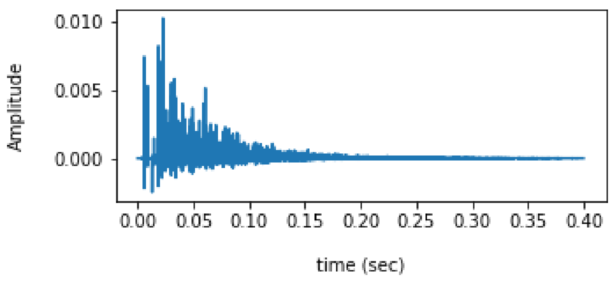


Figure 3: 1-Dimensional room impulse response of reverb

2.5 Reverb

In the case of reverb, we referred studies of room impulse response generation [1, 2]. We only set one parameter which controls the room size. Example of generated RIR is depicted at Figure 3.

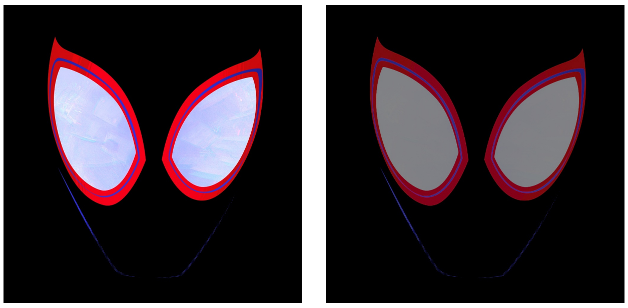
3. METHODOLOGY

3.1 Implementing Audio Effects in OpenCV

There are 5 main components for implementing effects; 1) the compressor, 2) the rgbImgFFT, 3) the other effects except the compressor, 4) the impulseConversion, and 5) the reconstruction. The compressor is implemented at the spatial domain of image, not the frequency domain. In the rgbImgFFT, we separated the input image into three RGB channels and applied the Fourier transforms, respectively, using `DFT()` function of OpenCV. And we made the transfer function of filter in a 1-dimensional frequency domain using `complex`, C++'s standard library. We also made the 1-dimensional impulse responses of both delay and reverb in a time domain and applied the Fourier transforms, respectively. In the impulseConversion, we converted the $H[w]$ into $H[u,v]$. Finally, in the reconstruction, (4) are implemented using the `melSpectrums()` and `iDFT()` function of openCV.

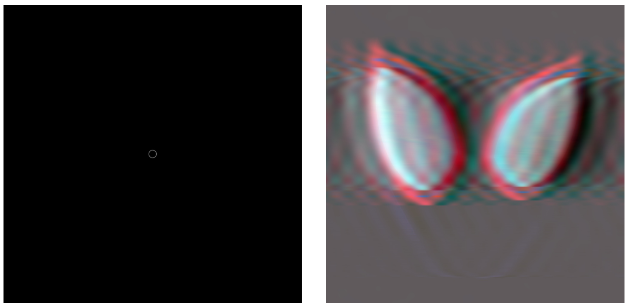
3.2 Implementing JUCE

3 main components compose the overall GUI; 1) the imageList(directory) component, 2) the imagePreview component, and 3) slider box component. We can visualize the list of files inside a certain directory, and display the certain file that has been clicked, using the `WildcardFileFilter`, `TimeSliceThread`, `DirectoryContentsList`, and `FileTreeComponent` class. In order to obtain parameter value in real time and apply it to images, the Timer class' timerCallback method is overridden, and periodically receives the information whether the parameter has been manipulated by the user. Additionally, we implemented functions which maps juce::Image data into cd::Mat data and vice versa.



(a) Original input image (b) Compressed image

Figure 4: (a) Original input image. (b) Compressed image of (a). ($I_t = 120$ and $ratio = 10$)



(a) (b)

Figure 5: (a) Example of low frequency bell-shaped filter. (b) Result of filtering with Figure 4a and (a). (f_c = 400 Hz, $gain$ = 0.5 dB, and Q = 20)

4. RESULT

4.1 Results on Implementing Audio Effects

4.1.1 Compressor

Through Figure 4, we could see that the output compressed image was a little darker.

4.1.2 Filter

- Low-Frequency Bell-Shaped Filter

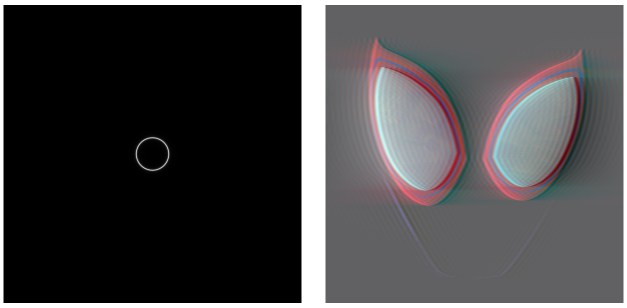
  We set the Q value very large. It means that the width of filter get more narrow. Then, in the frequency domain of image, the peak point of filter becomes an ideal impulse. It causes the ringing artifacts. So, in Figure 5b, we could see that the ringing artifacts are appeared at the low-frequency region.

- Mid-Frequency Bell-Shaped Filter

  We also set the Q value very large like an example of the low-frequency bell-shaped filter. So, in Figure 6b, the ringing artifacts are appeared at the mid-frequency region.

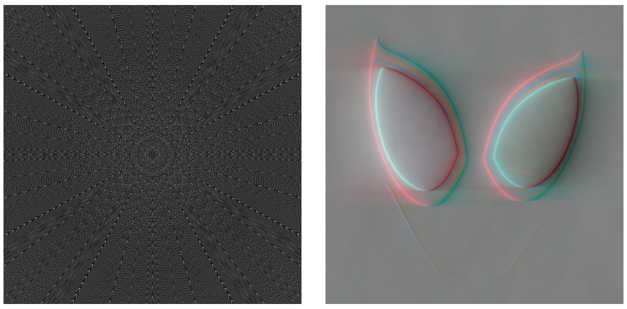
4.1.3 Delay

The 2-d delay impulse response at the frequency domain can be obtained as in Figure 7a. The origin input image is then reconstructed using the delay impulse response. Example of delay result is depicted in Figure 7b.



(a) (b)

Figure 6: (a) Example of mid frequency bell-shaped filter. (b) Result of filtering with Figure 4a and (a). (f_c = 1100 Hz, $gain$ = 0.5 dB, and Q = 20)



(a) (b)

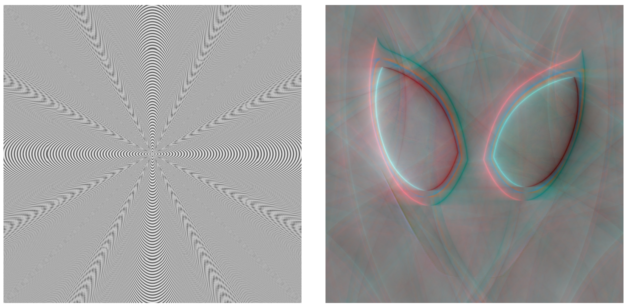
Figure 7: (a) Example of delay impulse response. (b) Reconstruction of Figure 4a with (a). (k = 0.9 and t_d = 0.2 sec)

4.1.4 Reverb

The value of size factor is set to 3, which is the maximum value. We thought that the result of this value was most meaningful. In Figure 8b, the eye edge of "spider-man" are appeared repeatedly.

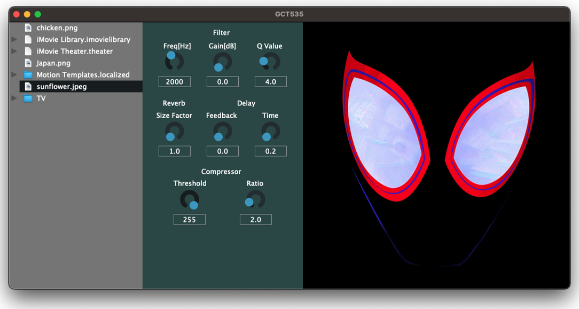
4.2 Overall Results Including JUCE GUI

Overall GUI is shown in Figure 9 and Figure 10. At the leftmost section we can choose any image saved at the specified local directory. The middle section with dark green background is where you can adjust various effect-



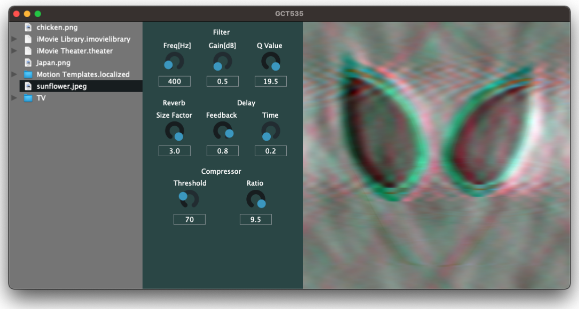
(a) (b)

Figure 8: (a) Example of room impulse response of reverb. (b) Reconstruction of Figure 4a with (a). ($size\,factor$ = 3.0)



Filter
Freq[Hz] 2000
Gain[dB] 0.0
Q Value 4.0
Reverb
Size Factor 1.0
Delay
Feedback 0.0
Time 0.2
Compressor
Threshold 255
Ratio 2.0

Figure 9: Overall GUI. Before applying effects



Filter
Freq[Hz] 400
Gain[dB] 0.5
Q Value 19.5
Reverb
Size Factor 3.0
Delay
Feedback 0.8
Time 0.2
Compressor
Threshold 70
Ratio 9.5

Figure 10: After applying effects

related parameters, such as threshold and ratio for the compressor. The chosen image is displayed on the right section, and is manipulated in a near real-time fashion.

5. DISCUSSION

We discuss the results in 4. The compressor lowers the whole intensity of the image, making the image more dim. In the case of filters, ringing effects are observed starting from the red edge part of each figure and spreading horizontally. We are presuming this outcome due to the sinc shaped waveform of the inverse fourier transformed bell-shaped filter. For the band pass filter (f_c = 1100Hz), the ringing effects appear in a more sharp and dense fashion due to the fact that abrupt change in the intensity of pixels (which indicates the edges inside the image) of each channel leads to a high amount of high frequency. Through the band pass filter more of these edges will be preserved relative to the low pass filter. In the case of delay we can observe some blurs which can be anticipated, and for the reverb we could observe some complex puddle flow-like patterns. Another thing to point out is that for the filter, delay, and reverb, the overall abrupt change in the color components, especially for the background(black to grey). This phenomenon is more drastic in other complex images consisting of various colors. This is due to the fact that we apply each filter separately to the RGB channels, therefore the frequency information of separate RGB channels are retrieved and manipulated differently.

6. REFERENCES

[1] J. B. Allen and D. A. Berkley. Image method for efficiently simulating small-room acoustics. *Journal of the Acoustical Society of America*, 65:943–950, 1976.

[2] P. M. Peterson. Simulating the response of multiple microphones to a single acoustic source in a reverberant room. *The Journal of the Acoustical Society of America*, 80 5:1527–1529, 1986.