AMATH 482 Assignment 2: Rock & Roll and Gabor Transform

Ming Lou

Abstract:

The project mainly talks about the theories and algorithms used to filter music signals according to its frequency. In the project, we were given two music clips with time domain and amplitude. By Using Gabor transform and Spectrogram, I visualized the frequency with varying sizes of windows to adjust between time and frequency.

Section I. Introduction and Overview :

To isolate the bass in Comfortably Numb and see how much the guitar solo can put together in Comfortably numb, I firstly use MATLAB to produce the amplitude graphs with time domain and frequency domain to have a basic concept. With different music instrument has different keys, I can locate the key of bass and guitar to isolate them. However, the difficult thing is we get a good accuracy for frequency while we lost the accuracy for time domain. Thus, I have to try different window sized to obtain a propriate proportion.

The music clips are audio files, so I firstly transferred them to readable graphs with y-axis as the amplitude and x-axis with time and frequency. With both time and frequency information, we are able to find which portion are constructed by guitar and bass.

Section II. Theoretical Background:

From the last assignment we already see that Fourier transform ((1) and (2)) is an integral transformed either from time to frequency or form frequency to time. It's useful because we can observe both domain once we have one of them. However, we lost the time accuracy when we use the frequency domain.

$$\hat{f}(k) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x) e^{-ikx}\, dx \qquad (1)$$

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{f}(k) e^{ikx} dk \qquad (2)$$

Gabor transform is a good solution to this dilemma. The main idea of Gabor transform is to break the Fourier transform to several small windows and add filter g(τ)to each piece. We can also observe this from the definition of Gabor transform (3).

$$\tilde{f}_g(\tau, k) = \int_{-\infty}^{\infty} f(t) g(t-\tau) e^{-ikt}\, dt \qquad \textbf{(3)}$$

One common filter for Gabor transform, also the one I use in the code, is Gaussian curve with a mean of $\boldsymbol{\tau}$, The standard deviation varies depending on the decided-upon width of the time-window. This function is $e^{-a(t-\boldsymbol{\tau})^2}$.

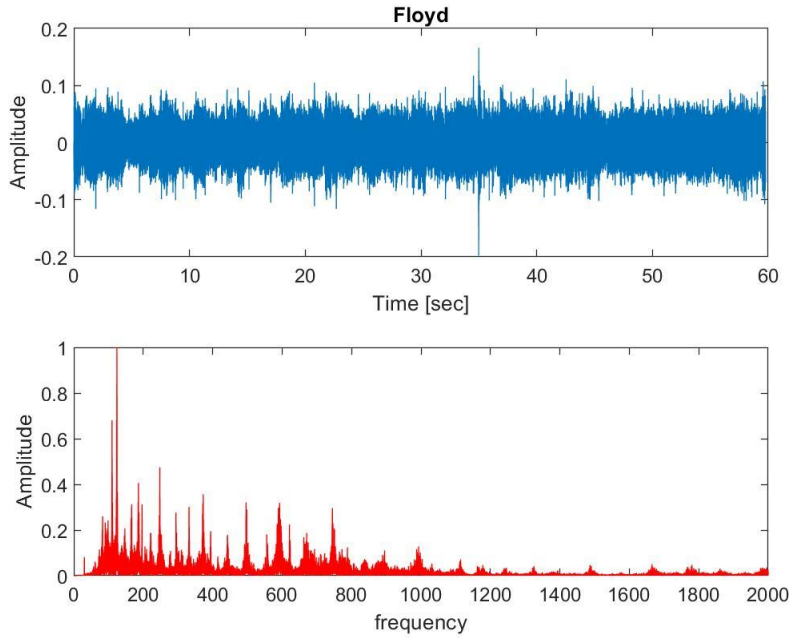
Section III. Algorithm Implementation and Development

Main steps:

1. Loading the music clips to MATLAB and visualizing the raw data for both the time domain and the frequency domain
2. Using different sizes of windows to adjust the accuracy between time and frequency. Applying the Gabor transform and Spectrogram to creating the graph with both time and frequency
3. According to the key of bass and guitar, isolate them form the overtone

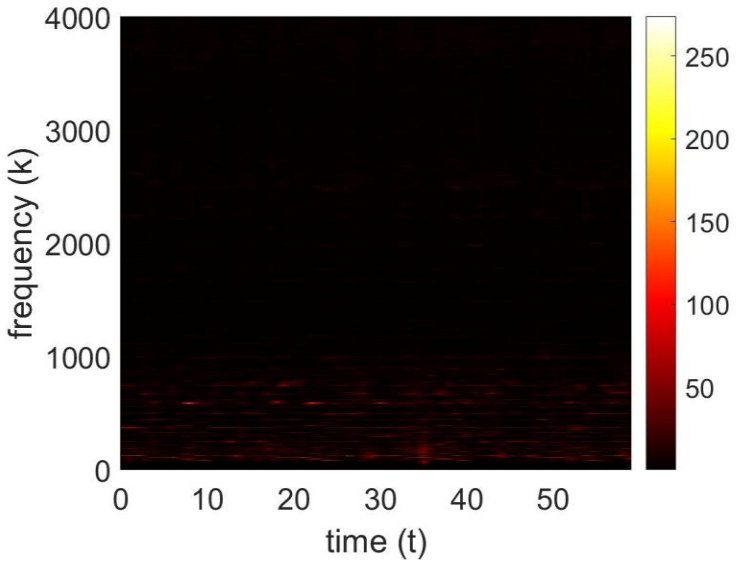
Details:

Firstly, I loaded the Floyd.m4a and created the plots for time domain and frequency domain. The figure is shown below (figure1).



(figure 1)

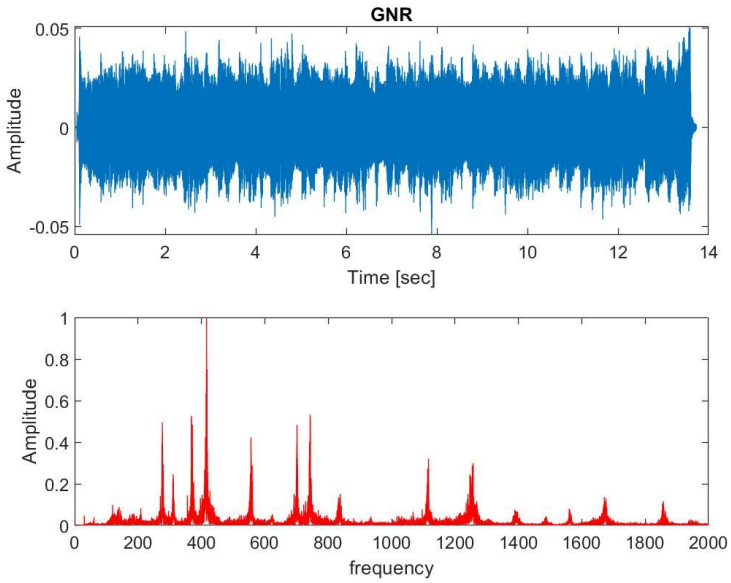
Form this figure, we have a basic idea about how this music clip looks like. The next step is to use a proper window size to combine the time and frequency together. With the multiplication of filter and the good window size. I created a spectrogram (figure2) .



(figure2)

From the figure, we can observe the relationship between time and frequency. Searching online, I found the frequency for bass is around 160. So we can isolate the bass with frequency near to 160 and get the corresponding time.

The next step is very similar, only transfer from Floyf.m4a to GNR.m4a. Similarly, I loaded the file and created figure3.



(figure3)

Section IV. Computational Results

For this project, I tried lots of code but the raw data is large, my computer are unable to run it. The computational result is not shown in this paper. I can only tell that we can draw a pretty good music notes if we use the spectrogram to visualize the different keys for the music clip. We can then isolate them.

Section V. Summary and Conclusions

Gabor transform is a very useful algorithm to identify the frequencies at different time for signals. After isolate the instrument we want, we can use the inverse Gabor transform to a new music notes.

Appendix A. MATLAB functions used and brief implementation explanation

1. fftshift: This function shifts the zero-frequency component of discrete Fourier transform to center of spectrum and it helps the visualization of FFT in the frequency domain.
2. fft : Performs a fast fourier transform on X, a 2-D array. We used this fourier transform the filter multiplied by the data at each of our time points.

Appendix B. MATLAB codes

```
%%
figure(1)
[y, Fs] = audioread('Floyd.m4a');
n=length(y);
L=n/Fs; %record time in seconds
t=(1:n)/Fs;
k=(Fs/n)*[0:(n-1)/2 -(n-1)/2:-1];
ks=fftshift(k);
S = y';
St = fft(S);

subplot(2, 1, 1);
plot(t, S);
xlabel('Time [sec]'); ylabel('Amplitude');
```

```
title('Floyd');

subplot(2, 1, 2);
plot(ks, abs(fftshift(St))/max(abs(St)), 'r');
set(gca, 'XLim', [0 2e3]);
xlabel('frequency'); ylabel('Amplitude');


%%
a = 20;
tau = 0:1:L;
for j = 1:length(tau)
    g = exp(-a*(t - tau(j)).^2);
    Sg = g.*S;
    Sgt = fft(Sg);
    Sgt_spec(:,j) = fftshift(abs(Sgt));
end


pcolor(tau,ks,Sgt_spec)
shading interp
set(gca,'ylim',[0 4000],'Fontsize',16)
colormap(hot)
colorbar
xlabel('time (t)'), ylabel('frequency (k)')

%% %% GNR
clear variables;
figure(2)
[y, Fs] = audioread('GNR.m4a');
n=length(y);
L=n/Fs; %record time in seconds
t=(1:n)/Fs;
k=(Fs/n)*[0:(n-1)/2 -(n-1)/2:-1];
ks=fftshift(k);

S = y';
St = fft(S(1:659121));


subplot(2, 1, 1);
plot(t, S);
xlabel('Time [sec]'); ylabel('Amplitude');
title('GNR');

subplot(2, 1, 2);
plot(ks, abs(fftshift(St))/max(abs(St)), 'r');
```

```
set(gca, 'XLim', [0 2e3]);
xlabel('frequency'); ylabel('Amplitude');



%%
a = 20;
tau = 0:1:L;
for j = 1:length(tau)
    g = exp(-a*(t - tau(j)).^2);
    Sg = g.*S;
    Sgt = fft(Sg);
    Sgt_spec(:,j) = fftshift(abs(Sgt));
end


pcolor(tau,ks,Sgt_spec)
shading interp
set(gca,'ylim',[0 4000],'Fontsize',16)
colormap(hot)
colorbar
xlabel('time (t)'), ylabel('frequency (k)')
```