AMATH 482 Assignment 2

Carter Peyton

February 10, 2021

Abstract

This assignment aims to analyze music clips in order to identify instruments. Through the use of a Gabor filter, two music clips are filtered in order to find certain instruments inside them. Then after the instruments have been identified, a filter in frequency space will be applied to the clip in order to isolate those instruments. The results will illustrate the clips with isolated instruments.

1 Introduction and Overview

Say you want to analyze portions of the songs 'Sweet Child O' Mine' by Guns N' Roses and 'Comfortably Numb' by Pink Floyd. Both clips contain multiple instruments with unknown frequencies to you. Through the use of a Gabor filter, the guitar in the Guns N' Roses clip and the bass in the Pink Floyd clip will be identifiable. Then using the Fourier transform and a filter in frequency space, the bass in 'Comfortably Numb' can be isolated. Lastly, through trial and error, the guitar solo in 'Comfortably Numb' can be reconstructed with a filter in frequency space.

2 Theoretical Background

2.1 Gabor Filtering

2.1.1 The Gabor Transform

The Gabor transform is a transform applied to a function $f(t)$ so that

$$f_g(t,k) = \int_{-\infty}^{\infty} f(t) * g(t-\tau) * e^{-ikx} dx, \tag{1}$$

Where τ is the center of the Gabor transform's window. It's important to note that $g(t-\tau)$ can be any function. For our purposes we choose a Guassian where

$$g(t-\tau) = e^{-a*(t-\tau)^2} \tag{2}$$

The Gabor transform assumes that the function g is real and symmetric and that the L_2 norm of g is set to unity ($||g||_2 = 1$), meaning that no added energy is put into the system. The absence of added energy, allows the transform to serve as an accurate filter. The window size of the filter (corresponding to a where $a > 0$) controls the accuracy of time versus frequency. A large window may capture all the frequencies within that window, but it will not be well localized in time and vice-versa.

2.1.2 Sliding the Gabor Transform

What makes the Gabor transform powerful, is when you slide the transform across many values of τ (i.e across the range of your original function), you can get information about the frequencies throughout the entire signal.

2.2 Low/High Pass Filtering

A low pass filter is one that only lets frequencies below a certain value "through the filter." A high pass filter only lets values above a certain frequency through. We construct a low pass filter by using a piece wise function $f(x)$ where

$$f(x) = \begin{cases} f(x) & x < k \\ 0 & else \end{cases} \tag{3}$$

Note that a high pass filter would have $f(x) = f(x)$ for any $x \geq k$.

2.3 Spectrograms

A spectrogram is an image that is used to convey information about frequency over time. The image itself contains all the values of τ that the Gabor filter sweeps over. The y axis contains a range of frequency values, and the color of each point corresponds to the strength of that frequency at that point in time.

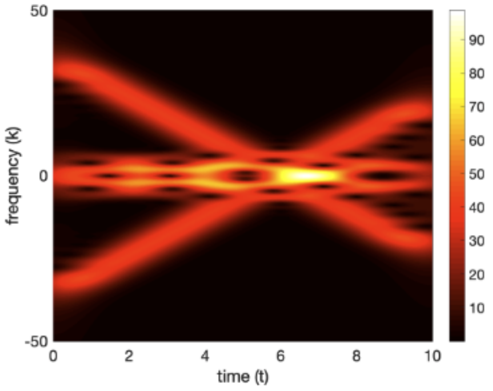

Figure 1: An example of a spectrogram. Credit: Jason Bramburger

3 Algorithm Implementation and Development

3.1 Initialization

The following algorithm reads each music clip into MATLAB and sets up Fourier domains for each clip, scaling by 2π and the appropriate L according to the clip. The GNR clip is around 14 seconds long, so the Gabor filter can be applied to the entire clip. However, the Floyd clip is around 60 seconds long. Due to its length, memory issues are incurred when computing the Fourier transform, so only a fifth of the clip is analyzed. This reduces memory usage. A time vector is set up that runs the length of each clip, and an additional tau vector (with time step $dt = 0.05$ for GNR and $dt = 0.1$ for Floyd) is set up to control the center of the Gabor filter.

3.2 Gabor Filtering

The same Gabor filtering process is applied to both clips (with the main difference being the lengths of each clip). First a window size is defined ($a = 100$ in both cases). Then, for all values of the tau vector, a Gabor filter is created with the given width and center at the current value of tau. The filter is then multiplied against the transpose of the original sample data, and the FFT is computed on that product. In order to produce a spectrogram, we need to use fftshift to realign the parts of the product and take the absolute value of the result. This results in a matrix containing values for each frequency at each time.

3.3 Filtering Floyd in Frequency Space

In order to isolate the bass in Floyd, we can use a low-pass filter in order to filter out all frequencies above 200 Hz. The frequencies below 200 Hz correspond to the base line in the Floyd clip. Importantly, this low pass filter must be applied at all times during the Gabor filtering process. After the low pass filter has been applied, a spectrogram is computed to show only the bass line.

3.4 Reconstructing the Floyd Guitar

Similar to isolating the bass in Floyd, we can use a filter in frequency space to isolate the Guitar solo in Floyd as well. Due to the higher frequency of the guitar than the bass, we can use a high pass filter to try and isolate the guitar. The process is nearly identical to filtering the bass, except we filter to frequencies above 300 Hz instead of below 200 Hz. However, there are upper end frequencies as well, likely due to other instruments or vocals. Therefore, we can apply a low pass filter at some frequency higher than 300 (say 600 Hz) in order to isolate a channel of frequencies that ranges from 300 to 600 Hz.

4 Computational Results

4.1 Gabor Filtering on Music Clips

4.1.1 GNR Results

The following spectrogram is computed for the GNR clip. Note the distinct lines of "dots" these dots correspond to notes being played. A longer dot corresponds to a note being held for longer. There are 'bands' of these dots around 275, 375, and 425 Hz respectively. These correspond to the notes C#, F#, and an A flat all in the fourth octave.

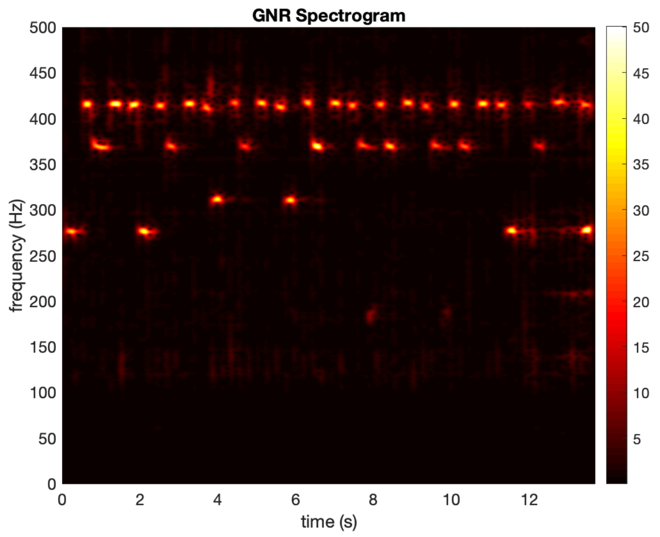

Figure 2: The spectrogram for the GNR clip with $a = 100$ and timestep $dt = 0.05$

4.1.2 Floyd Results

The following spectrogram is computed for the first fifth (12 seconds) of the Floyd clip. We can clearly see, starting at 125 Hz, the base line that is played in the song. There are 4 distinct notes where the 4th note seems to start the pattern again. The notes fall on the frequencies: 125 Hz, 110 Hz, 100 Hz, and back to 125 Hz. The corresponding times for these notes are (0 - 3, 3 - 6, 6 - 8, 10 - 12) in seconds. The frequencies correspond to the notes B, A, A flat, and back to B in the fourth octave.

Note: There is a line starting around 190 Hz that follows a similar pattern to the main bass line. This is likely an overtone.

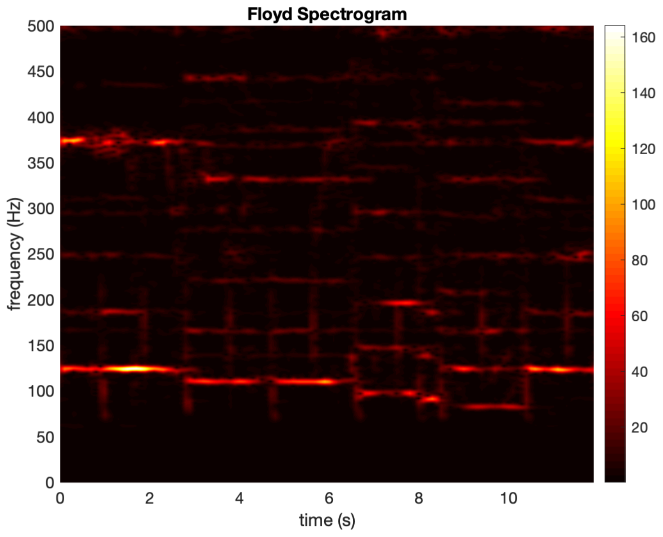

Figure 3: The spectrogram for the Floyd clip with $a = 100$ and timestep $dt = 0.1$

4.2 Bass Isolation in Floyd

After the bass has been isolated, the main bass and overtone notes are clearly distinguishable.

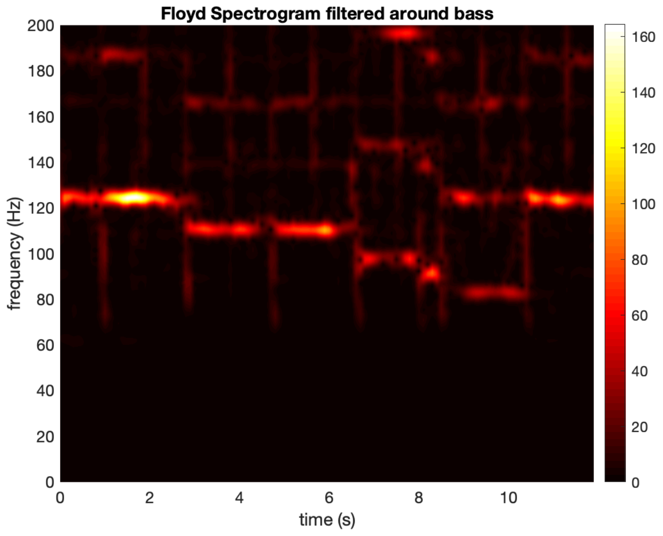

Figure 4: The spectrogram of Floyd after applying a low pass filter for frequencies less than 200 Hz.

4.3 Floyd Guitar Reconstruction

After applying a high pass filter for frequencies greater than 300 Hz in conjunction with a low pass filter for frequencies less than 600 Hz, the following spectrogram is computed.

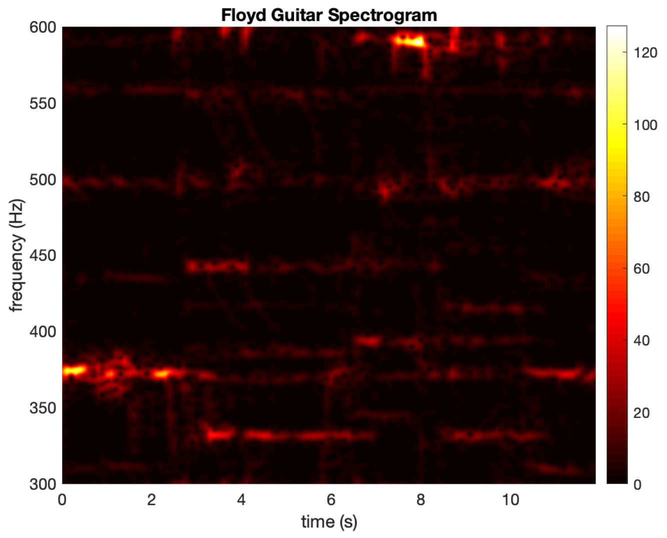

Figure 5: The spectrogram of Floyd after isolating the guitar.

The guitar solo is much more complex than the bass line. However, in the first 12 seconds of the clip we can distinguish three specific notes around 375 Hz, 340 Hz, and 580 Hz respectively. These frequencies correspond to F# and E in the fourth octave, with the last note being a D in the fifth octave.

5 Summary and Conclusion

Through the use of Gabor filtering we were able to identify the notes played by the guitar in GNR, the Bass in Floyd, and the Guitar in Floyd. The guitar in the GNR clip played the notes C#, F#, and an A flat all in the fourth octave. The bass in Floyd played a pattern consisting of B, A, A flat, and back to B in the second octave. Lastly, we were able to determine three major notes of the first 12 seconds of the guitar solo, F# and E in the fourth octave, with the last note being a D in the fifth octave.

Appendix A MATLAB Functions

- `[y, Fs] = audioread('file.m4a')` returns values of an audio file in y, along with a sampling rate Fs
- `audioplayer([y,Fs)` plays an audio clip corresponding to y and Fs
- `pcolor(x,y,z)` plots a 2D version of a 3D plot, where the color corresponds to the height (or z value) of the 3D plot
- `y = linspace(x1,x2,n)` returns a row vector of `n` evenly spaced points between `x1` and `x2`.
- `kx = fftshift(k)` rearranges a Fourier transform X by shifting the zero-frequency component to the center of the array.
- `avg = zeros(n,n,n)` creates an n x n x n matrix of zeros.
- `B = reshape(A,sz)` reshape(A,sz) reshapes A using the size vector, sz, to define size(B).
- `Y = fftn(X)` returns the multidimensional Fourier transform of an N-D array using a fast Fourier transform algorithm.
- `[val,ind] = max(oneD)` returns the maximum value and corresponding index of the vector oneD

- `I = ifftn(F)` computes the inverse multidimensional fast Fourier transform.

Appendix B MATLAB Code

```
%% Part 1a - GNR (only Guitar), once have spectrogram try log and the bass/guitar line will be very cle
clear all; clc
[y, Fs] = audioread('GNR.m4a'); % note that Fs is the sample rate (Hz)
trgnr = length(y)/Fs; % record time in seconds

L = trgnr;
n = length(y);
k = (2*pi/(L))*[0:(n/2 - 1) -n/2:-1]; ks = fftshift(k);
t = (1:length(y))/Fs;
tau = 0:0.05:L;


figure(1)
a = 100;
tic
for i = 1:length(tau)
    g = exp(-a*(t-tau(i)).^2);
    Sg = g.*y';
    Sgt = fft(Sg);
    Sgtspec(:,i) = fftshift(abs(Sgt));
end
toc
pcolor(tau,ks/(2*pi),Sgtspec)
shading interp
set(gca,'Ylim',[0 500],'Fontsize',12)
colormap(hot)
colorbar
title('GNR Spectrogram')
xlabel('time (s)'), ylabel('frequency (Hz)')
print('gnr_spec_new.png','-dpng')

%% Part 1b - Floyd bass
clear all; clc
figure(2)
[y, Fs] = audioread('Floyd.m4a'); % note that Fs is the sample rate (Hz)
y = y(1:((length(y)-1)/5)); % look at the first tenth of the clip (due to memory issues)
trgnr = (length(y))/Fs; % record time in seconds
L = trgnr;
n = length(y);
k = (2*pi/(L))*[0:(n/2 - 1) -n/2:-1]; ks = fftshift(k);
t = (1:n)/Fs;
tau = 0:0.1:L;
size(t)
size(y')

a = 100;
tic
for i = 1:length(tau)
    g = exp(-a*(t-tau(i)).^2);
    Sg = g.*y';
    Sgt = fft(Sg);
    Sgtspec(:,i) = fftshift(abs(Sgt));
end
toc
pcolor(tau,ks/(2*pi),Sgtspec)
shading interp
set(gca,'Ylim',[0 500],'Fontsize',12)
colormap(hot)
colorbar
xlabel('time (s)'), ylabel('frequency (Hz)')
title('Floyd Spectrogram')
print('floyd_spec_new.png','-dpng')


%% Part 2 - Applying low pass filter to Floyd (isolate bass)
clear all; clc
figure(3)
[y, Fs] = audioread('Floyd.m4a'); % note that Fs is the sample rate (Hz)
y = y(1:((length(y)-1)/5)); % look at the first tenth of the clip (due to memory issues)
trgnr = (length(y))/Fs; % record time in seconds
L = trgnr;
n = length(y);
k = (2*pi/(L))*[0:(n/2 - 1) -n/2:-1]; ks = fftshift(k);
t = (1:n)/Fs;
tau = 0:0.1:L;
size(t)
size(y')

a = 100;
tic
for i = 1:length(tau)
    g = exp(-a*(t-tau(i)).^2);
    Sg = g.*y';
    Sgt = fft(Sg);
    for j = 1:length(Sgt)
        freq = k(j)/(2*pi);
        if abs(freq) > 200
            Sgt(j) = 0;
        end
    end
    Sgtspec(:,i) = fftshift(abs(Sgt));
end
toc
pcolor(tau,ks/(2*pi),Sgtspec)
shading interp
set(gca,'Ylim',[0 200],'Fontsize',12)
colormap(hot)
colorbar
xlabel('time (s)'), ylabel('frequency (Hz)')
title('Floyd Spectrogram filtered around bass')
print('-dpng','floyd_bass_filt.png')
```

```
%% Part 3 - Recreate the guitar solo in Comfortably Numb (high pass filter)
clear all; clc
figure(4)
[y, Fs] = audioread('Floyd.m4a'); % note that Fs is the sample rate (Hz)
y = y(1:((length(y)-1)/5)); % look at the first tenth of the clip (due to memory issues)
trgnr = (length(y))/Fs; % record time in seconds
L = trgnr;
n = length(y);
k = (2*pi/(L))*[0:(n/2 - 1) -n/2:-1]; ks = fftshift(k);
t = (1:n)/Fs;
tau = 0:0.1:L;
size(t)
size(y')

a = 100;
tic
for i = 1:length(tau)
    g = exp(-a*(t-tau(i)).^2);
    Sg = g.*y';
    Sgt = fft(Sg);
    for j = 1:length(Sgt)
        freq = k(j)/(2*pi);
        if abs(freq) < 300 || abs(freq) > 600
            Sgt(j) = 0;
        end
    end
    Sgtspec(:,i) = fftshift(abs(Sgt));
end
toc
pcolor(tau,ks/(2*pi),Sgtspec)
shading interp
set(gca,'Ylim',[300 600],'Fontsize',12)
colormap(hot)
colorbar
xlabel('time (s)'), ylabel('frequency (Hz)')
title('Floyd Guitar Spectrogram')
print('-dpng','floyd_guitar_filt.png')
```

Code from homework2.m