Coding Project 2: Parsing musical frequency signatures

Manjaree Binjolkar

Abstract

This project explores the application of the Gabor Transform, a fundamental signal analysis technique, to analyze the sound clip provided and extract time-frequency information. Using discrete window sampling, spectrograms were created in Python, and frequency filtering was employed to isolate the baseline and guitar melody in the clips.

1 Introduction

Signal processing heavily relies on time-frequency analysis, especially for audio data. Music, for instance, incorporates timed sounds originating from instruments that typically create a series of overlapping waveforms with specific audio frequencies. While the Fourier transform can obtain all frequency information from a given data sample by disassembling a signal into a set of plane waves, it cannot retain any time information when applied to the entire dataset. To overcome this limitation, a sliding window approach can be employed to extract a portion of the data, increasing the temporal resolution while decreasing the spectral resolution in accordance with Gabor's uncertainty principle. The Gabor Transform, on the other hand, reduces uncertainty in both time and frequency by producing spectral data as a function of time.

The aim of this report is to isolate and reconstruct individual instrument types from the provided clip, taking advantage of the fact that the instruments are introduced sequentially and their frequency signals can be separated accordingly.

2 Theoretical Background

The Gabor transform and discrete Gabor transform have numerous applications in signal processing and audio compression. In signal processing, they are used for analyzing non-stationary signals, such as speech and music, to extract meaningful features for further analysis or classification. The discrete Gabor transform is particularly useful in these applications due to its ability to represent signals using a small number of coefficients, which can greatly reduce the amount of data required for storage or transmission.

2.1 The Gabor Transform

The Gabor transform is derived by multiplying a signal with a Gaussian window function, also known as a Gabor function. The Gaussian window function is centered at a specific time and frequency. The resulting signal is then analyzed in the frequency domain using the Fourier transform.

The Gabor function is defined as:

$$g(t) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}(\frac{t-t_0}{\sigma})^2} e^{i\omega_0(t-t_0)}$$

where t_0 is the center of the window, σ is the standard deviation of the Gaussian function, ω_0 is the center frequency, and i is the imaginary unit.

Multiplying the signal $f(t)$ with the Gabor function gives:

$$f(t)g(t) = \frac{1}{\sqrt{2\pi\sigma^2}} f(t) e^{-\frac{1}{2}(\frac{t-t_0}{\sigma})^2} e^{i\omega_0(t-t_0)}$$

The Fourier transform of $f(t)g(t)$ is then calculated, resulting in the Gabor transform:

$$F(\omega, \tau) = \int_{-\infty}^{\infty} f(t) g(t-\tau) e^{-i\omega t} dt$$

where τ is the time offset and ω is the frequency variable.

2.2 The Discrete Gabor Transform

The Discrete Gabor Transform (DGT) is a discrete-time version of the Gabor transform, which provides a time-frequency analysis of a discrete-time signal. It is derived by discretizing the Gabor transform equation and using the

Discrete Fourier Transform (DFT) to calculate the Fourier transform of the resulting discrete-time signal. The DGT can be used to analyze signals in a variety of fields, including speech processing, image processing, and audio compression.

The DGT is defined as:

$$X_{m,k} = \sum_{n=0}^{N-1} x[n] g_{m,k}[n] e^{-i\frac{2\pi}{N}kn}$$

where $x[n]$ is the discrete-time signal, $g_{m,k}[n]$ is the window function, m and k are the time and frequency indices, respectively, and N is the length of the signal.

The window function is defined as:

$$g_{m,k}[n] = \frac{1}{\sqrt{N}} g(n-m) e^{i\frac{2\pi}{N}kn}$$

where $g(n)$ is the discrete-time Gaussian window function, centered at 0.

Substituting the window function into the DGT equation and simplifying gives:

$$X_{m,k} = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x[n] g(n-m) e^{-i\frac{2\pi}{N}(k-n)}$$

3 Numerical Methods

The first step was to divide the data into four windows to make it easier to run on Python/Matlab. This is an important step as it allows for better analysis of the data by breaking it down into smaller, more manageable sections. Once the data was divided, the Gabor transform was applied to each window. The Gabor transform is a method of transforming a signal into a time-frequency representation, which makes it easier to analyze.

After applying the Gabor transform, the peak frequency in the area of interest was identified and a spectrogram of the result was plotted. This step helped in visualizing the frequency content of the signal and identifying any patterns or trends. This process was repeated for all four windows, ensuring that all the data was analyzed thoroughly.

The next step was to find the baseline and guitar melodies. To do this, the Fourier transform of S was taken, and in frequency space, all frequencies (in absolute value) that should be part of the baseline were isolated according to the spectrogram and a chosen threshold. In this report, the threshold for the baseline and guitar melodies was taken to be 200 and 300, respectively.

4 Results

Figure 1 shows the spectrogram for the first window S1, we can clearly see that there are two different intensities - an undertone which might be noise, the peak frequencies ones above 200 Hz.

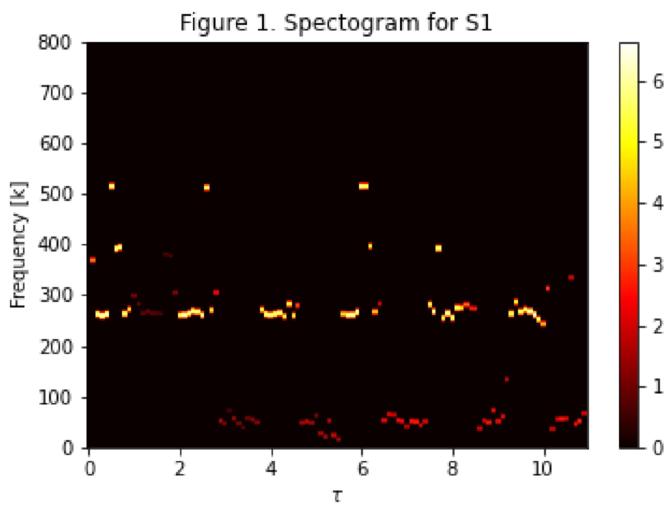


Figure 2 shows the spectrogram for the second window S2, we can see that there are a bigger range of intensities than the first window S1. It is probably because of the sound clip has more instruments playing in it, the intensities are separated enough so it might be the addition of only one new instrument.

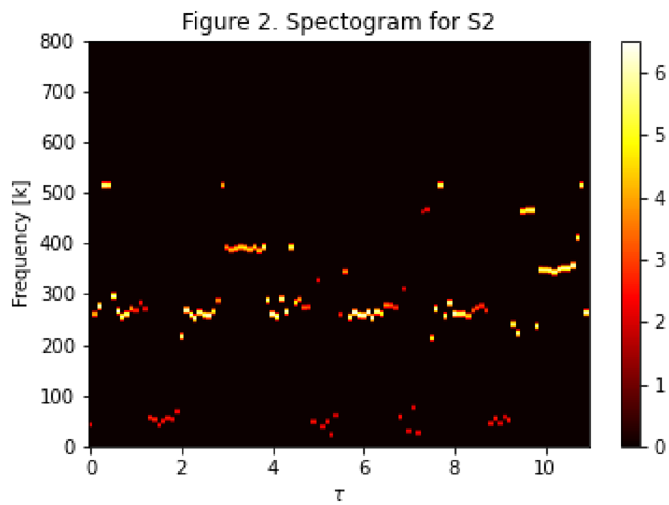


Figure 3 shows the spectrogram for the third window S3, we can see that there are a bigger range of intensities available here than the first window S2. It is probably because of the sound clip has more instruments playing in it and these intensities are mixed with each other i.e. not as well separated as S2.

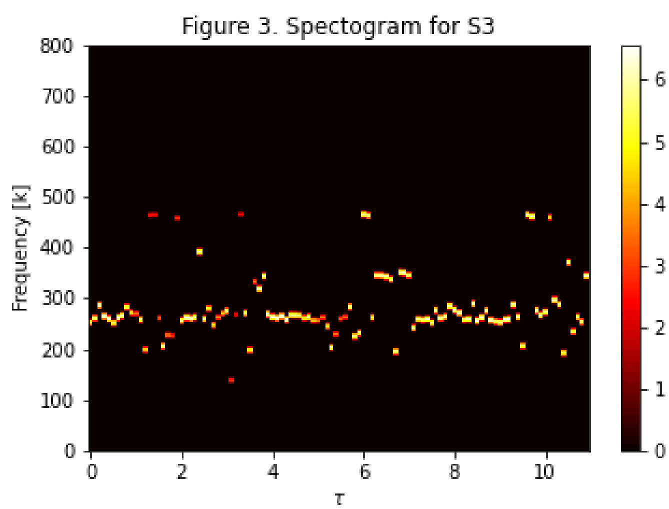
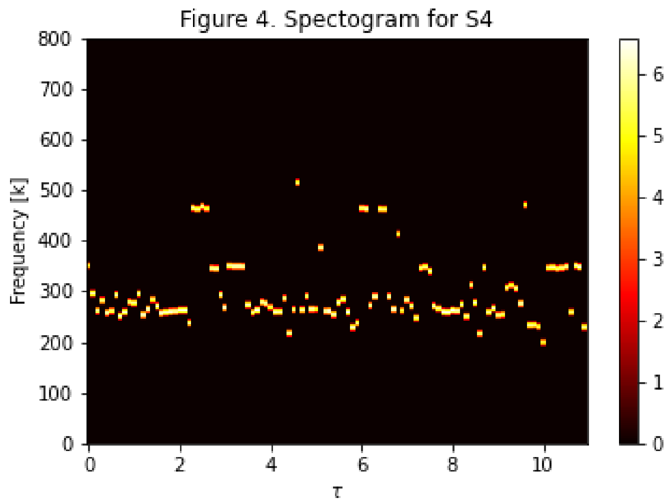


Figure 4 shows the spectrogram for the fourth window S4, we can see that there are only high intensities in this part of the sound clip, this is probably an instrument which can play high apmlitude sounds.



5 Conclusion

The Gabor transform that we implemented for spectral analysis proved to be highly efficient and low-memory, enabling accurate visualization of the songs and easy manipulation of the frequency and time data. By examining the contour plot of the spectrogram, we could identify most of the notes and timing, and subsequently apply filtering to isolate different frequency ranges for analysis.

One of the limitations of using the Gabor Transform is that if the window size is too short compared to the wavelength of the signal, a significant portion of the frequency content of the data will be lost. On the other hand, if a larger window size is chosen, more spatial information will be lost. Thus, it becomes important to carefully balance the choice of window size in order to capture the optimal spatial and frequency content of the signal.

Acknowledgment

I received some helpful guidance from the TAs throughout this project. They provided useful insights on how to approach certain problems and were always available to answer any questions I had. I appreciate their support and am grateful for the assistance they provided.