University of Washington, Seattle

Time-frequency Analysis with Gabor Transform

With Application in Localizing Sound Frequency in Time Period

Abstract:

This paper illustrates the advantages of Gabor transform and its modification to Fourier transform to perform time-frequency analysis. This technique allows us to extract feature from subsets of the original data and understand better properties of each signal's part. This paper demonstrates the feature extraction on audio analysis application by using spectrograms and sound frequency retrieval from musical recordings.

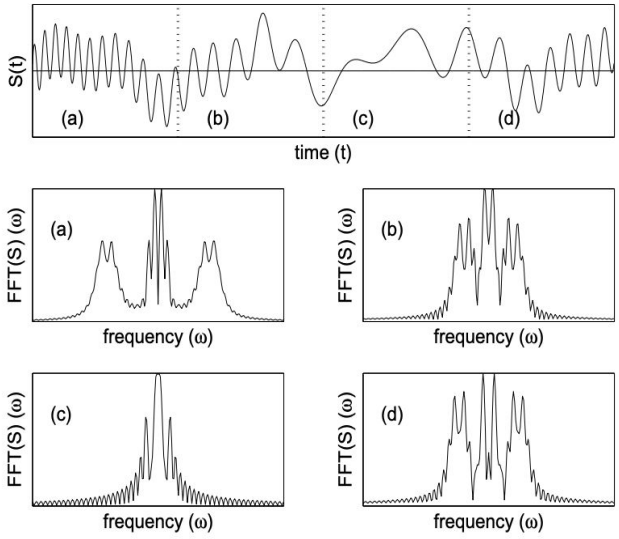
Hung Vinh Ngo
AMATH482
Professor Kutz
Friday, February 8, 2019

Section 1: Introduction and Overview

Fourier transform and its application give us another view of signal processing by letting us consider the frequency aspects. In many practical cases, it is extremely useful to analyze signal based on its frequency since most natural phenomenons are easier to express in wavelength and frequency. However, because the transformation only focuses on converting to frequency domain, it loses the original time localization as a result and fail to capture the moment in time of each frequency. Fourier transform provides us the whole range of frequencies from low to high but does not tell us when those occur. This imposes one disadvantage of Fourier transform since a lot of applications depend on both time and frequency aspects such as radar detection or sound analysis where exact classification and identification are vital. For instance, the application of spectrogram is useful for many audio processing tasks such as classification, discovering pattern and even as input to neural network for speech recognition[1]. Therefore, windowed Fourier transform, or in particular, Gabor transform, modified the original ideas by letting us perform multiresolution time-frequency analysis.

Section 2: Theoretical Background

Gabor transform works by using a filter "window" sliding through the signal so that it will only focus on a small section of the signal for each time step. After obtaining each subset, it will perform Fourier transform and obtain the range of frequencies occurred in that period.



This idea of segmenting the original data to many subsets gives us a better understanding of the signal at a smaller unit scale. By applying Fourier transform, we can approximately obtain the time that each signal occurs, provided that window width parameters is small enough. By varying the translation (how fast the window is moving) and the dilation (how wide is the window), we can obtain different views of the time and frequency in the signal.

Figure 1: Gabor transform on 4 windowed subsets of a signal S(t)[2]

[1] "Investigating modulation spectrogram features for ... - Semantic Scholar." https://www.semanticscholar.org/paper/Investigating-modulation-spectrogram-features-for-Baby-hamme/109bca29f6fae51b71fd48e65b9c5e5f3881e591. Accessed 10 Feb. 2019.

[2] "Basic Computation & Visualization - University of Washington." https://faculty.washington.edu/kutz/KutzBook/KutzBook.html. Accessed 10 Feb. 2019.

Mathematically, this idea of applying filters to obtain windowed is relatively simple provided a legal filter with correct mathematical properties is used. For instance, a basic Gaussian kernel with peak height of one unit can be multiplied with the original data to achieve the desired result since the tails of a Gaussian curve will essentially cancel out other data points:

$$g_{t,\omega}(\tau) = e^{i\omega t} g(\tau - t)$$

Equation 1: Gaussian Kernel for Gabor transform

There are a variety of filters that we can choose from for signal analysis depending on the application. In the following section, we will consider two more filters which are Mexican Hat and Shannon filters. So intuitively, by sliding our window across the data and Fourier transform it, we obtain information about the frequency in that particular domain. A spectrogram can be generated by plotting that matrix of data as a colormap.

It is interesting to note that this idea of obtaining insights of each smaller subset of data is important and widely applied in computer vision tasks such as face recognition or feature extraction[3]. For instance, by focusing on smaller section of an image and discover its patterns, we can extract and use those smaller features to compute the similarity with other images in the database.

Section 3: Algorithm Implementation and Development

Section 3.1: Understand Spectrogram Plot

This section will use a section of Handel's Messiah to illustrate the time-frequency analysis with Gabor transform and how to create spectrogram from sound data.

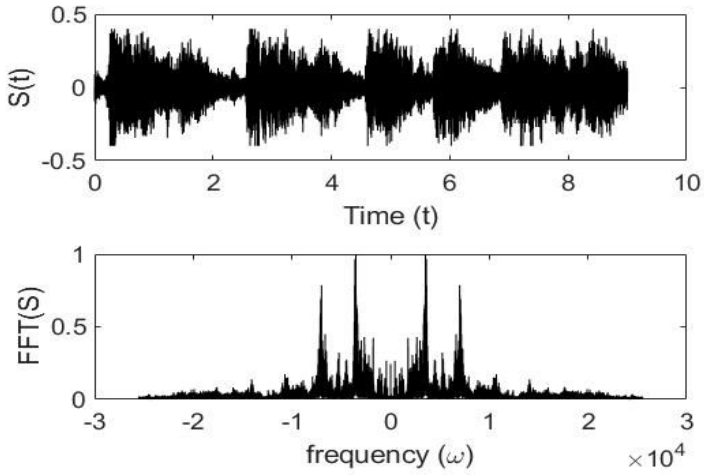


Figure 2 : (top) A section of the Handel's song plotted in time vs amplitude recorded. (bottom) Fourier transform of that section in the whole time domain.

Given the recorded song, it is necessary to know what sound frequency occurs across timesteps. Fourier transform can help us distinguish the number of frequencies happen during the recording. However, a song is significant only when the frequencies are discovered along with the exact time that they happened, for example, to recreate the song. As a result, we need a windowed slide transform, or Gabor transform, to obtain a more detailed multi-resolution analysis of both the time and frequency data of this particular song.

[3] "Gabor Wavelet Transform and Its Application (PDF)." http://disp.ee.ntu.edu.tw/~pujols/Gabor%20wavelet%20transform%20and%20its%20application.pdf. Accessed 11 Feb. 2019.

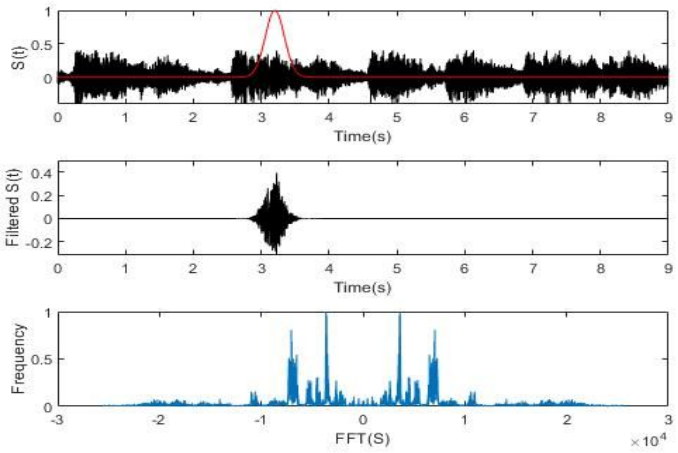


Figure 3: (top) Time signal S(t) and the Gabor time filter (bold line) for a Gaussian filter. (middle) The product of the filter and the original signal. (bottom) Fourier transform the filtered signal gives the frequency of that particular filtered timestep.

When using Gabor transform across timesteps, we get a better understanding of what frequency occurs in each timestep, and therefore can better visualize how the song notes occurs through time. We can collect the Fourier domain frequency signal over all timesteps and plot them out as a spectrogram.

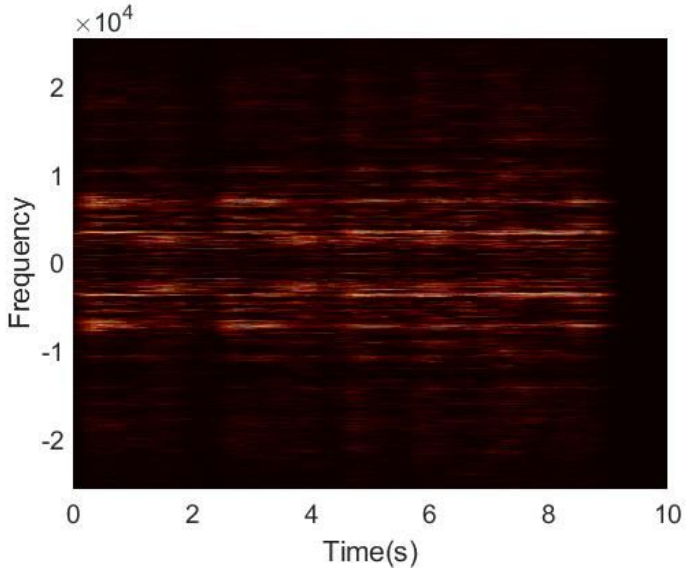


Figure 4: Spectrogram for Gaussian filter with dilation of 25 unitand translation of 0.1 unit.

Furthermore, we can flexibly vary the parameters of the model including the filter width, filter translation or even different filter equation to detect more signal or better time resolution. For instance, a larger filter width will catch a bigger range of frequency, but will lose time accuracy and vice versa. This choice of parameters yield a bias problem to obtain a certain degree of correctness in either frequency or time localization, but not both. Figure 5 and 6 illustrates the decrease in window width and filter translation yield better and “sharpness” in time aspect as we can clearly see which frequency occur at each time step. However, it fails to capture higher frequencies compared with bigger window width and vice versa.

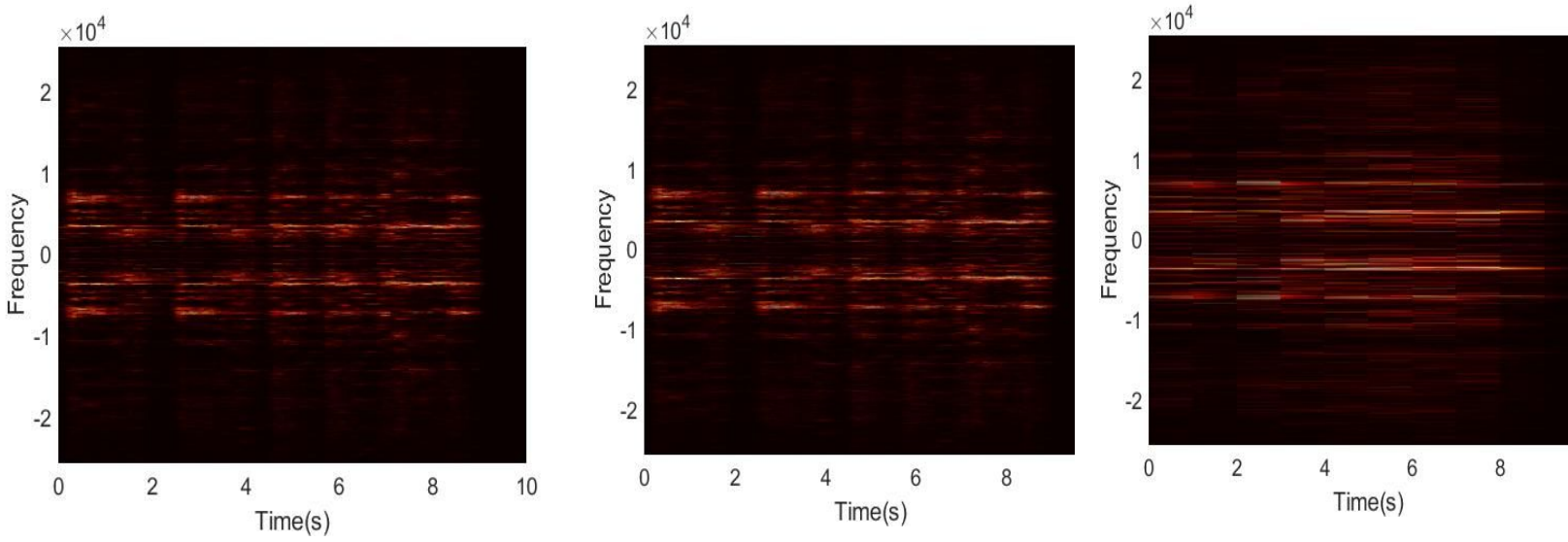


Figure 5: Comparison of different Gaussian window width and same translation parameter. (left) width of 100. (middle) width of 50. (right) width of 1.

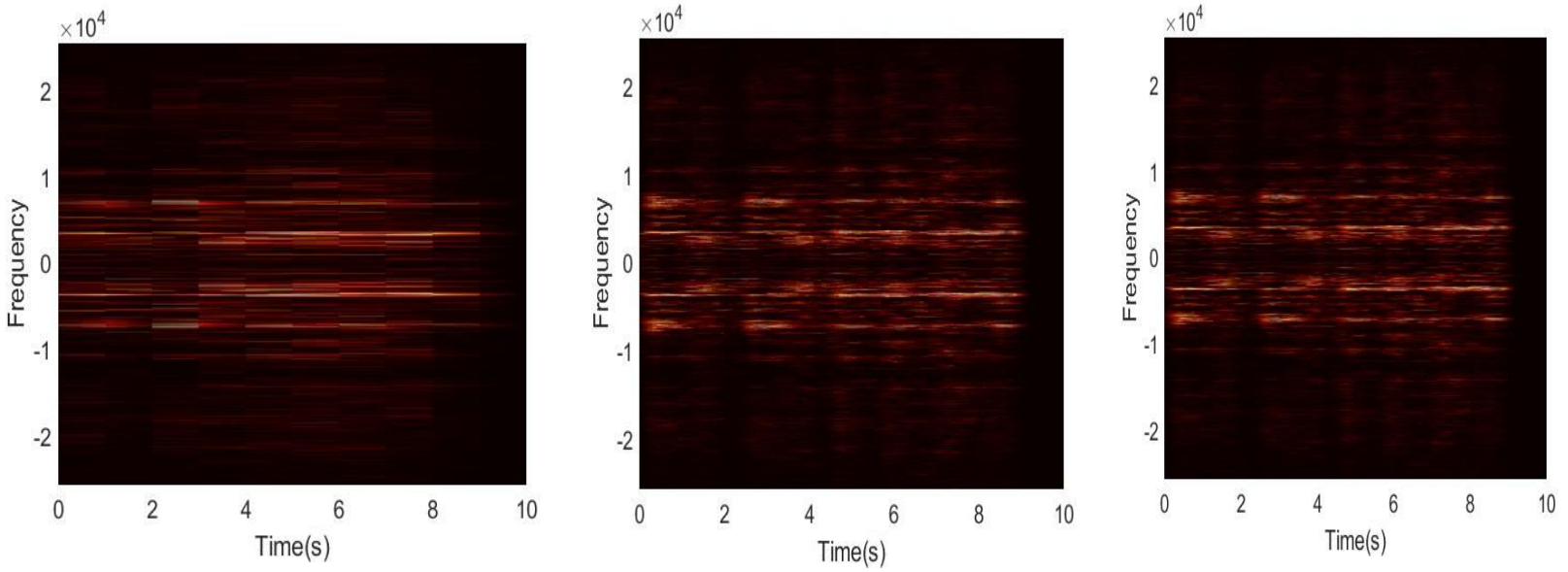


Figure 6: Comparison of different translation for Gaussian filter with same width (dilation). (Left) translation 0.5. (Middle) translation 0.1. (Right) translation 0.01.

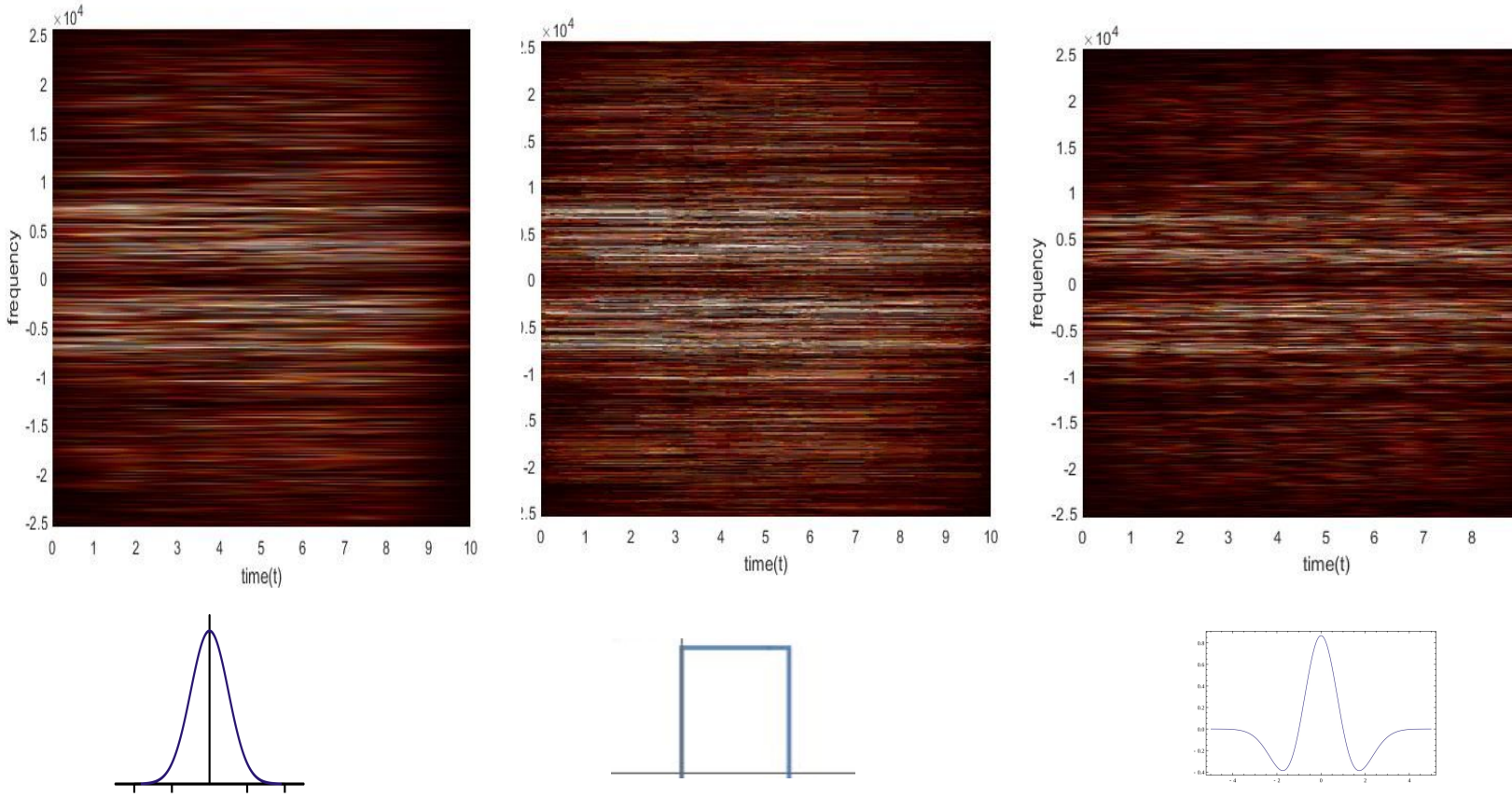


Figure 7: Comparison of 3 different filters with same dilation and translation factors. (Left) Gaussian filter, (middle) Shannon filter, (right) Mexican Hat filter. Each filter can extract unique time and frequency features.

Section 3.2: Application in retrieving music song notes from a piano recording

With Gabor transform, we can better understand signal in both time-frequency resolution, in particular, we can localize which frequency occur at which exact time step, provided with using approximately correct parameter for the filter. In fact, being able to identify the occurrences of signal is important for many applications such as radar detection, sound classification. In this paper, we will discuss one application of using Gabor transform to identify sound frequency from a piano recording and from there, recreate the music sheet for that acoustic song. We will use a piece of recording "Mary had a little lamp" and analyze its components.[4]

We will apply the same filtering techniques and retrieved the frequencies at each timestep. However, it is worth noting that in this case, we need to convert to sound frequencies instead of angular frequency from Fourier domain, and as a result, convert the frequencies to piano music notes. By comparing the same song from two different instruments, we can easily distinguish those by each range of frequencies, for instance, piano notes are lower in frequencies

[4] "Basic Computation & Visualization - University of Washington." https://faculty.washington.edu/kutz/KutzBook/KutzBook.html. Accessed 9 Feb. 2019.

(200 ~ 300Hz) compared with flute notes at around 800 ~ 1000Hz. Furthermore, we can see the effect of timbre, also known as overtones in instruments, when we analyze the Fourier transform plot of each signal. In particular, beside each spike in each timestep, there are smaller spikes around which are considered overtones generated by the instrument for a center frequency. Each instrument has its own distinguished patterns of overtones.

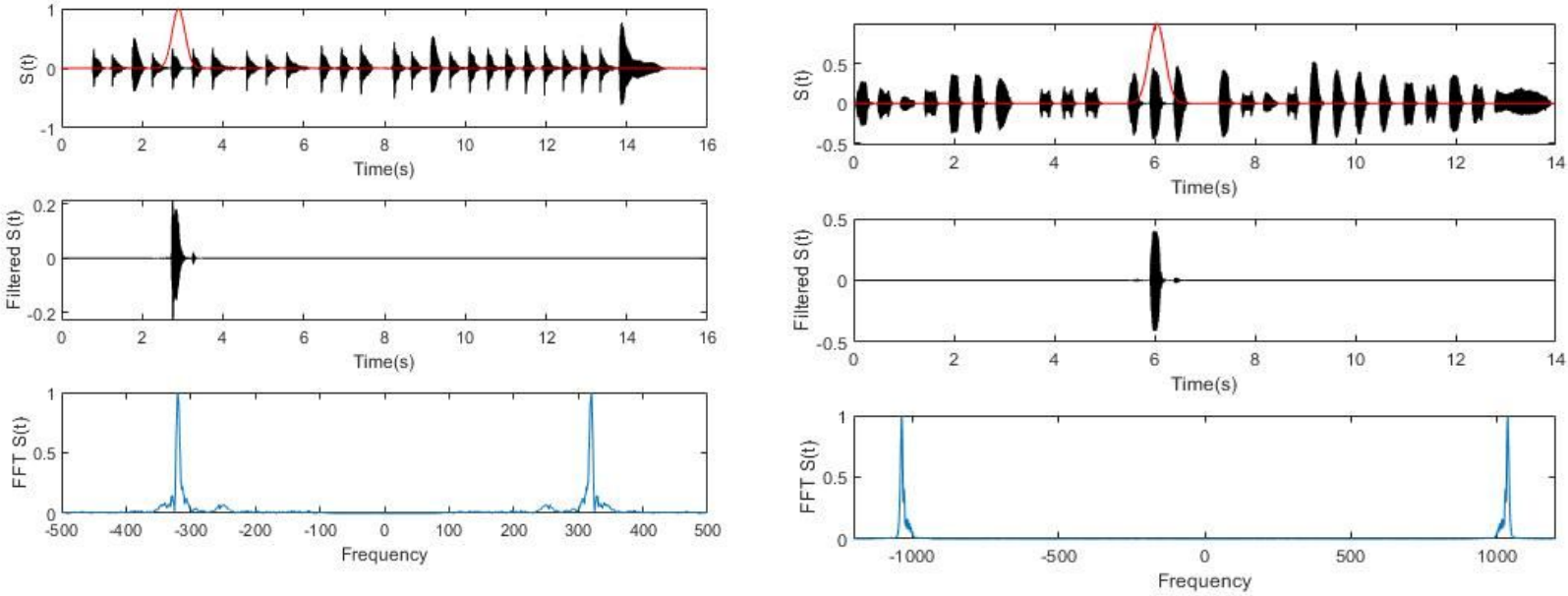


Figure 8: One time step of Gabor transform upon piano signal (left) and recording signal (right).

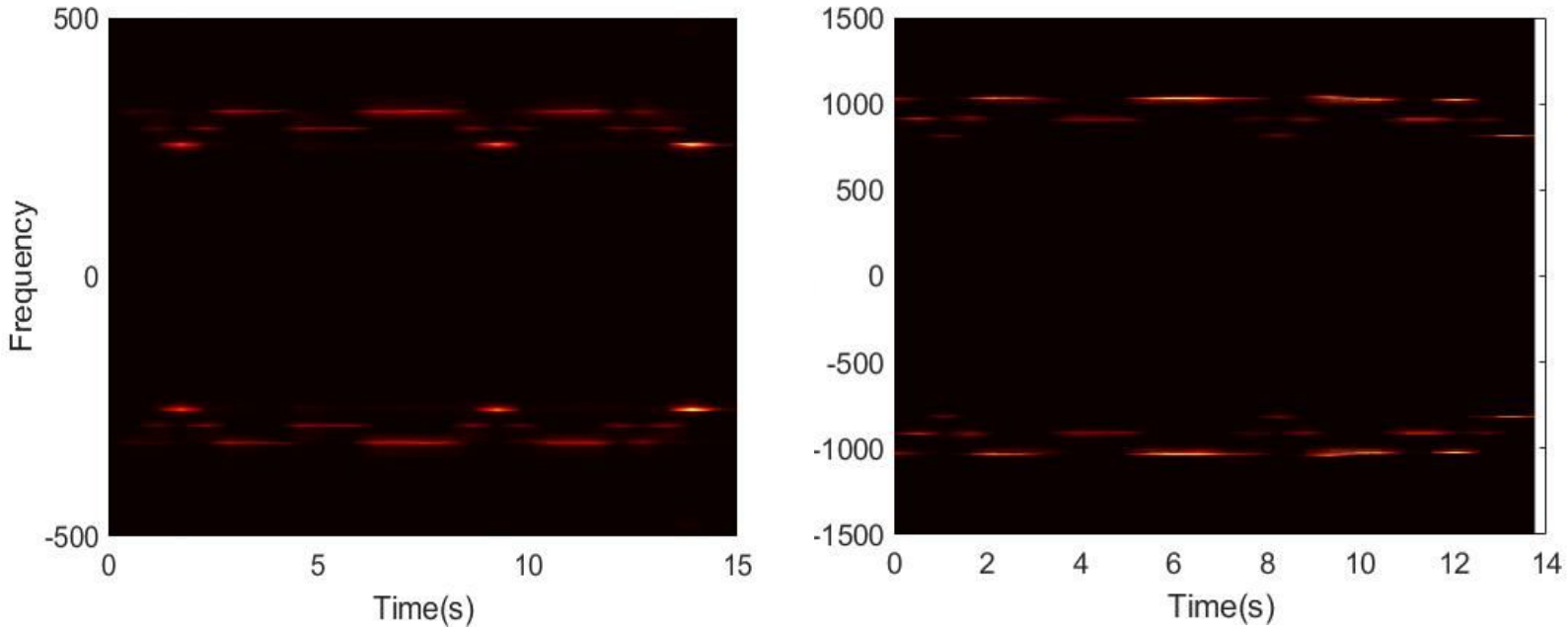


Figure 9: Spectrogram of piano signal (left) and recording signal(right).

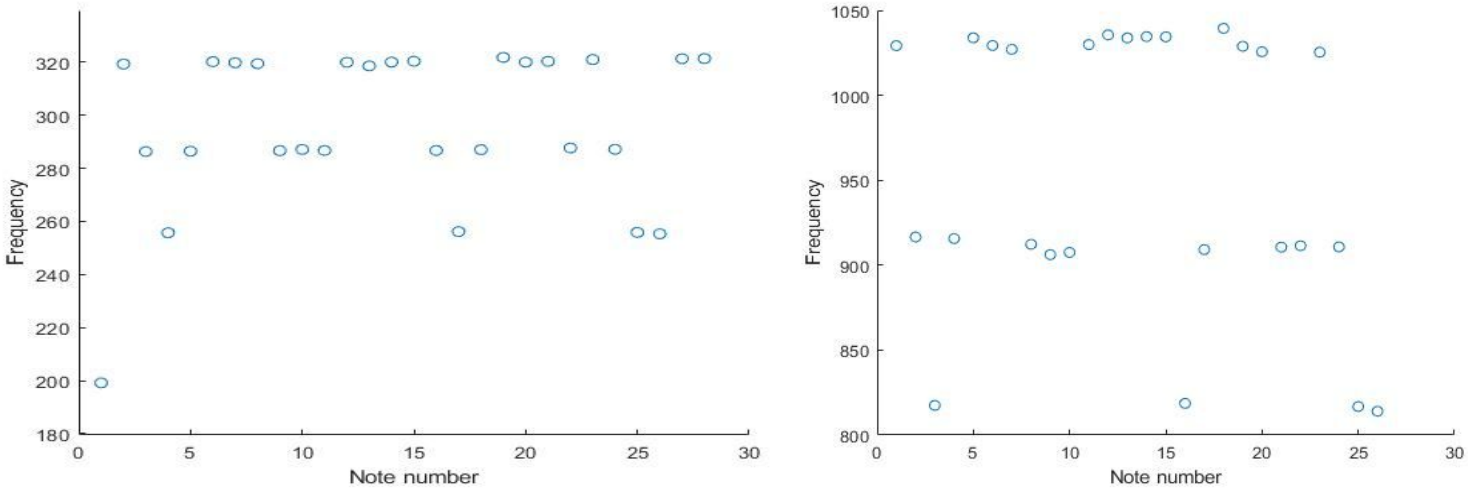


Figure 10: Scatter plot of musical notes occur in the song time period of piano (left) and recording(right). Both successfully extract out the similar tempo and overall rhythms compared with the original song.

Section 4: Computational Results

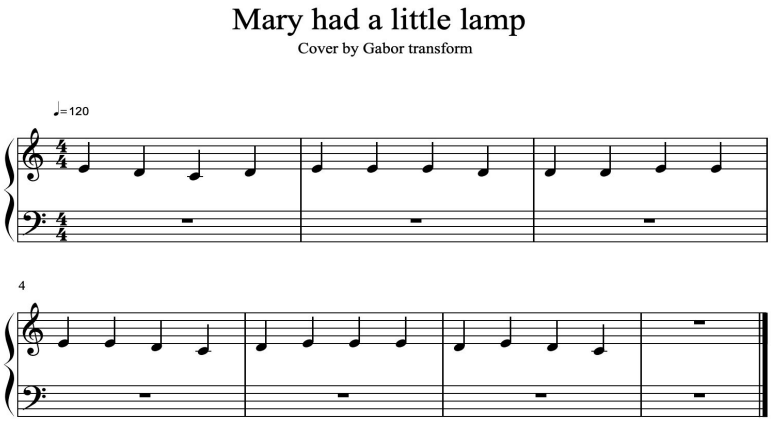


Figure 11: Music sheet recover from piano recording.

Section 5: Summary and Conclusions

This paper discusses advantages of Gabor transform and its modification to plain Fourier transform to perform time-frequency analysis. In particular, Gabor transform and its idea of sliding window across timesteps is powerful to help us understand better about the signal's properties at each timestep. This technique provides better tools in various applications that require feature extraction and time localization such as radar tracking, audio analysis, image processing. This technique helps to identify and compare data by discovering and utilizing hidden structure in signal and has been widely applied in many noticeable fields such as face recognition and texture classification.

Appendix A: MATLAB functions used and brief implementation explanation

```
g=exp(-width*(t-tslide(j)).^2); % Gaussian Equation
```

Appendix B: MATLAB Codes

```
%Section 3.1: Spectrogram analysis
clear all; close all; clc
L=9;
load handel
S = y'/2;
S = S(1:end - 1);
% p8 = audioplayer(S,Fs);
% playblocking(p8);
n=length(S);
t2=linspace(0,L,n+1); t=t2(1:n);
k=(2*pi/L)*[0:n/2-1 -n/2:-1]; ks=fftshift(k);

St=fft(S);
figure(1)
subplot(2,1,1) % Time domain
plot(t,S,'k')
set(gca,'Fontsize',[14]),
xlabel('Time (t)'), ylabel('S(t)')
subplot(2,1,2) % Fourier domain
plot(ks,abs(fftshift(St))/max(abs(St)),'k');
set(gca,'Fontsize',[14])
xlabel('frequency (\omega)'), ylabel('FFT(S)')
%%Gabor transform
figure(3);
Sgt_spec=[];
tslide = 0:0.01:10;
width = 25;
for j=1:length(tslide)
    g=exp(-width*(t-tslide(j)).^2); % Gaussian %width1 is the largest
gaussian while width 100 is the thinnest one
    Sg=g.*S;
    %Shanon
    %shannon = (abs(t-tslide(j)) <= width/2);
    %Sg = shannon .* S;
    %mexican = (1 - t.^2) .* exp(- (t.^2) /2)
    %mexican = (1 - (t - tslide(j)).^2) .* exp(-width * ((t -
tslide(j)).^2) /2)
%     center = (t - tslide(j)) .^ 2;
%     mexicanHat = (1 - center) .* (exp(-1 * center/2));
%     Sg = mexicanHat .* S;
    Sgt=fft(Sg);
    Sgt_spec=[Sgt_spec; abs(fftshift(Sgt))];
    subplot(3,1,1), plot(t,S,'k',t,g,'r')
    %subplot(3,1,1), plot(t,S,'k',t,shannon,'r')
%     subplot(3,1,1), plot(t,S,'k',t,mexicanHat,'r')
    subplot(3,1,2), plot(t,Sg,'k')
```

```
    subplot(3,1,3), plot(ks,abs(fftshift(Sgt))/max(abs(Sgt)))
    drawnow
    pause(0.1)
end
%Spectrogram
figure(2);
pcolor(tslide,ks,Sgt_spec.'), shading interp
set(gca,'Fontsize',[14])
colormap(hot)
%Section 3.2: Music note classification
close all; clear all; clc;
tr_piano=16; % record time in seconds
y_piano=audioread('music1.wav'); Fs_piano=length(y_piano)/tr_piano;
y_piano = y_piano';
%FFT
L = tr_piano;
n = length(y_piano)
t2=linspace(0,L,n+1); t=t2(1:n);
k=(2*pi/L)*[0:n/2-1 -n/2:-1]; ks=fftshift(k);% ks: # of cycles in 16 sec
hz_piano_frequency = ks / (2 * pi); % of cycles in 1 sec = hertz
figure(1)
Sgt_spec=[];
tslide = 0:0.58:tr_piano
frequencies_in_time = [];
for j=1:length(tslide)
    g=exp(-20*(t-tslide(j)).^2); % Gabor
    Sg=g.*y_piano; Sgt=fft(Sg);
    max_frequency = max(abs(fftshift(Sgt)));
    a = abs(fftshift(Sgt));
    max_index = find(abs(fftshift(Sgt)) == max_frequency);
    piano_frequency_at_current_time = hz_piano_frequency(max_index);
    frequencies_in_time = [frequencies_in_time;
piano_frequency_at_current_time];
    Sgt_spec=[Sgt_spec; abs(fftshift(Sgt))];
    subplot(3,1,1), plot(t, y_piano,'k',t,g,'r')
    subplot(3,1,2), plot(t,Sg,'k')
    subplot(3,1,3),
plot(hz_piano_frequency,abs(fftshift(Sgt))/max(abs(Sgt)))
    axis([-500 500 0 1]);
    drawnow
    pause(0.1)
end
%Spectrogram
figure(2);
pcolor(tslide,hz_piano_frequency,Sgt_spec.'), shading interp
axis([0 16 -500 500]);
set(gca,'Fontsize',[14])
colormap(hot)
% figure(3);
%spectrogram(y_piano, Fs_piano);
```

```
figure(4);
scatter(1:length(tslide), frequencies_in_time(:, 2));
%Piano sheet
tr_rec=14; % record time in seconds
y_rec=audioread('music2.wav'); Fs_rec=length(y_rec)/tr_rec;
%FFT
y_rec = y_rec';
L = tr_rec;
n = length(y_rec);
t2=linspace(0,L,n+1); t=t2(1:n);
k=(2*pi/L)*[0:n/2-1 -n/2:-1]; ks=fftshift(k);% ks: # of cycles in 16 sec
hz_rec_frequency = ks / ( 2 * pi); % of cycles in 1 sec = hertz
figure(5)
Sgt_spec=[];
tslide = 0:0.55:tr_rec
frequencies_in_time = [];
for j=1:length(tslide)
    g=exp(-20*(t-tslide(j)).^2); % Gabor
    Sg=g.*y_rec; Sgt=fft(Sg);
    max_frequency = max(abs(fftshift(Sgt)));
    a = abs(fftshift(Sgt));
    max_index = find(abs(fftshift(Sgt)) == max_frequency);
    rec_frequency_at_current_time = hz_rec_frequency(max_index);
    frequencies_in_time = [frequencies_in_time;
rec_frequency_at_current_time];
    Sgt_spec=[Sgt_spec; abs(fftshift(Sgt))];
    subplot(3,1,1), plot(t, y_rec,'k',t,g,'r')
    subplot(3,1,2), plot(t,Sg,'k')
    subplot(3,1,3),
plot(hz_rec_frequency,abs(fftshift(Sgt))/max(abs(Sgt)))
    axis([-500 500 0 1]);
    drawnow
    pause(0.1)
end
%Spectrogram
figure(6);
pcolor(tslide,hz_rec_frequency,Sgt_spec.'), shading interp
axis([0 14 -1000 1000]);
set(gca, 'Fontsize',[14]);
colormap(hot)
%%
figure(7);
scatter(1:length(tslide), frequencies_in_time(:, 2));
%Piano sheet
```