Fourier Transforms for Object Detection:
Tracking a Target in Noisy Ultrasound Data

Franklin Williams
https://github.com/streater512/AMATH-582
March 12, 2020

Abstract
Fourier transforms are a key method to an array of problems within the physical, biological, and engineering sciences. In this report, the Fast Fourier transform and frequency filtering are utilized to track a marble (target) as it passes through a dog's small intestines. The target is tracked through time and the final position is determined so the veterinarian can remove the target from the dog's small intestines.

Section I: Introduction
This report begins with an introduction to the practicality of Fourier transforms as well as an overview of data collected and experiments run. In Section II, a theoretical background of the techniques used will be discussed. Section III covers the algorithm implementation and development in MATLAB. Section IV provides the computational results of our tests. To conclude, Section V summarizes the results and ends with final remarks.

The Fourier transform revolutionized the field of signal processing. The key concept behind the applications of the transform is to analyze and manipulate data within the frequency domain. One useful technique is to utilize various frequency (band pass) filters to highlight central frequencies. In real world applications and experiments, high levels of noise can be expected to be collected along with a true signal. Noise can be generated from imperfect data sensors or other externalities. If models are built assuming the data collected is a combination of true data plus Gaussian White Noise (GWN), the central frequency can be extracted by averaging the frequency signatures. Because GWN has an expected value of zero, averaging frequency signatures will reduce the noise component of the collected data to reveal the true central frequencies collected.

The data used in this report were taken from the course website. Twenty samples of ultrasound data were collected by a veterinarian inspecting a dog's small intestines. Because of the fluids and movement of the dog, the data contains large amounts of noise and no clear signature of the target can be immediately seen. The Multidimensional Fast Fourier transform is utilized to view the data within the frequency domain. The twenty samples are then averaged to reduce noise and denote the central frequency generated by the target. Using inputs from the central frequency, a Gaussian filter is applied to the frequency samples and the Inverse Fast Fourier transform is applied to generate x-y-z-coordinates of the target.

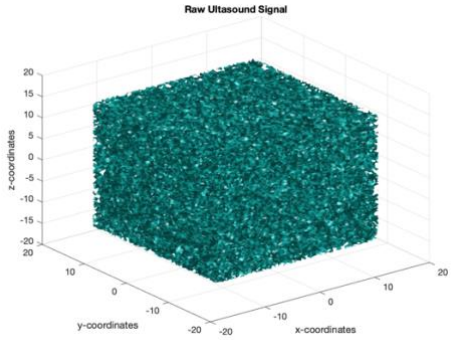

Figure (1): Raw Ultrasound Signal Example

Section II: Theoretical Background

Fourier introduced the representation of a function by a series of sines and cosines:

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} (a_n \cos nx \; + \; b_n \sin nx) \; x \in (-\pi, \pi] \tag{1}$$

The complex version of the expansion produces the Fourier series on the domain $x \in [-L, L]$:

$$f(x) = \sum_{-\infty}^{\infty} c_n e^{in\pi x/L}, \qquad x \in [-L, L] \tag{2}$$

The Fourier transform is an integral transformed defined over the entire line $x \in [-\infty, \infty]$. The transform is defined by:

$$F(k) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-ikx} f(x) dx \tag{3}$$

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{ikx} F(k) dk \tag{4}$$

It should be noted that the formal definition is over the real number line, while in real-world applications the transform lies on the domain, $x \in [-L, L]$.

A simple example of a filter is the Gaussian filter:

$$\mathcal{F}(k) = e^{-\tau(k - k_o)^2} \tag{5}$$

This one-dimensional Gaussian filter can be applied to the transformed signals to give more weight to frequencies that are near to the central frequency determined by k_o.

Section III: Algorithm Implementation and Development

The Fast Fourier transform (FFT) was developed to perform the forward and back-ward (inverse) Fourier transform by Cooley and Tukey in the mid-1960s. It has a low operation count: $O(N \log N)$, finds the transform on an interval, $x \in [-L, L]$, and implies the solution on this finite interval have period boundary conditions. Finally, it has accuracy properties well beyond that of standard discretization schemes.

Sixty-four modes were used to transform the spatial domain into a period periodic domain. For each sample of data, the 3-Dimensional FFT was performed to transform the spatial data into frequency data. The transformed data were averaged, and the central frequencies along the three dimensions were found.

The central frequencies along each dimension were input into a 3-dimensional Gaussian filter centered at each central frequency. The transformed signals were then multiplied by the created filter and the inverse of the product was determined. The positions corresponding to the maximum value of the output were found. These positions correspond to the coordinates of the target at each time-step.

Section IV: Computational Results

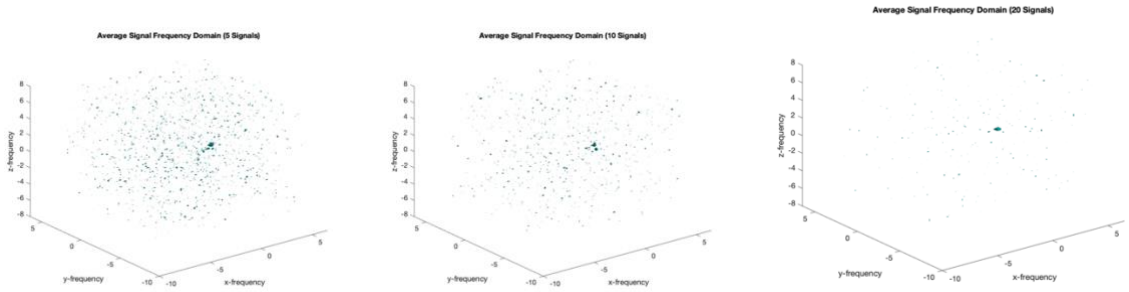

Figure (2): Averaged Frequency Signal Example

Figure (2) demonstrates the power of averaging along the frequency domain. The far-left plot shows the average frequencies determined by the FFT using the first five sample. While there appears to be a stronger signal coming from the frequencies in the middle of the plot, it is surrounded by many instances of less prominent frequencies generated by the noise. The middle plot shows the same strong frequency, and has far fewer instances of noise interfering with the central frequency. On the far-right, the frequencies generated by the noise are now sparse and a much clearer central frequency can be seen.

The central frequency of the target is:

$$\begin{aligned} k_x &= -4.82 \\ k_y &= 5.65 \\ k_z &= -6.70 \end{aligned} \tag{6}$$

Resulting in the following Gaussian filter:

$$\mathcal{F}(k) = e^{-\tau\left((k-k_x)^2+(k-k_y)^2+(k-k_z)^2\right)} \tag{6}$$

Using this central frequency within the Gaussian filter, and tracking the location of the marble generates the following target path. The below image shows that as time passes, the target appears to be moving in a downward spiral through the dog's small intestines. The veterinarian should make attempts to remove the target at the x-y-z coordinate, (-6.09, 4.22, -6.09).

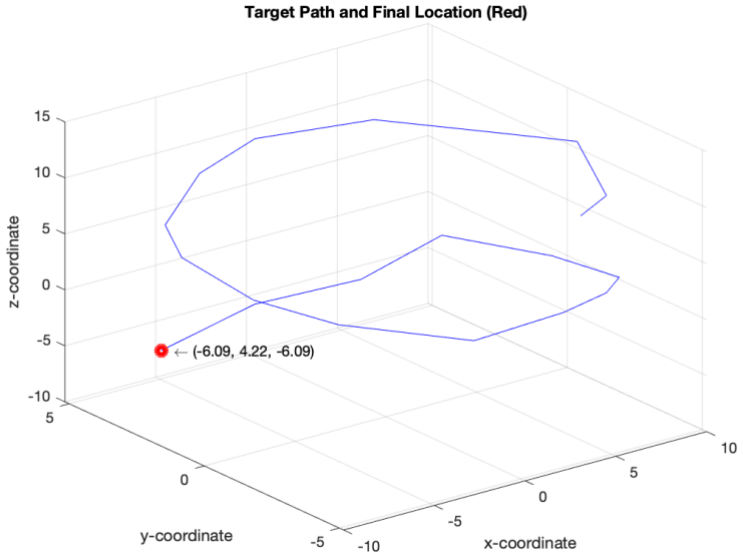

Figure (3): Moving target and position of final position for extraction

Section V: Summary and Conclusion

In this report, the practicality of the FFT, frequency averaging, and spectral filtering is shown. The central frequency was determined by averaging the transformed raw signals. After the Gaussian filter was applied to the path of the target was easily determined. Comparing Figure (1) to Figure (3) demonstrates the power of the techniques applied in this report. The raw signals generated from ultrasound data give little to no information on the location of the target. Careful filtering allowed the veterinarian to find the signal within the noise. To further improve on results, other filters and more data should be utilized to determine if a more exact location can be tracked.

Appendix A:
Functions Implemented

3-Dimensional Fast Fourier Transform of rawdata, Un - `fftn(Un);`

Gaussian Filter: `filter = exp(-2*pi*((Kx - fx).^2 + (Ky - fy).^2 + (Kz - fz).^2));`

Filtered transformed data: `Untf = fftn(Un).* filter;`

Inverse Fast Fourier Transform of filtered data, Untf - `ifftn(Untf);`

Appendix B:

```
clear all; close all; clc;
load Testdata
L=15; % spatial domain
n=64; % Fourier modes
x2=linspace(-L,L,n+1); x=x2(1:n); y=x; z=x;
k=(2*pi/(2*L))*[0:(n/2-1) -n/2:-1]; ks=fftshift(k);
[X,Y,Z]=meshgrid(x,y,z);
[Kx,Ky,Kz]=meshgrid(ks,ks,ks);

% plot raw signal for figure(1)
Un(:,:,:)=reshape(Undata(1,:),n,n,n);
close all, isosurface(X,Y,Z,abs(Un),0.4);
axis([-20 20 -20 20 -20 20]), grid on, drawnow;
xlabel('x-coordinates');
ylabel('y-coordinates');
zlabel('z-coordinates');
title('Raw Ultasound Signal');


%%
% plot average signal at 5, 10, 20 for figure(2)
Uave = zeros(n,n,n);
for j=1:20
Un(:,:,:)=reshape(Undata(j,:),n,n,n);
Uave = Uave + fftn(Un);
end
Uave = Uave/20;
close all, isosurface(Kx,Ky,Kz,abs(Uave)/max(abs(Uave(:))),0.6);
% axis([-20 20 -20 20 -20 20]), grid on, drawnow;
xlabel('x-frequency');
ylabel('y-frequency');
zlabel('z-frequency');
title('Average Signal Frequency Domain (20 Signals)');


%% filter marble frequency
[max_val, position] = max(abs(Uave(:)));
[x, y, z] = ind2sub(size(Uave),position);
fx = Kx(x, y, z);
fy = Ky(x, y, z);
fz = Kz(x, y, z);
filter = exp(-2*pi*((Kx - fx).^2 + (Ky - fy).^2 + (Kz - fz).^2));

%%
target=zeros(3,20);
for j=1:20
    Un = reshape(Undata(j,:),n,n,n);
    Untf = fftn(Un).* filter;
    Unf = ifftn(Untf);
    [max_value, position] = max(abs(Unf(:)));
    [marblex, marbley, marblez] = ind2sub([n n n], position);
    target(1,j) = X(marblex, marbley, marblez);
    target(2,j) = Y(marblex, marbley, marblez);
    target(3,j) = Z(marblex, marbley, marblez);
end
```

```
figure()
plot3(target(1,:), tar2get(2,:), target(3,:),'b'), grid on;
hold on;
plot3(target(1,20), target(2,20), target(3,20),'r-o','LineWidth',[25]);
text(target(1,20), target(2,20), target(3,20), '   \leftarrow (-6.09, 4.22, -6.09)');
xlabel('x-coordinate');
ylabel('y-coordinate');
zlabel('z-coordinate');
title('Target Path and Final Location (Red)');
```