


COM3536 Project

Name, Surname: Elvin Huseynli - Muaz Kartal
Student Number: 20290122 - 20290349

Date: 22/05/2022
Sign: 

1. Introduction

The main objective of this project is to implement the idea learned in this course and use Image Processing and Signal Processing tools in MATLAB to process some data in different types (like bitmap images and png sources) and implement Fourier Transforms of those images using graphs, using different techniques.

2. Tasks

In this section, we are going to write MATLAB code to apply different filters, based on some requirements. Using built-in functions in MATLAB, we are required to apply formulas and display the filtered images and frequency-response functions of them.

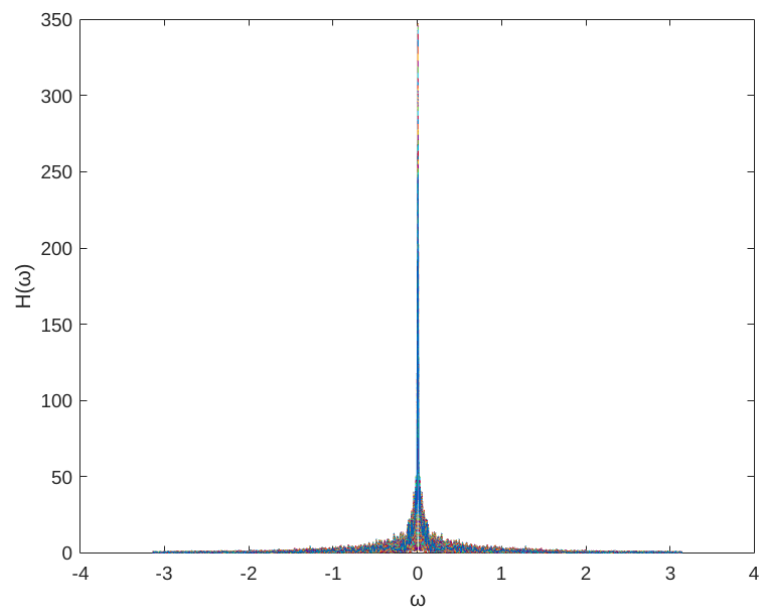
2.1. Task 1

In this task we have to choose one optional bitmap image and apply M -point averaging filter to it. As the formula is provided in the Project hints, we have to implement it for different values of M . After creating filtered images, we have to calculate the magnitude of the frequency response function according to the filter we used. The filtered images and plotted functions are displayed below in order. For the following implementations, "bridge.bmp" image is utilized.

- **$M = 11$:**
Filtered image of "bridge.bmp":



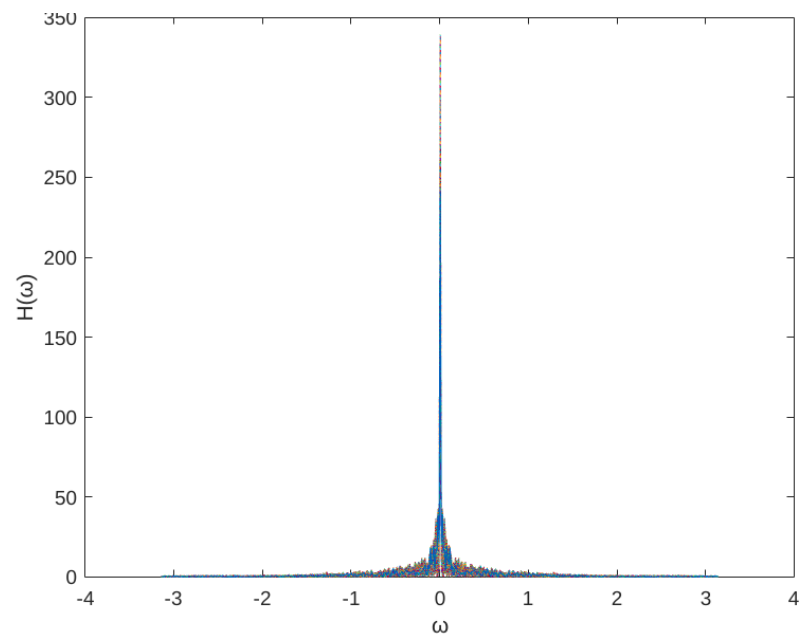
The magnitude of the frequency response function:



- **M = 31:**
Filtered image of “bridge.bmp”:



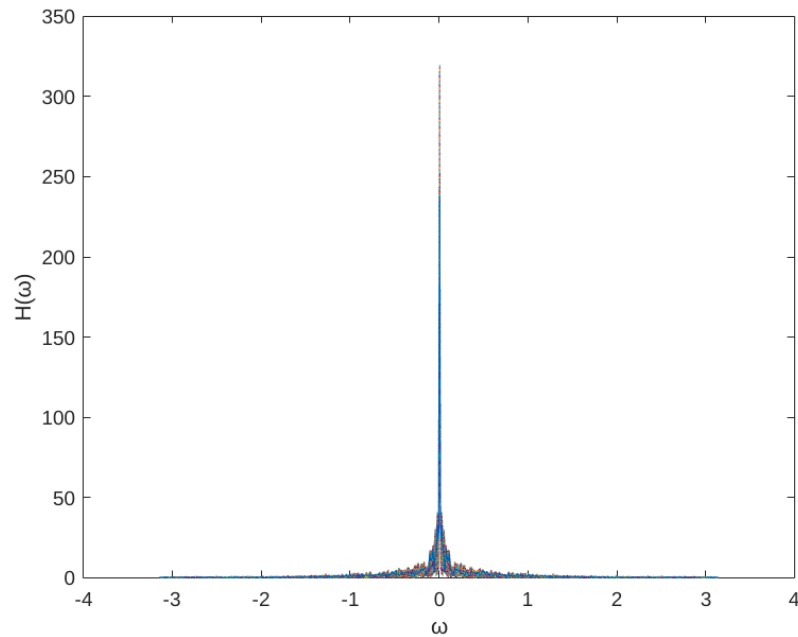
The magnitude of the frequency response function:



- **M = 61:**
Filtered image of “bridge.bmp”:



The magnitude of the frequency response function:



Questions:

- i) It looks like a motion blur effect; as if someone took a picture of the bridge while walking, quick stepping and running.
- ii) As the filter is applied to the image we chose, the details get less ostensible. While the value of **M** increases, the details get less conspicuous.
- iii) There is a horizontal difference in the filter we applied, there is no change in vertical alignment.
- iv) As the filter is applied to the image, the waveform gets more steady as the averaging value of the filter increases.
- v) Close to the edges of the image's blackness increases first, but as the value of *m* increases, its density decreases at one point, and its width starts to grow.

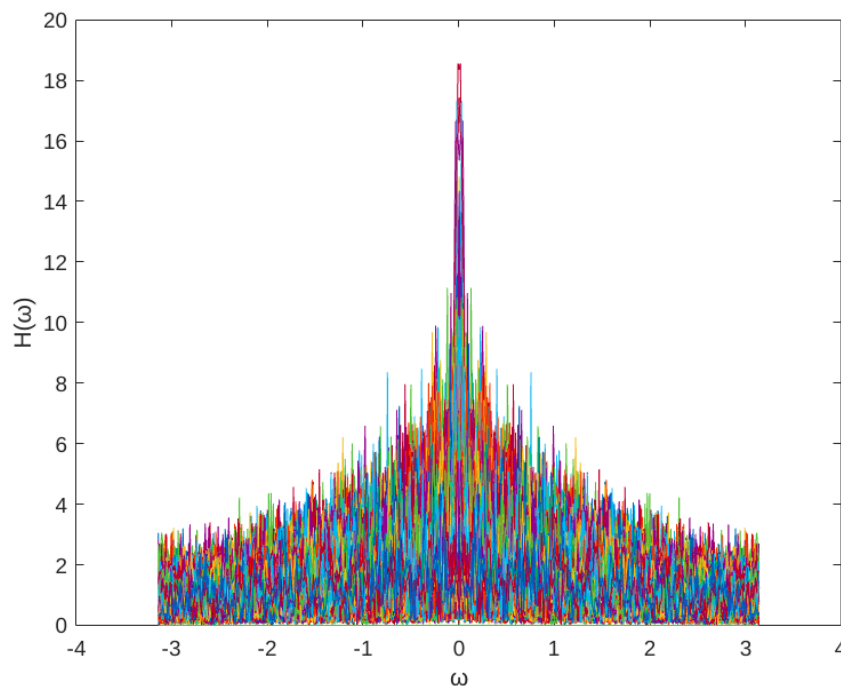
2.2. Task 2

In this task we have to choose same bitmap image used in the first task and apply the **First Difference** filter to it. As the formula is provided in the Project hints, we have to implement it on the image. After creating the filtered image, we have to calculate the magnitude of the frequency response function according to the filter we used. The filtered image and plotted function are displayed below. For the following implementations, “bridge.bmp” image is utilized.

Filtered image of “bridge.bmp”:



The magnitude of the frequency response function:



Questions:

- i) The filter looks like an edge detection filter, but the density and stroke of edges are not complete.
- ii) Theoretically, we expected that only the pixels which have a difference between the brightness of the adjacent pixels and their own brightness would appear as clear as the ratio of the difference. In contrast, the other pixels would be black. The filter worked in line with our expectations.

- iii) While the vertical lines appeared after the filter, the horizontal lines almost disappeared and couldn't be detected by the filter.
- iv) Frequency response values increase as the First Difference filter is applied. It affects the graph in a way that it gets more unstable while the filter is applied on the frequency values.

3. Tasks

i) Task 1

```
image = imread('bridge.bmp');
grayImage = mat2gray(image, [0 255]);
[x, y] = size(grayImage);
m = 11; %The default value is 11. It can be changed.
mPointImage = ones(1, m)/m;
result = imfilter(grayImage, mPointImage);
fourierTransform = fft(result);
fourierTransform = fftshift(fourierTransform);
xAxis = linspace(-pi, pi, y);
imshow(result);
plot(xAxis, abs(fourierTransform));
xlabel('ω');
ylabel('H(ω)');
```

ii) Task 2

```
a = imread("bridge.bmp");
j = mat2gray(a,[0 255]);
c = mat2gray(a,[0 255]);
M = 1;
for k = 1:512
    for l = 1:512
        if (l-M>0)
            c(k,l) = j(k,l)-j(k,l-1);
        else
            c(k,l) = 0;
        end
    end
end
fourierTransform = fft(c);
fourierTransform = fftshift(fourierTransform);
xAxis = linspace(-pi, pi, 512);
imshow(c);
plot(xAxis, abs(fourierTransform));
xlabel('ω');
ylabel('H(ω)');
```