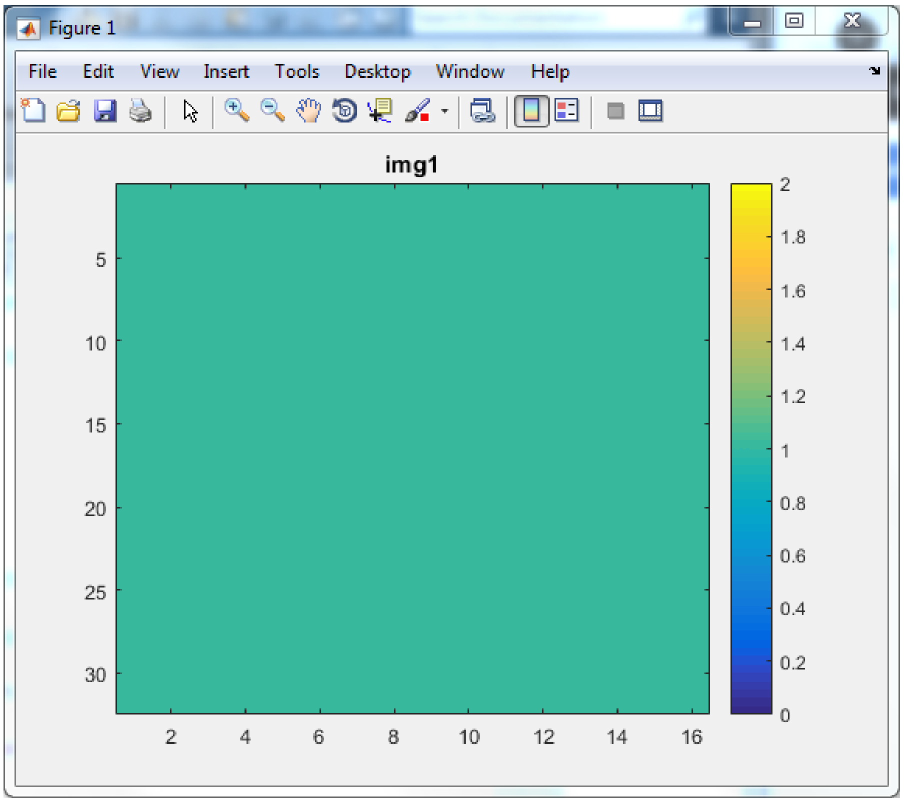
Date: 23.05.2022 8:15 - 12:00
Author: Sławomir Batruch

EDISP - Lab 6
Image Processing

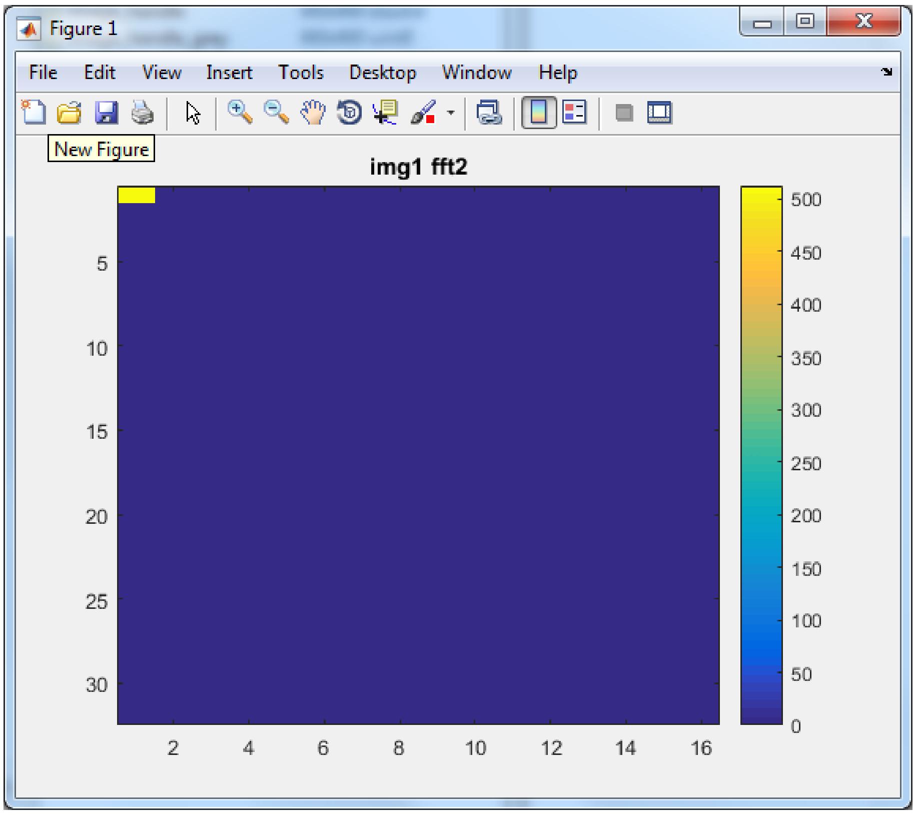
Task 1

Spectrum of a 2D signal

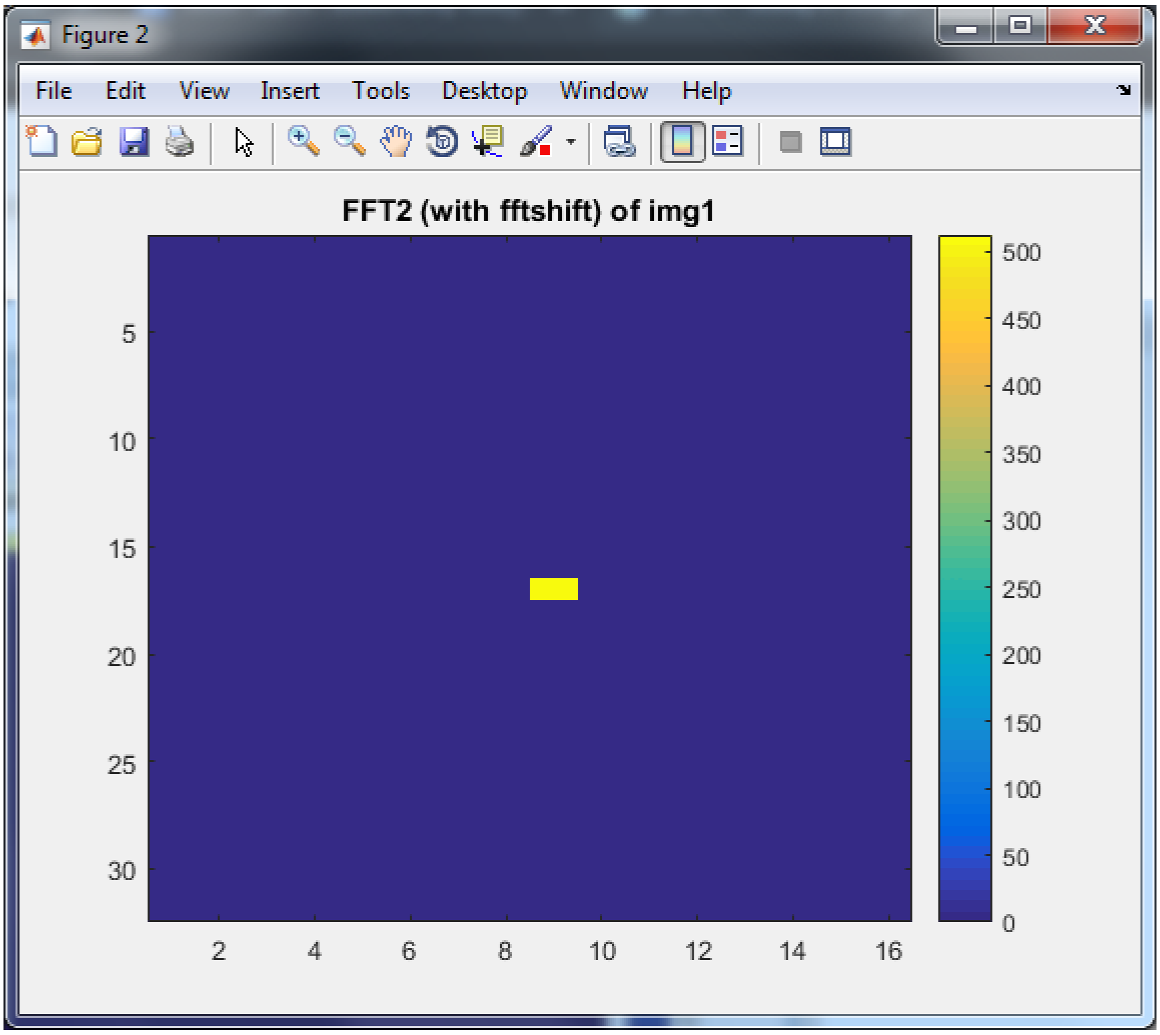
We begin by creating an all-white image of size 32x16 px. Then, we display this image and calculate 2d FFT of it.



Then we can obtain "raw" fft2 of this image to be. Yellow on image is = 512:

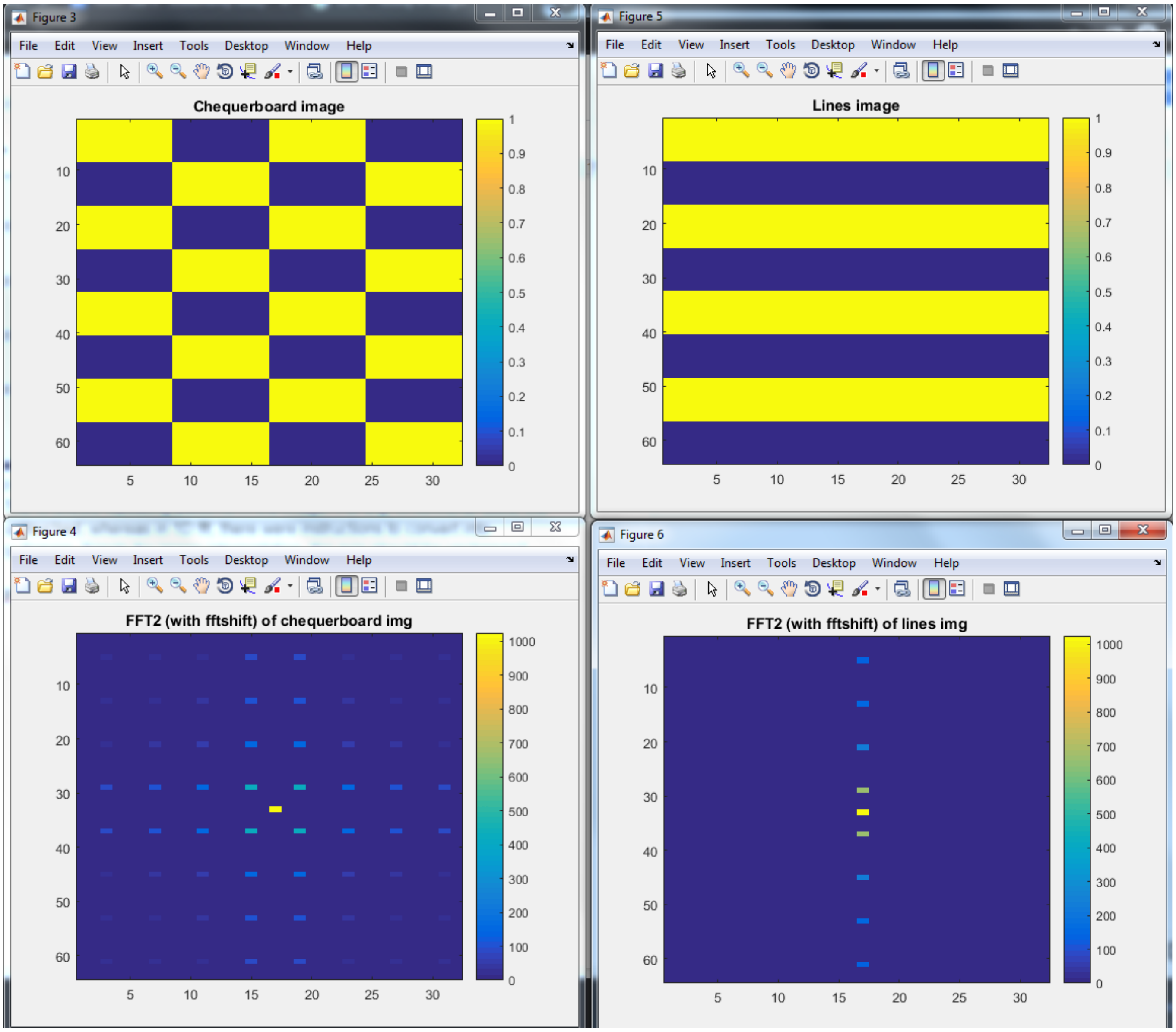


After using fftshift, we obtain:



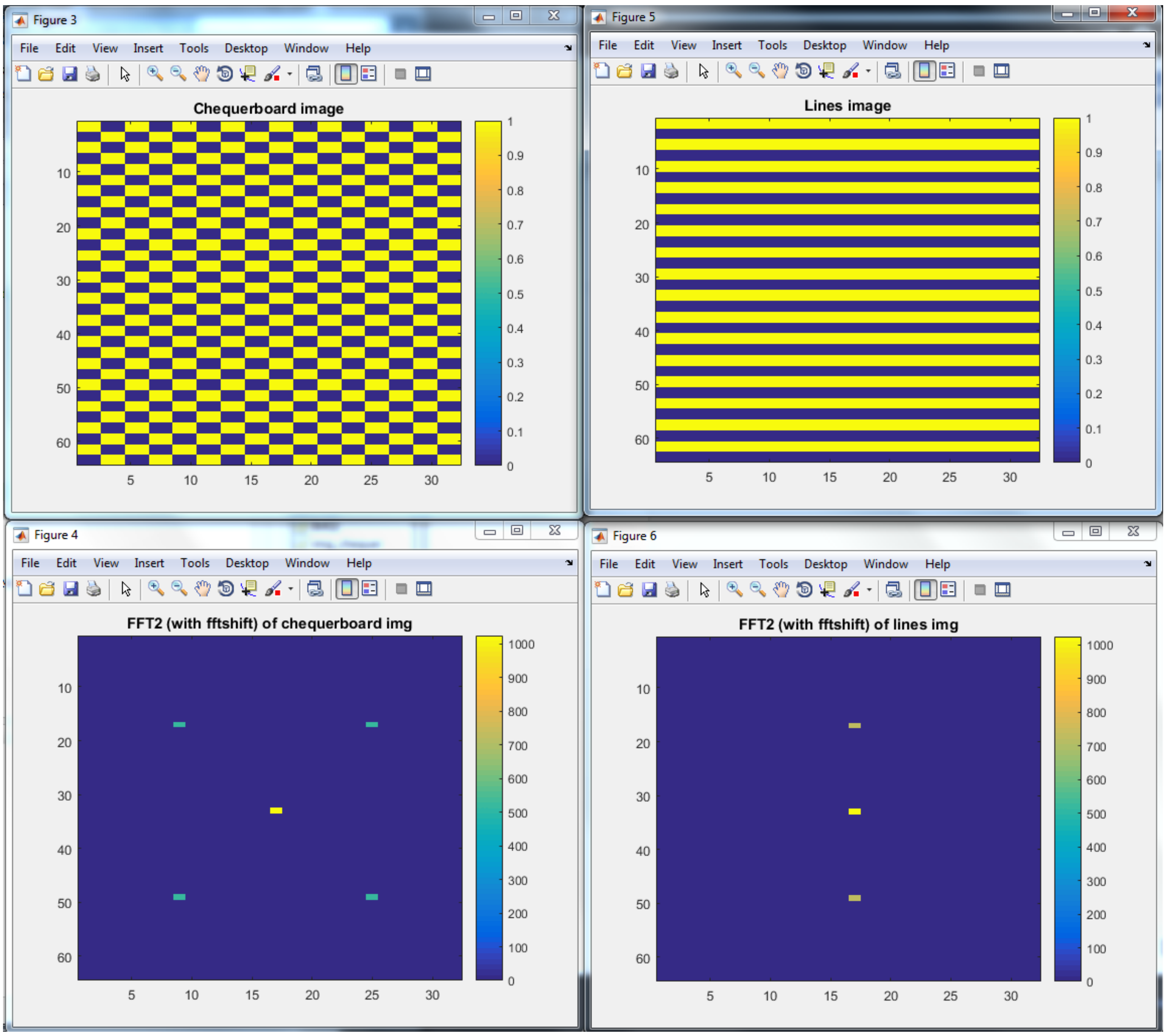
It is hard to theoretically agree to the labeling of the axis, due to the nature of 2D signals. The fft shows repeating patterns in the image. For e.g. in MATLAB docs, the axis are untouched, whereas in 1D fft, there were instructions to convert into frequency domain. Simply for image interpretation, the axis should be unchanged.

Then, I generated chequerboard and line images, as well as their 2d ffts. The setup for image generation was for stand number 2. 64;32;8 M;N;MS:

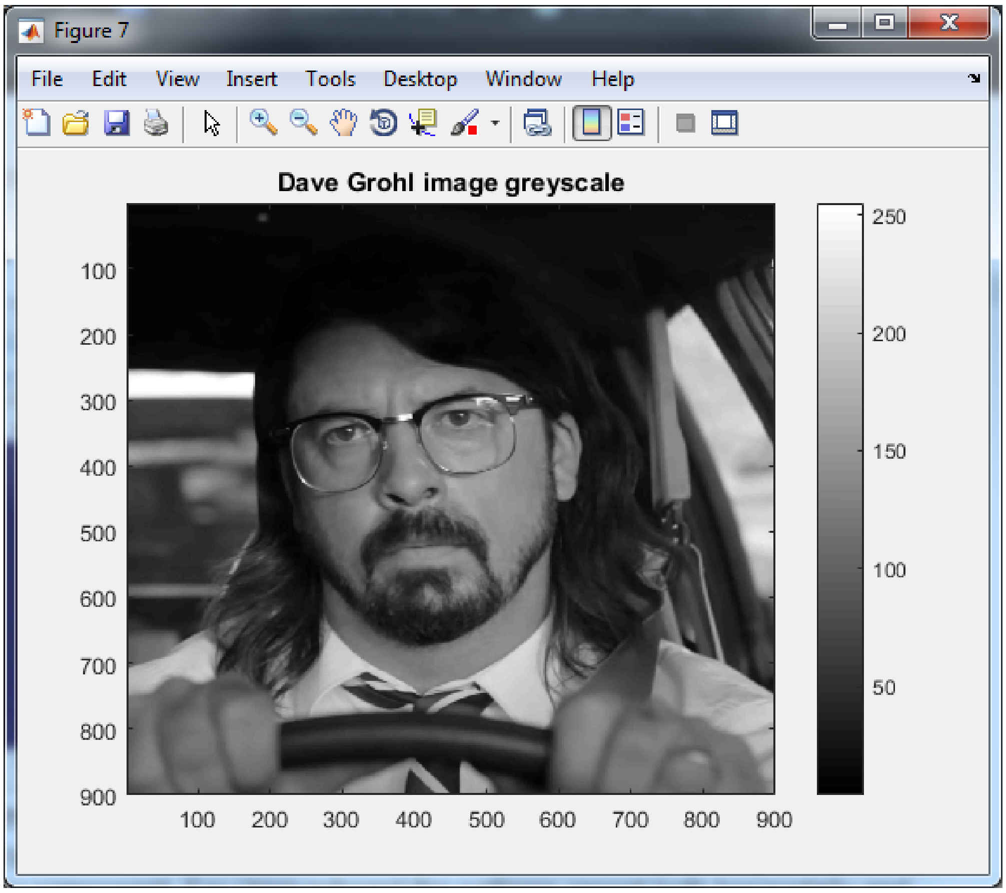


We see that for the chequerboard image we get both vertical and horizontal frequency components on the 2d fft, whereas for the lines image, we get only one frequency component. For chequerboard the patterns repeat both horizontally and vertically, whereas for the lines image it is only vertically. That is why we only see frequency components on the vertical axis.

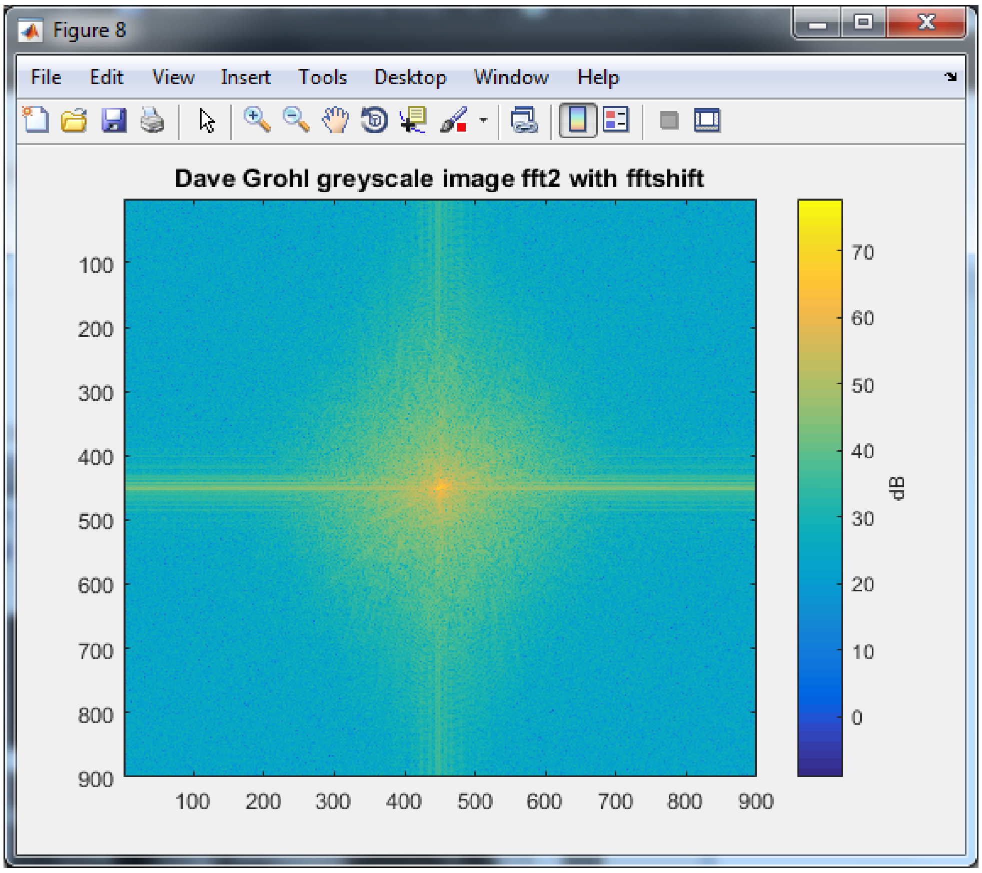
But if we narrow the pixels (increase the spatial frequency), we obtain less frequency components:



Then, I computed a 2d fft of a real life image. The image being:



Which resulted in the following 2d fft:

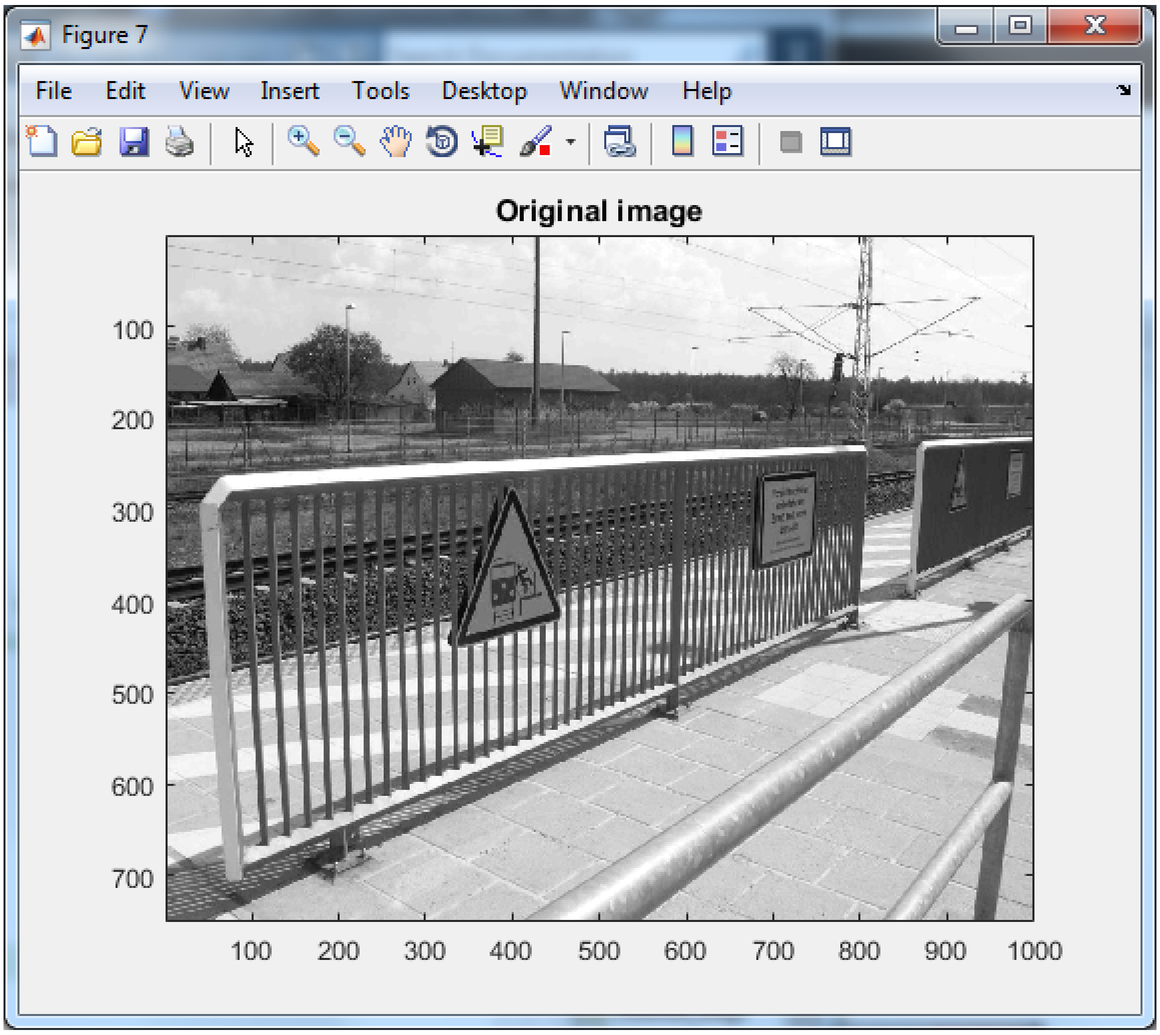


In real-life images, we don’t see any obvious repeating patterns. That is why the 2d fft has one very strong component in the middle (note that the scale is in dB), with other components being around 20-30 dB. We see the middle “radiating”, which means that still some repeating patterns in the image exist.

Task 2

2D filtering

Note: since the script was written at home, stand number 7 settings were chosen.
In this task we will apply filters to a real-life picture. The picture is the same as in the previous task. These are the obtained images:



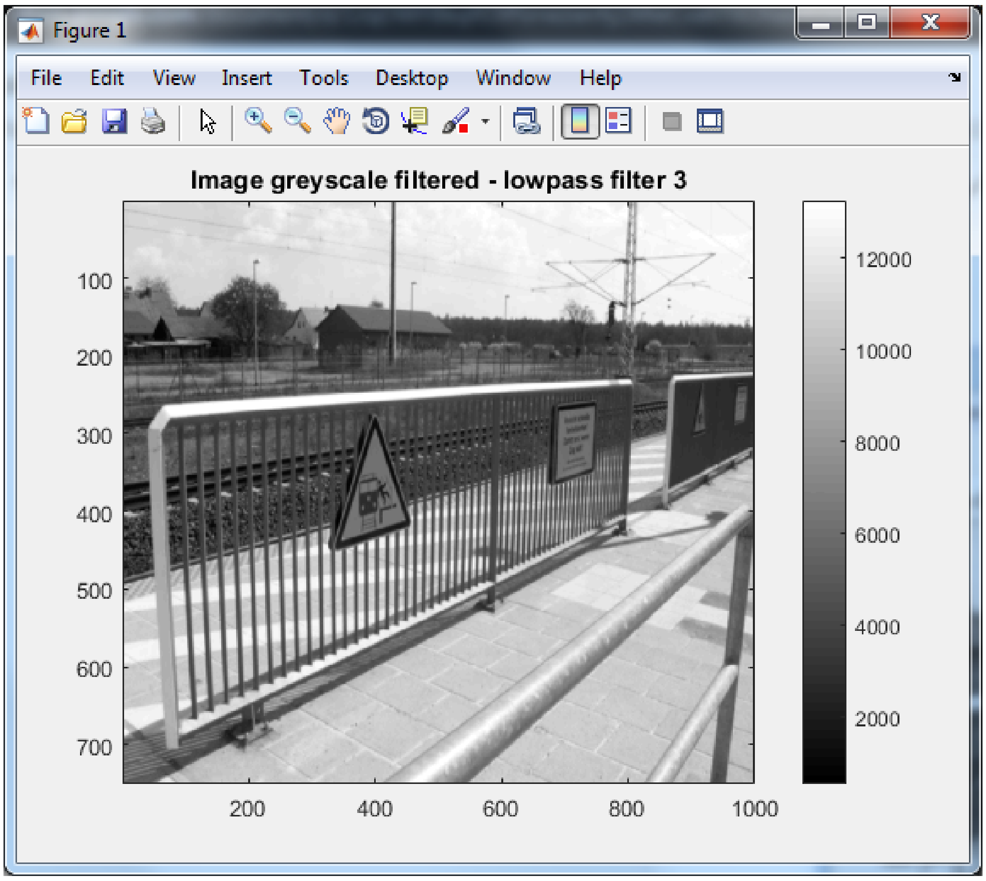


Figure 1
File Edit View Insert Tools Desktop Window Help
Image greyscale filtered - lowpass filter 3
100
200
300
400
500
600
700
200 400 600 800 1000
12000
10000
8000
6000
4000
2000

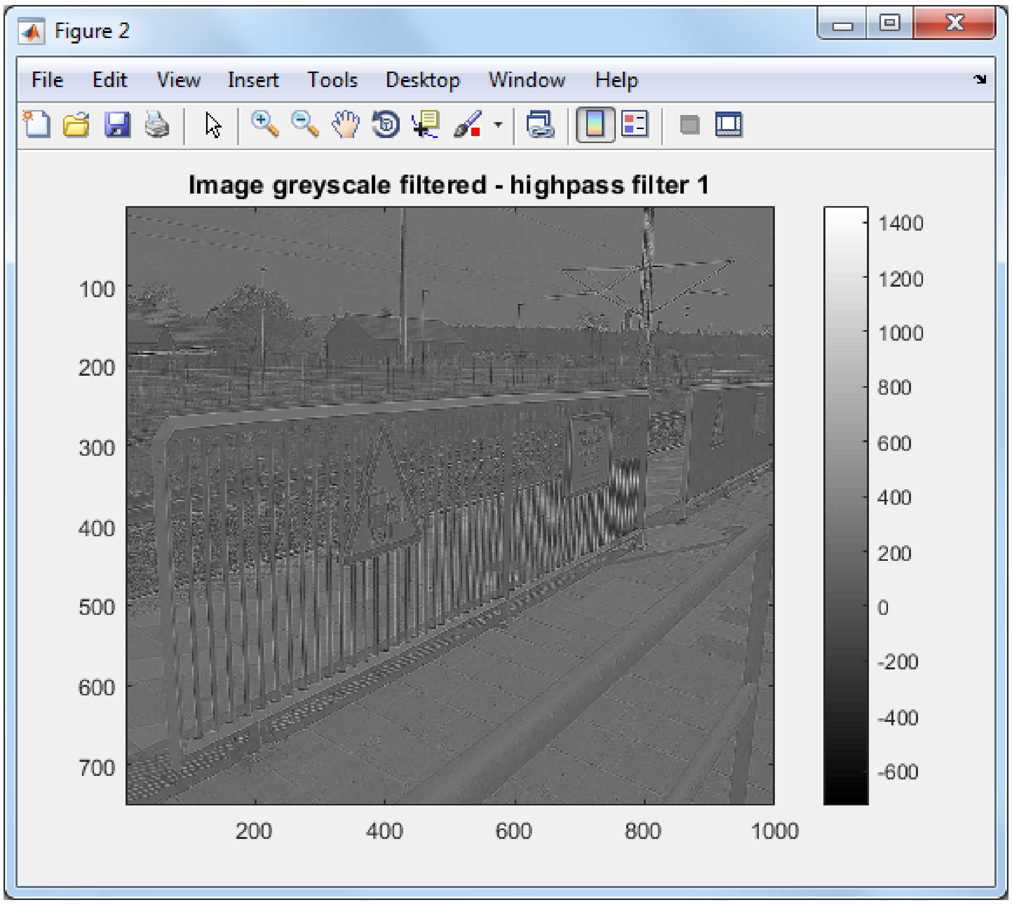
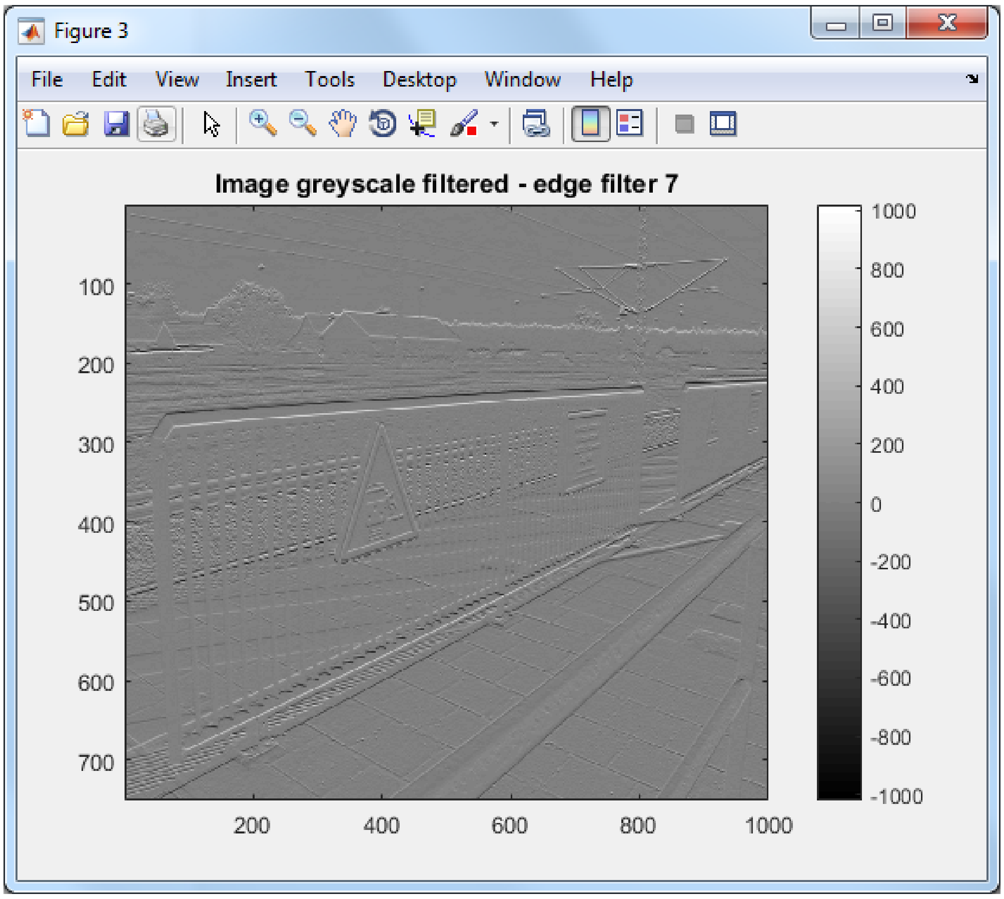
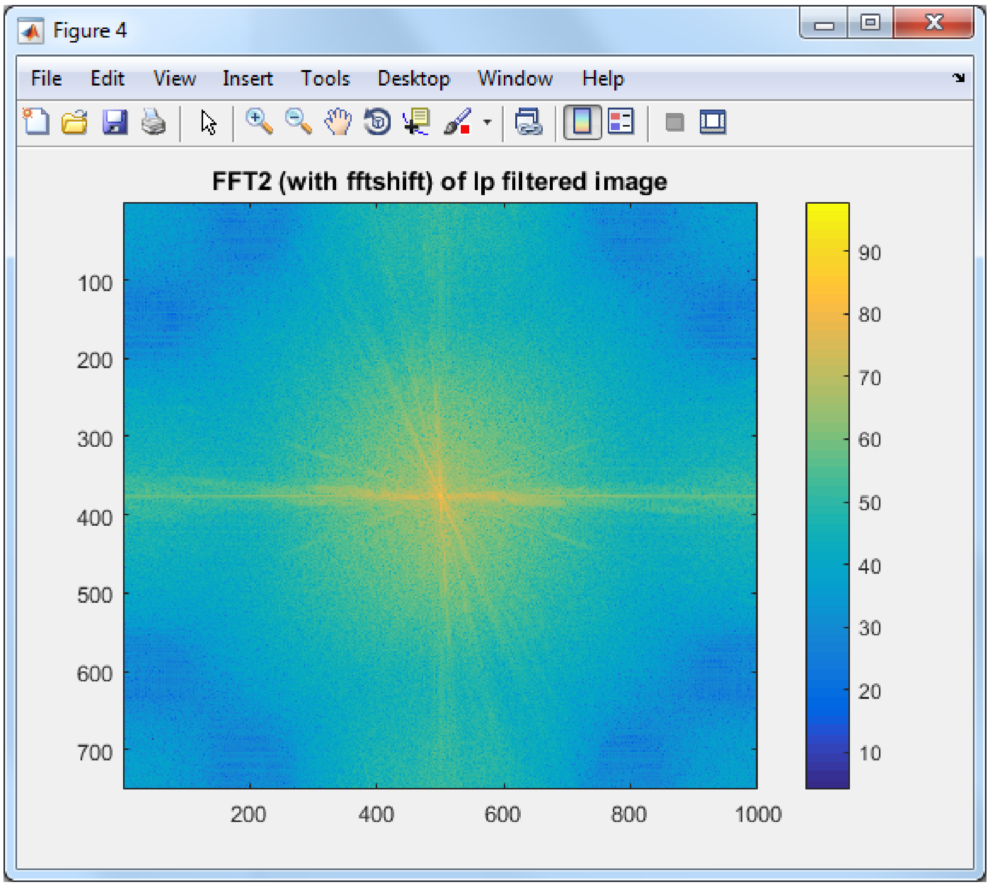


Figure 2
File Edit View Insert Tools Desktop Window Help
Image greyscale filtered - highpass filter 1
100
200
300
400
500
600
700
200 400 600 800 1000
1400
1200
1000
800
600
400
200
0
-200
-400
-600



The following 2d ffts were obtained:



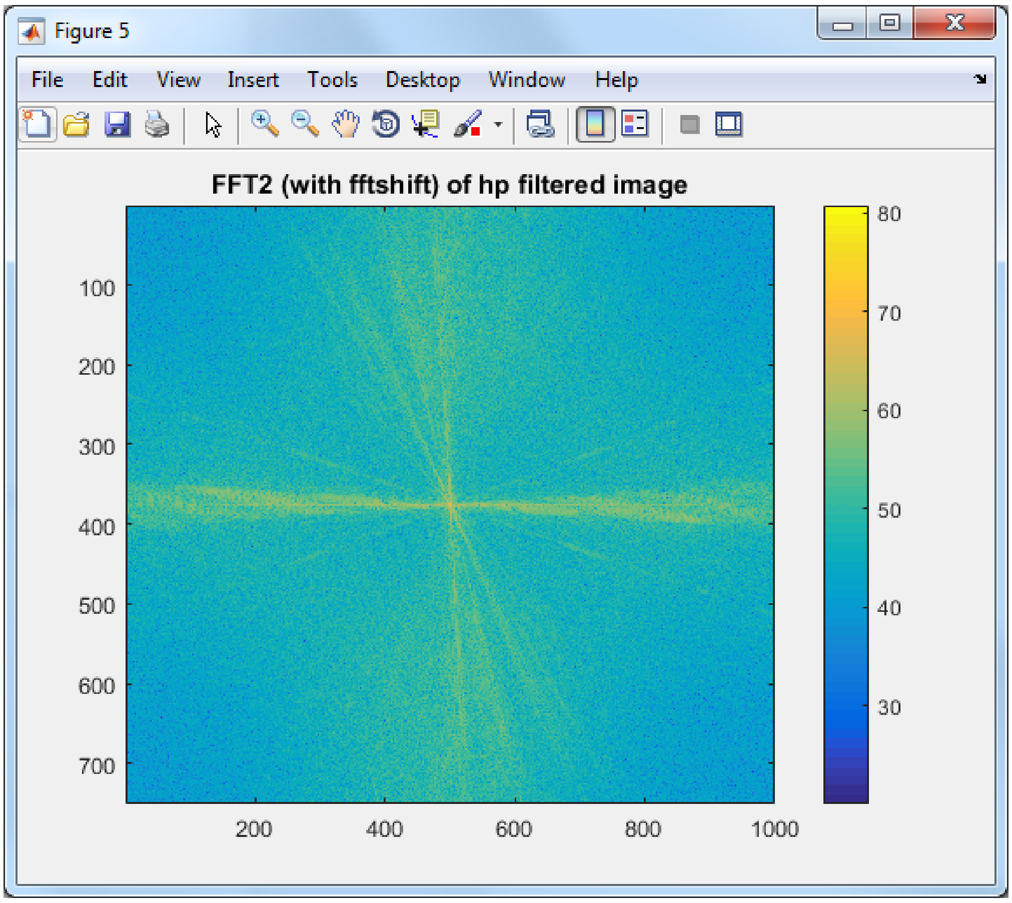


Figure 5
File Edit View Insert Tools Desktop Window Help
FFT2 (with fftshift) of hp filtered image
100
200
300
400
500
600
700
200
400
600
800
1000
80
70
60
50
40
30

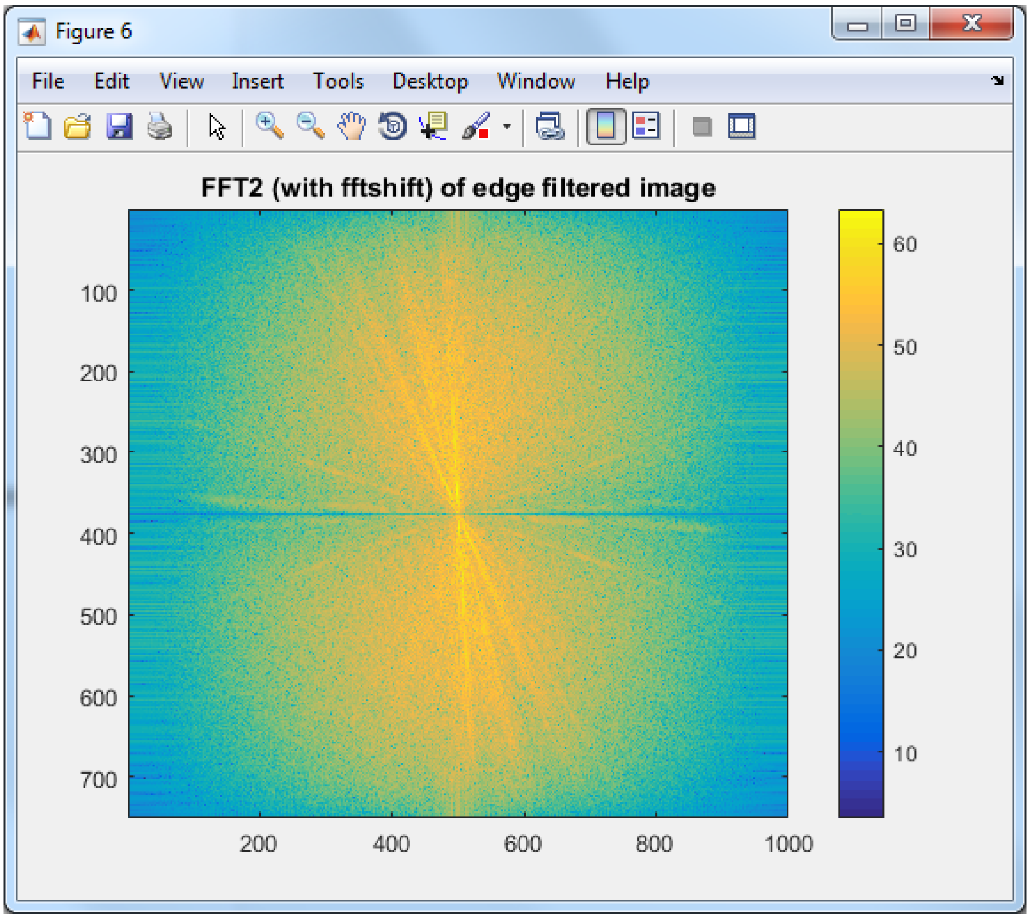


Figure 6
File Edit View Insert Tools Desktop Window Help
FFT2 (with fftshift) of edge filtered image
100
200
300
400
500
600
700
200
400
600
800
1000
60
50
40
30
20
10

Filters used in the task. lp4 was used to better see the effect on higher res images:

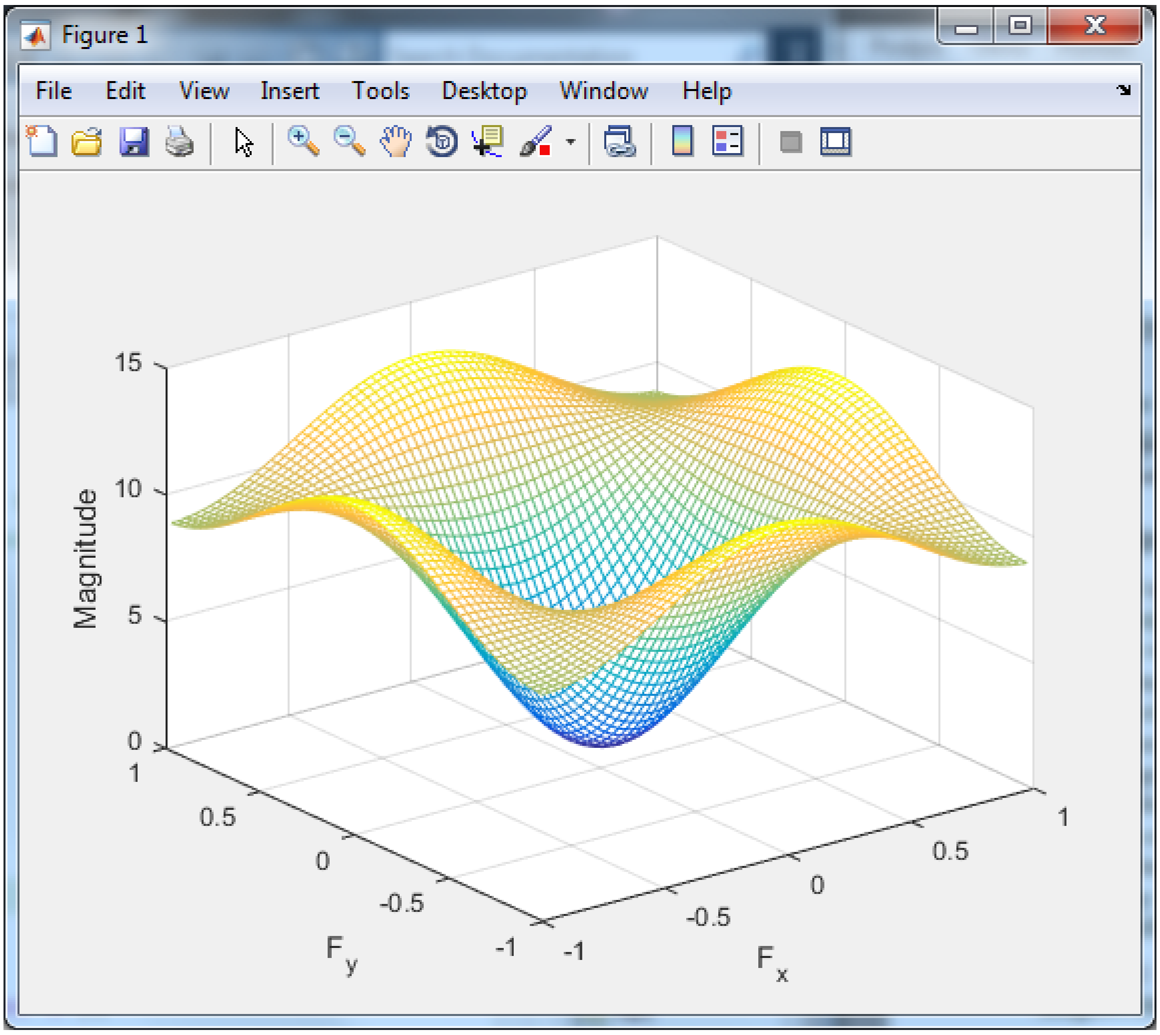
```
lp4=...
[1, 1, 2, 1, 1;...
1, 2, 4, 2, 1;...
2, 4, 8, 4, 2;...
1, 2, 4, 2, 1;...
1, 1, 2, 1, 1];

hp =

    -1    -1    -1
    -1     9    -1
    -1    -1    -1

edg =

     1     2     1
     0     0     0
    -1    -2    -1

Min and max for low-pass: 80 , 4068
Min and max for high-pass: -221 , 1088
Min and max for edge: -662 , 660
```

When we look at the low-pass filter image, we can conclude that this filter made the picture more "blurry". It is harder to see the edges of objects. On the fft we can see that the middle frequency component is kept, where the edge frequency components are cut off and have lower power. This filter should be completely avoided for edge detection. It makes edge detection harder than in the main image.

For high-pass the reverse is true. Middle frequency components (closer to 0) are cut off and have reduced magnitude, whereas the frequencies far away from the center are kept. This filter has some edge detection capabilities.

For the edge filter, we see that it has similar characteristics to the high-pass, with the difference that the main frequency components are of even higher magnitude than in the high-pass, but also the smaller magnitudes are kept even lower.

Using freqz2() I chose the following filter analysis:

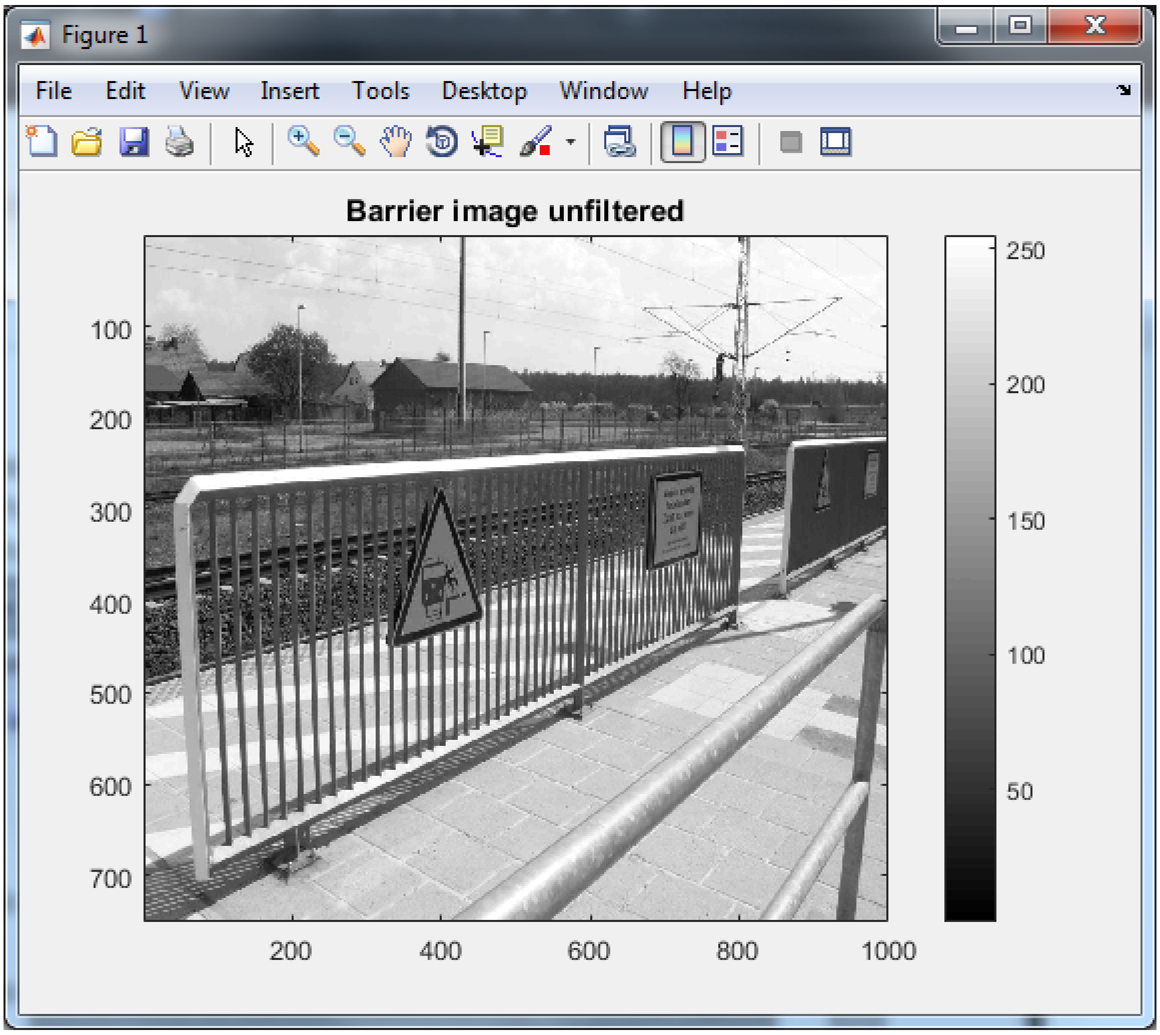


We can see that this is a high pass filter. Near the edges (high frequencies) we see an increase in magnitude, with a big decrease in the middle (lower frequencies).

Task 3

Edge detection

In this task I will apply different edge detection filters to choose the best filter suited for the chosen image, and then apply thresholding to display “an outline” of the image showing only edges. I have chosen edge filter number 4 to analyze. Unfiltered image is as follows:



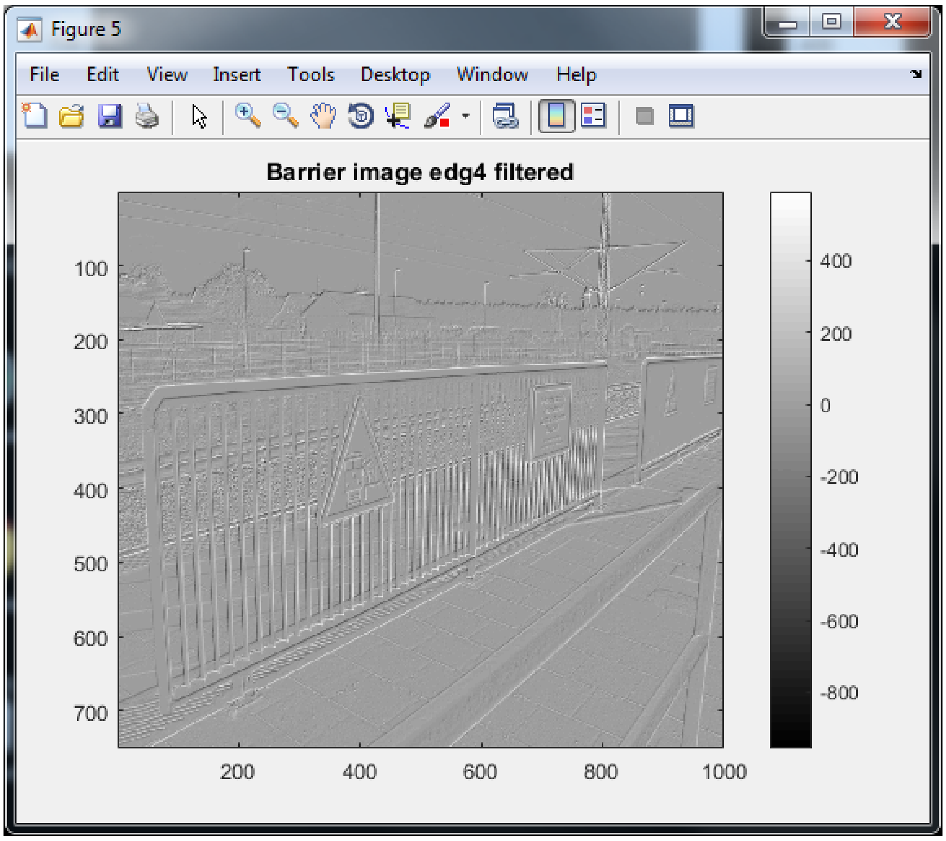


Figure 5
File Edit View Insert Tools Desktop Window Help
Barrier image edg4 filtered
100
200
300
400
500
600
700
200
400
600
800
1000
400
200
0
-200
-400
-600
-800

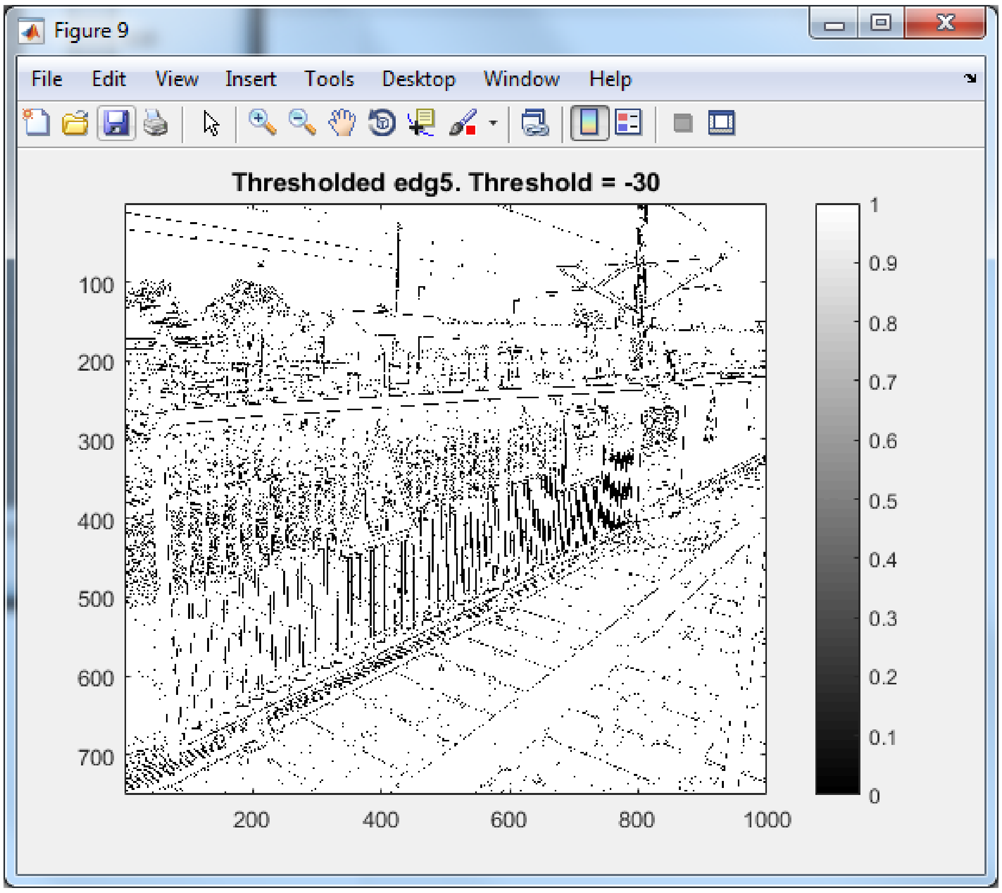


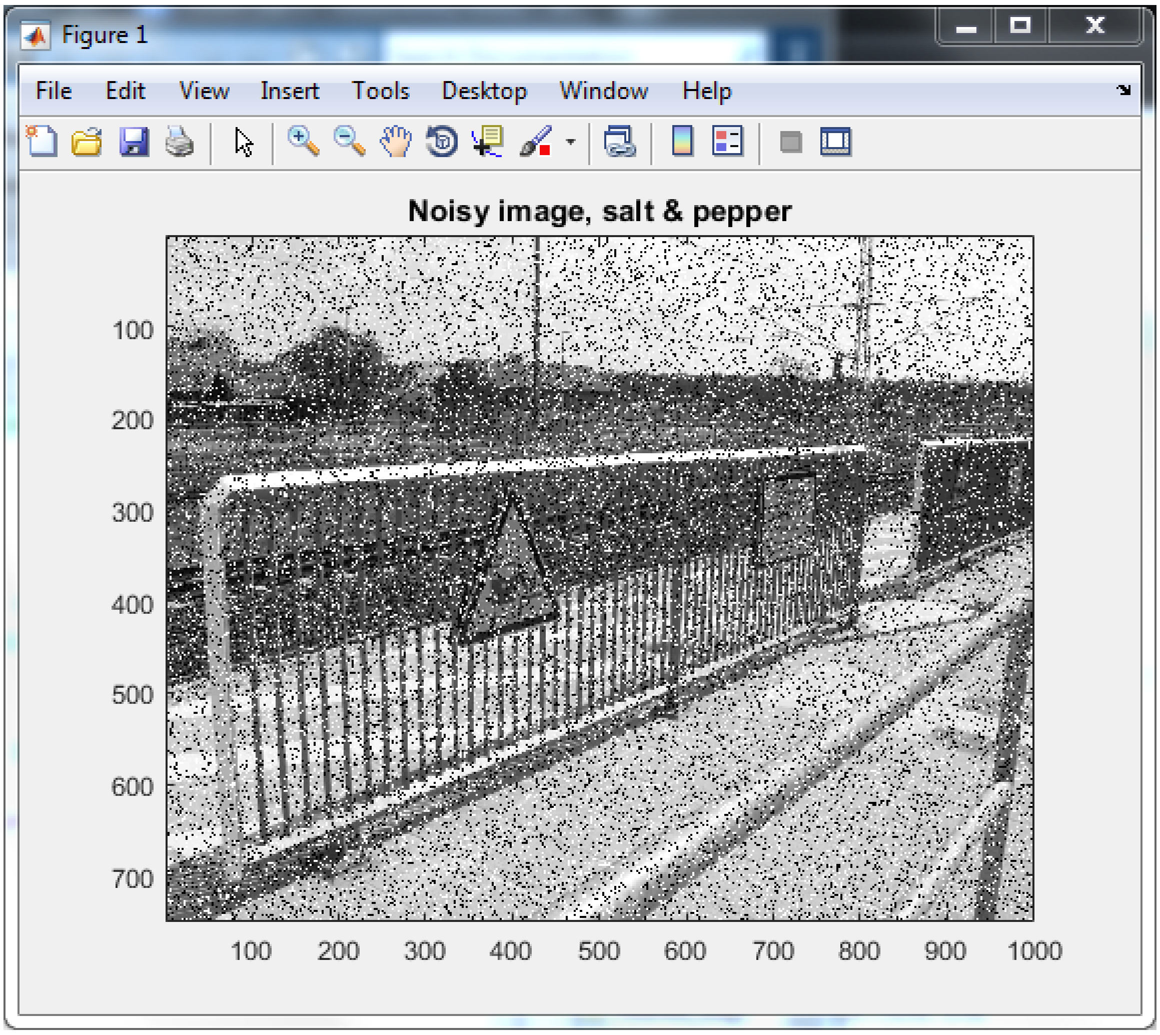
Figure 9
File Edit View Insert Tools Desktop Window Help
Thresholded edg5. Threshold = -30
100
200
300
400
500
600
700
200
400
600
800
1000
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0

Compared to other edge filters, in my opinion filter number 4 worked in the best way. We can obtain from it a good threshold image showing edges on the image.

Task 4

Noise removal

In this task I apply a salt and pepper noise to a signal and then check how the low-pass + median filter combo works when removing the noise. The obtained results are as follows:



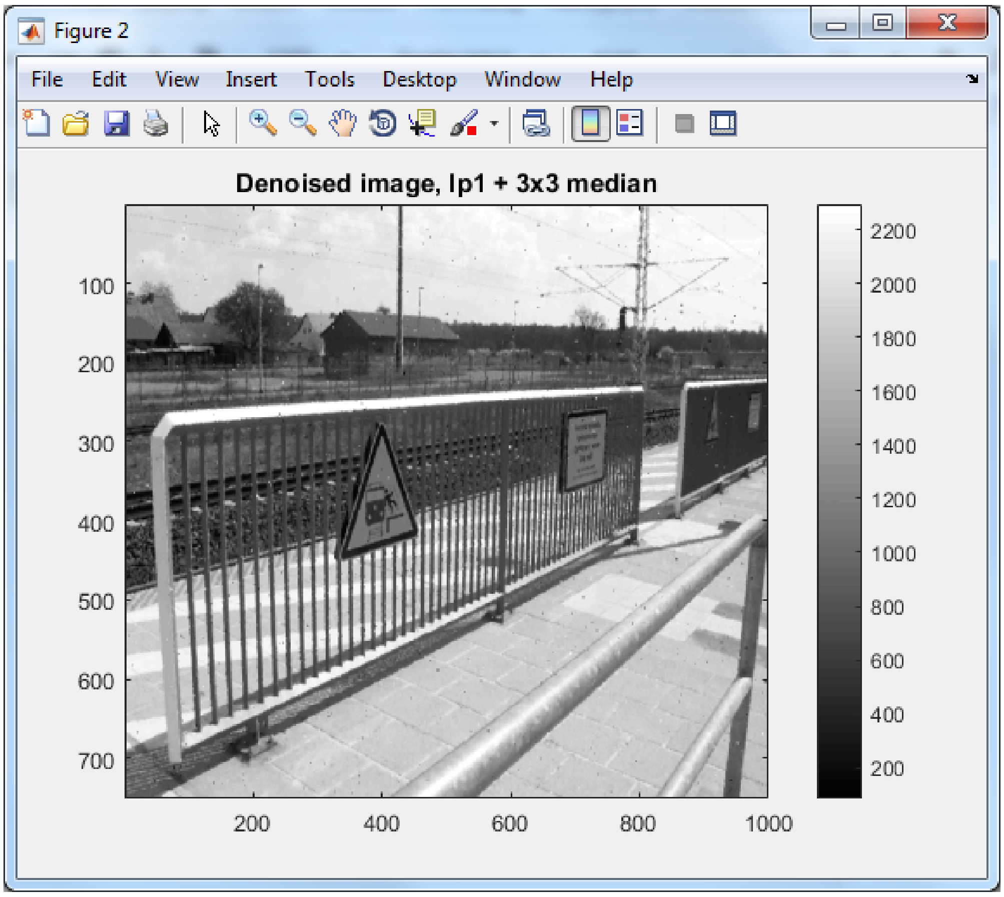


Figure 2
File Edit View Insert Tools Desktop Window Help
Denoised image, Ip1 + 3x3 median
100
200
300
400
500
600
700
200
400
600
800
1000
2200
2000
1800
1600
1400
1200
1000
800
600
400
200

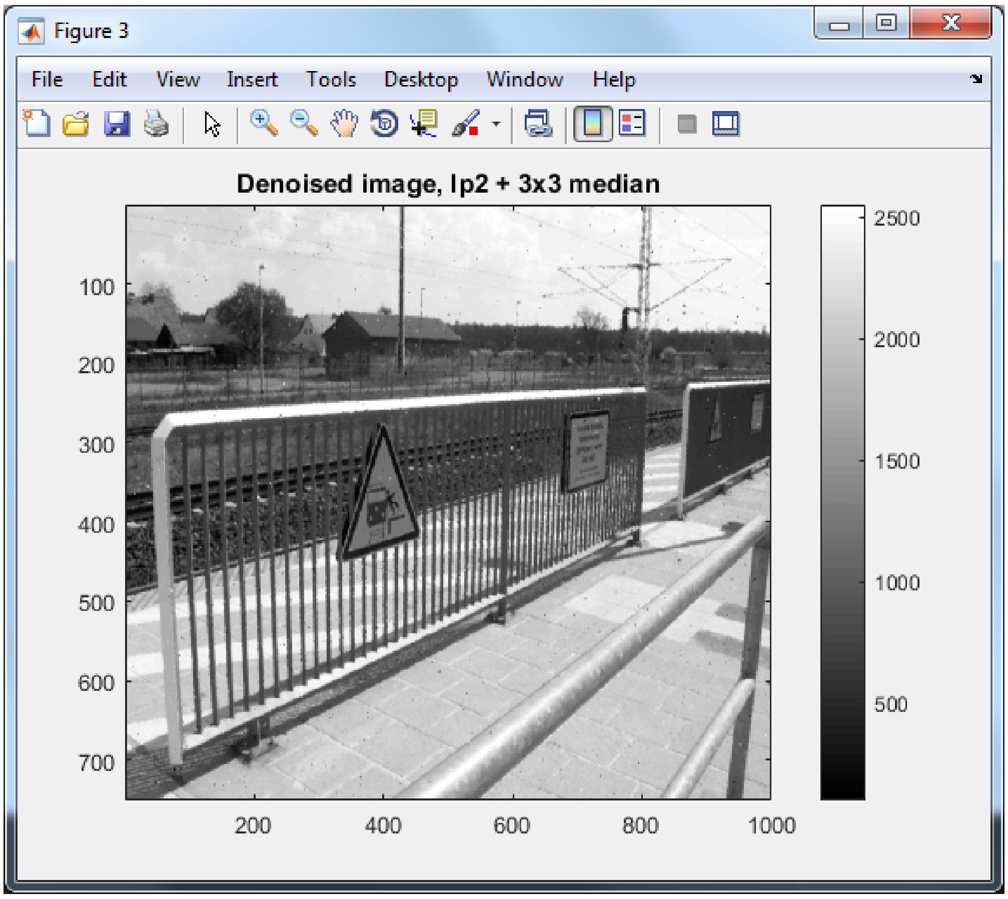


Figure 3
File Edit View Insert Tools Desktop Window Help
Denoised image, Ip2 + 3x3 median
100
200
300
400
500
600
700
200
400
600
800
1000
2500
2000
1500
1000
500

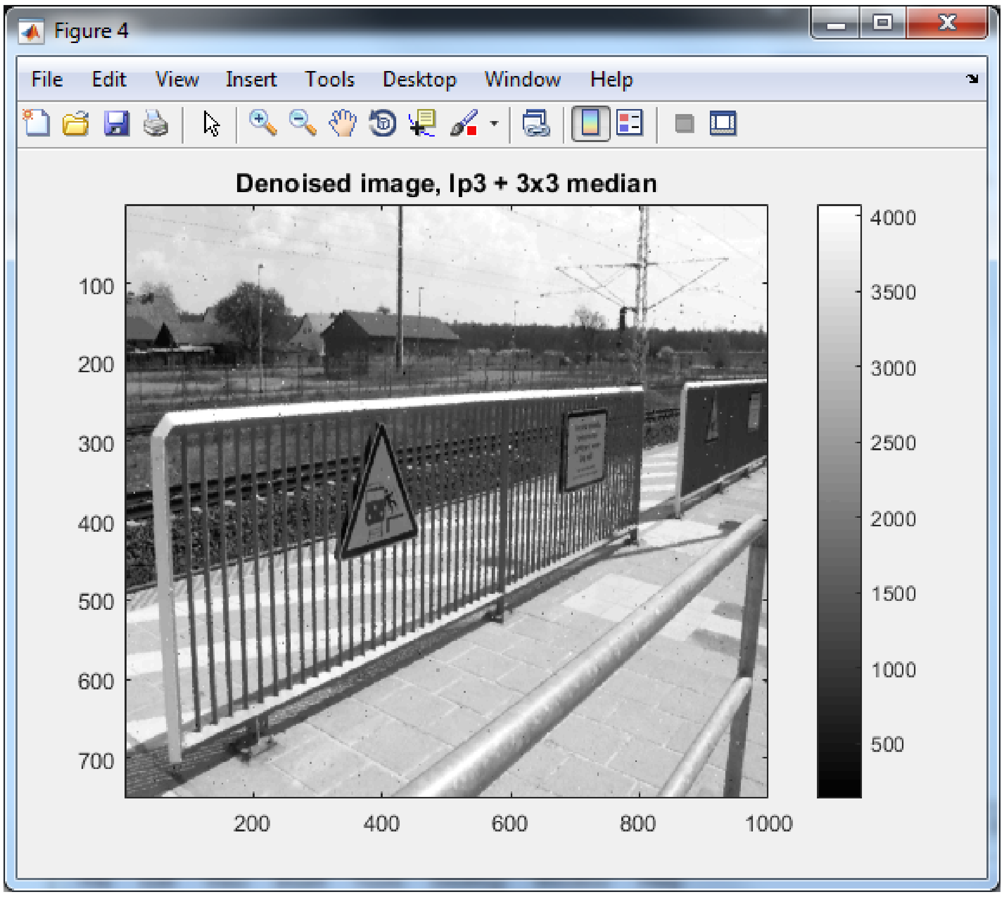
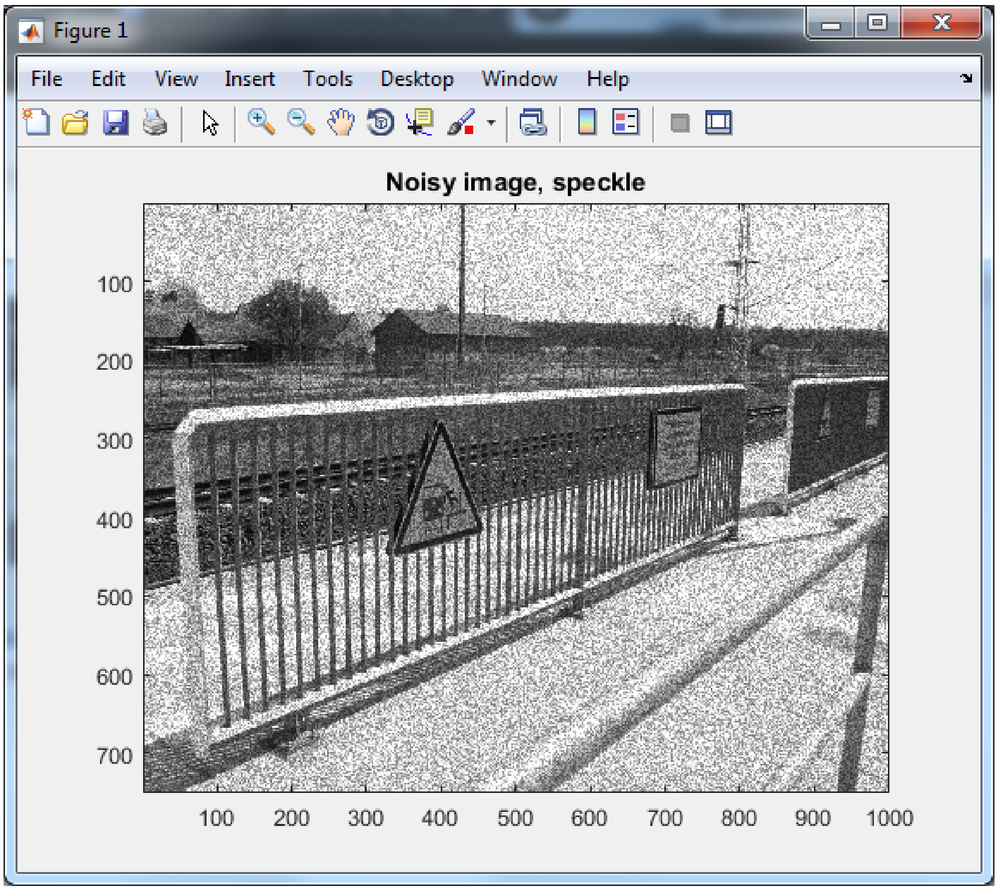


Figure 4
File Edit View Insert Tools Desktop Window Help
Denoised image, lp3 + 3x3 median
100
200
300
400
500
600
700
200
400
600
800
1000
4000
3500
3000
2500
2000
1500
1000
500

Speckle noise:



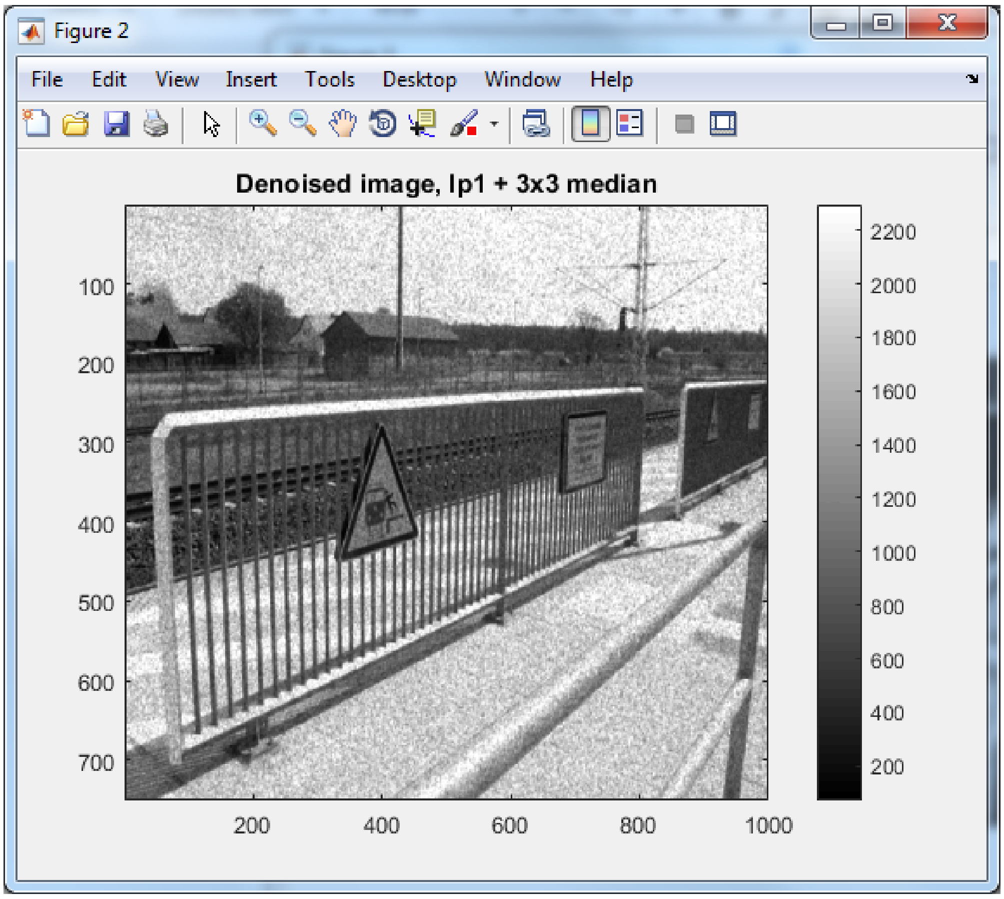


Figure 2
File Edit View Insert Tools Desktop Window Help
Denoised image, Ip1 + 3x3 median
100
200
300
400
500
600
700
200
400
600
800
1000
2200
2000
1800
1600
1400
1200
1000
800
600
400
200

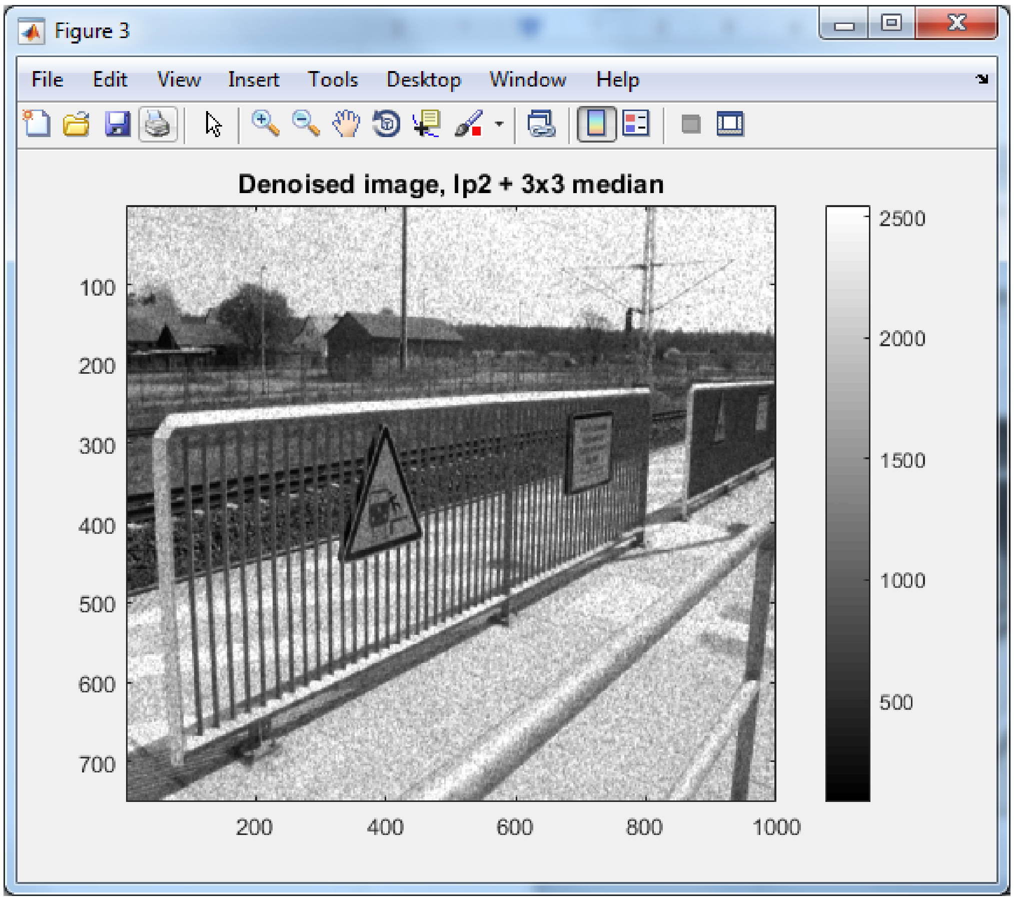
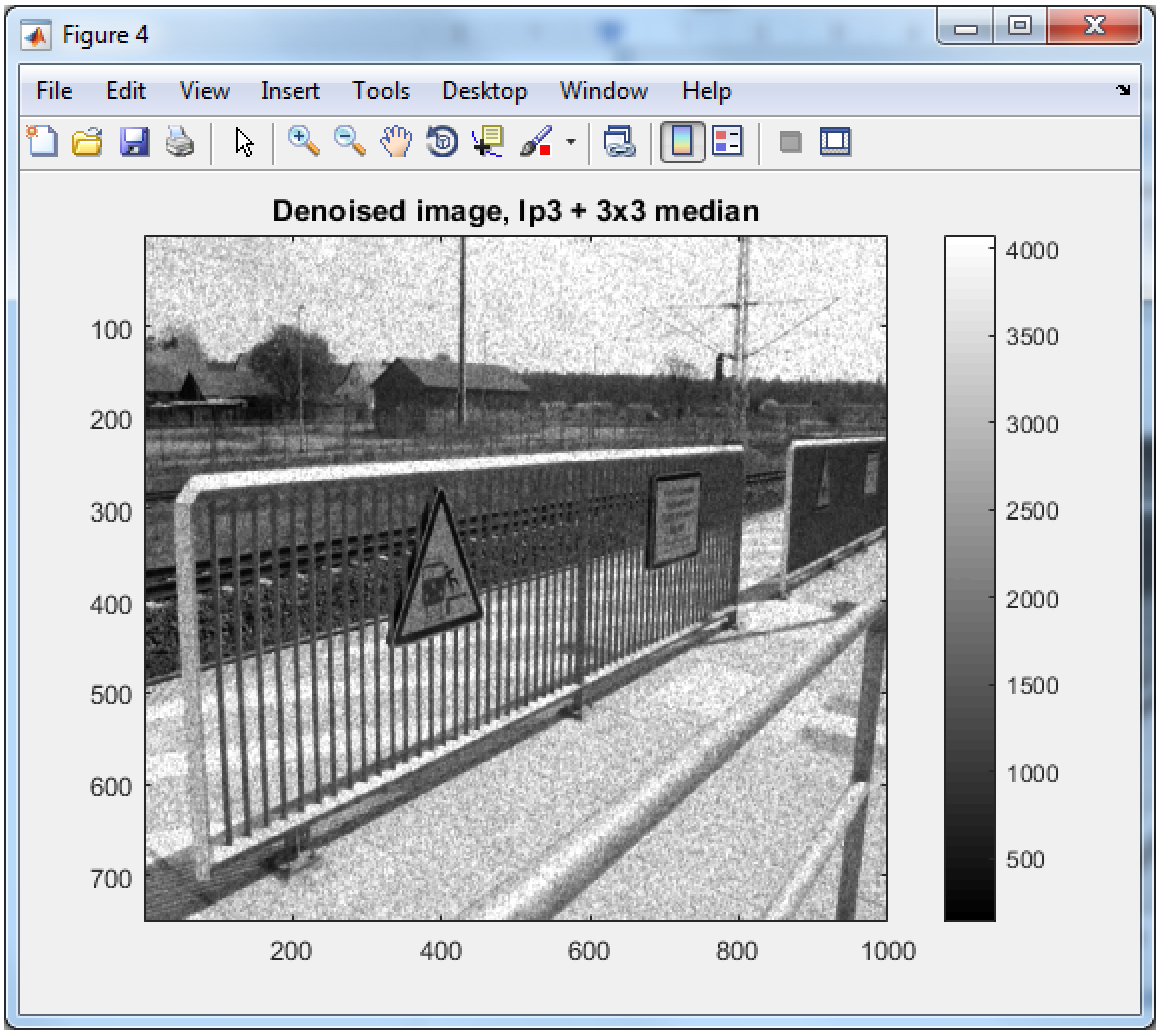


Figure 3
File Edit View Insert Tools Desktop Window Help
Denoised image, Ip2 + 3x3 median
100
200
300
400
500
600
700
200
400
600
800
1000
2500
2000
1500
1000
500



Filters perform similarly when removing the noise. However, the salt and pepper noise removal is better compared to speckles. The filters perform worse with speckle noise.