LAB 2

IMAGE FILTERING AND FOURIER TRANSFORM

Marco Demutti

Matteo Dicenzi

Introduction **3**

Procedure **3**

Results **6**

Conclusions **11**

1. Introduction

The aim of this experiment is to examine the effect of the **noise** on a natural image, to show the process of **image filtering**, and finally to show some examples of **Fourier transform (FFT)** of an image. Each image is often displayed together with its **histogram**, that is a graph that helps evaluate some useful properties, by showing the number of pixels in an image at each different intensity value.
Noise occurs as an inevitable result of the image formation process, but in this work we are going to add it artificially to show the different effects and how it is possible to intervene to remove it, depending on the different type of noise (in this particular case, gaussian or salt & pepper).
In this respect, we are going to examine the **image filtering** technique, based on convolution process, that allows to replace each pixel with a linear combination of its neighbors. In particular, we are going to deal with **smoothing filters**, that produce blurring for noise reduction. As for linear filters, *moving average filter* and *low-pass gaussian filter* are shown, while *median filter* is used as an example of a non-linear case. Other simple types of filters are shown too, such as a *shifting filter*, and a *sharpening filter*, that is able to accentuate differences with local average.
As a final step, the Fourier transform is used on natural images, as well as on two filters. The result of this operation is always a transformed image, with complex values (real component - magnitude, complex component - phase). In particular, their magnitude will be displayed to show some of their properties.

2. Procedure

All the experiments are carried out on two natural images in parallel.
First, after displaying each original image together with its histogram, two different types of noises are added: a *gaussian noise*, with *standard deviation* $\sigma = 20$ and a *salt & pepper noise*, with *density = 20%*.

Then, in the second part, three different smoothing filters are used to remove the two types of noise: a *moving average*, a *low-pass Gaussian filter*, and a *median filter*, each of them created with two different spatial supports: 3x3 and 7x7.
The first two, which are linear filters, were created by using the MATLAB function *fspecial*, displayed both as images and as surfaces, then applied to all images by using *imfilter* function.
As for Gaussian filters, the standard deviation was chosen following the empirical rule $halfwidth \approx 3\sigma$, in order to avoid *aliasing*.

Figure 2.1 shows the 3x3 low-pass Gaussian spatial filter (the others are omitted, but a 7x7 moving average filter is later shown in *Figure 2.3* as an example of *blurring filter*).

0.0113	0.0838	0.0113
0.0838	0.6193	0.0838
0.0113	0.0838	0.0113

Figure 2.1: 3x3 low-pass Gaussian filter

The third filter was directly applied to the images by using *medfilt2* function, since, it being non-linear, it is not possible to define an *impulse response*.

In the third part of the experiment, four other types of 7x7 linear filter were created and applied to the images: a *neutral filter,* which should not modify the images at all; a *shifting filter*, that returns a shifted version of the original image; a *blurring filter*, that is a moving average used to blur the images; finally a *sharpening filter*, that returns a sharpened version of the image, with accentuated differences.
In order to perform the required transformations, 4 different functions were written, one for each filter: ***neutralFilter, shiftFilter, blurringFilter, sharpeningFilter.*** All of them take as input an image, construct a filter (which is different for every function) with a spatial support of 7x7 pixels and then they return the filtered image, which is obtained through the Matlab function *imfilter,* the one that actually perform the filtering.
Figures 2.2-2.5 show all the four different spatial filters (i.e. *convolution kernels*) used for our purpose.

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

Figure 2.2: filter of function neutralFilter

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	1
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

Figure 2.3: filter of function shiftFilter

0.0204	0.0204	0.0204	0.0204	0.0204	0.0204	0.0204
0.0204	0.0204	0.0204	0.0204	0.0204	0.0204	0.0204
0.0204	0.0204	0.0204	0.0204	0.0204	0.0204	0.0204
0.0204	0.0204	0.0204	0.0204	0.0204	0.0204	0.0204
0.0204	0.0204	0.0204	0.0204	0.0204	0.0204	0.0204
0.0204	0.0204	0.0204	0.0204	0.0204	0.0204	0.0204
0.0204	0.0204	0.0204	0.0204	0.0204	0.0204	0.0204

Figure 2.4: filter of function blurringFilter

-0.0204	-0.0204	-0.0204	-0.0204	-0.0204	-0.0204	-0.0204
-0.0204	-0.0204	-0.0204	-0.0204	-0.0204	-0.0204	-0.0204
-0.0204	-0.0204	-0.0204	-0.0204	-0.0204	-0.0204	-0.0204
-0.0204	-0.0204	-0.0204	1.9796	-0.0204	-0.0204	-0.0204
-0.0204	-0.0204	-0.0204	-0.0204	-0.0204	-0.0204	-0.0204
-0.0204	-0.0204	-0.0204	-0.0204	-0.0204	-0.0204	-0.0204
-0.0204	-0.0204	-0.0204	-0.0204	-0.0204	-0.0204	-0.0204

Figure 2.5: filter of function sharpeningFilter

The final part of the laboratory was about the Fourier Transform (FFT).

First, 2 MATLAB functions are used to compute the FFT of the images:
Fft2 take as input an image and returns its two-dimensional Fourier transform.
fftshift takes as input the two-dimensional Fourier transform previously computed and shifts the zero-frequency component to the center of the spectrum.
The logarithm of the magnitude of the transformed image is then displayed, both as an image (function *imagesc*) and as a surface (function *mesh*).

Afterwards, the same functions are used again to produce and display the magnitude of a Gaussian filter. *fspecial* is again the Matlab function used to generate the filter, which now has a spatial support of 101x101 pixels and a standard deviation $\sigma = 5$.

Finally, we display the FFT of a sharpening filter.
In order to do so, ***sharpeningFilter*** function is used to create again a sharpening filter with spatial support of 7x7. Then, by using *padarray* function, the filter is placed at the center of a 101x101 zeros image.

3. Results

The two original images, along with their histograms, are shown in *Figure 3.1*.

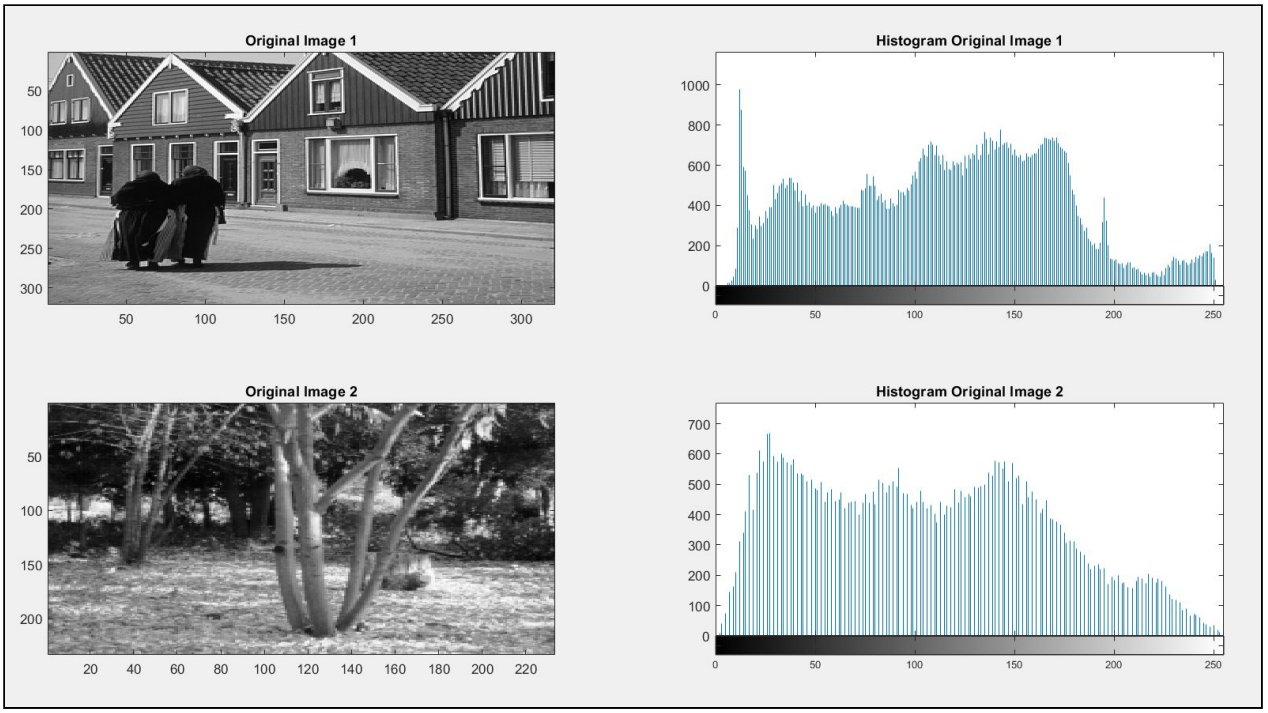

Figure 3.1 - Original images and corresponding histograms

Figures 3.2-3.7 show image 1 once the two types of noise were added, and then removed using the different filters (only the most relevant figures are shown). The same procedure was carried out on image 2, but images are not provided.

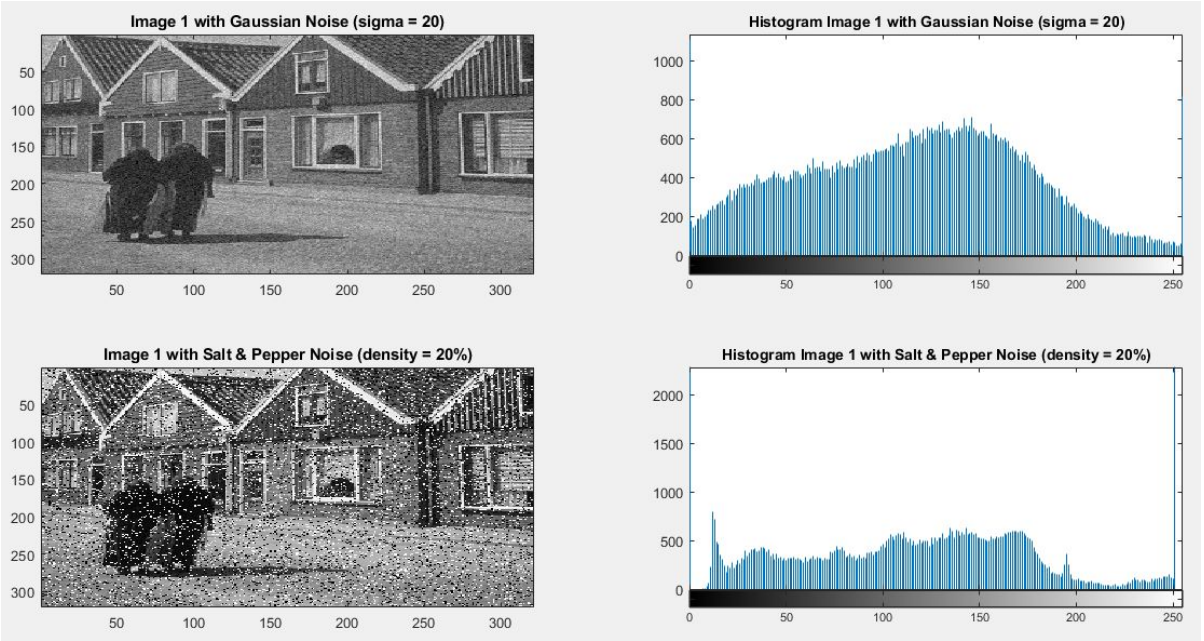

Figure 3.2 - image one with Gaussian Noise and with Salt & Pepper Noise

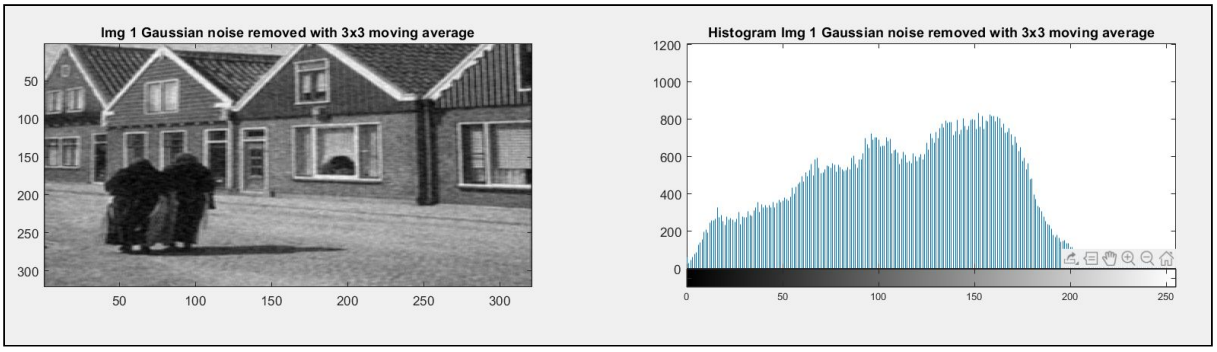

Figure 3.3 - image one with Gaussian Noise removed with moving average (3x3)

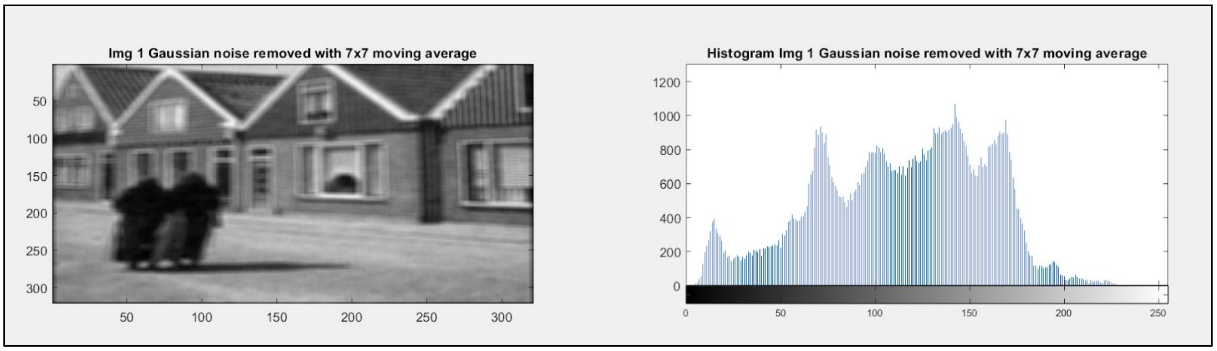

Figure 3.4 - image one with Gaussian Noise removed with moving average (7x7)

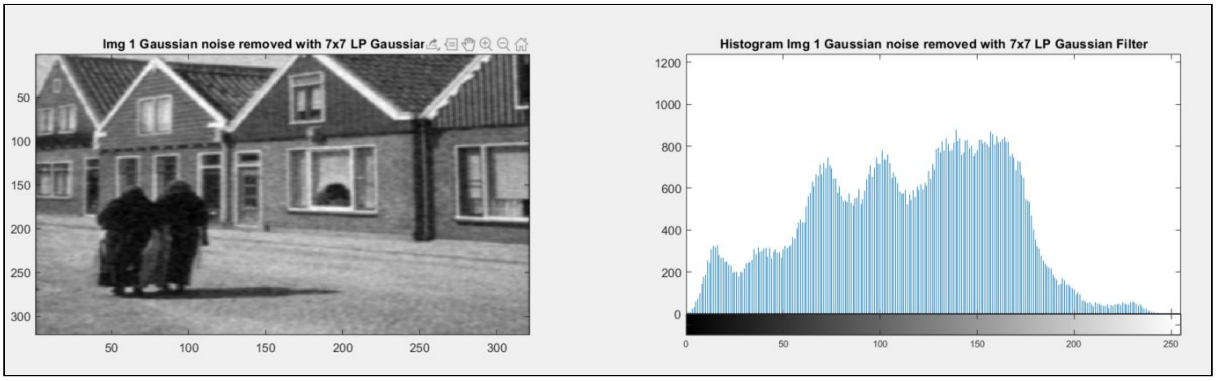

Figure 3.5 - image one with Gaussian Noise removed with LP Gaussian Filter (7x7)

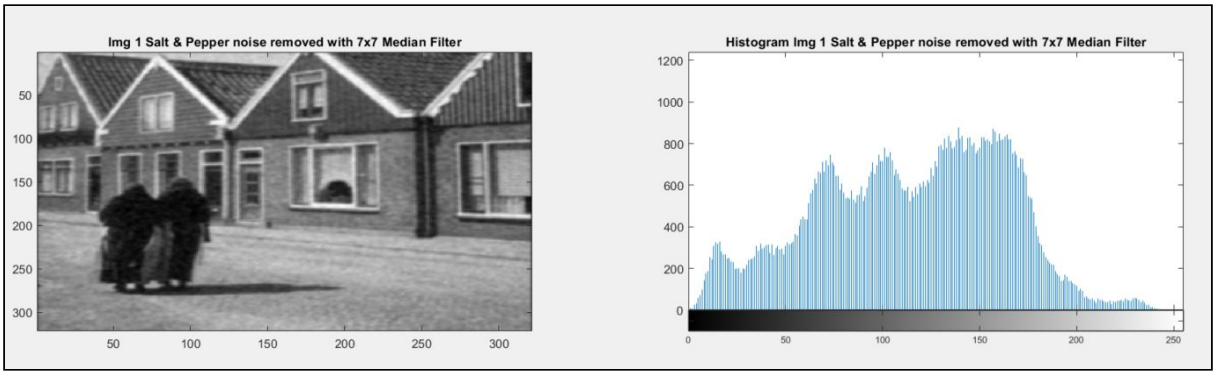

Figure 3.6 - image one with Salt & Pepper Noise removed with Median Filter (7x7)

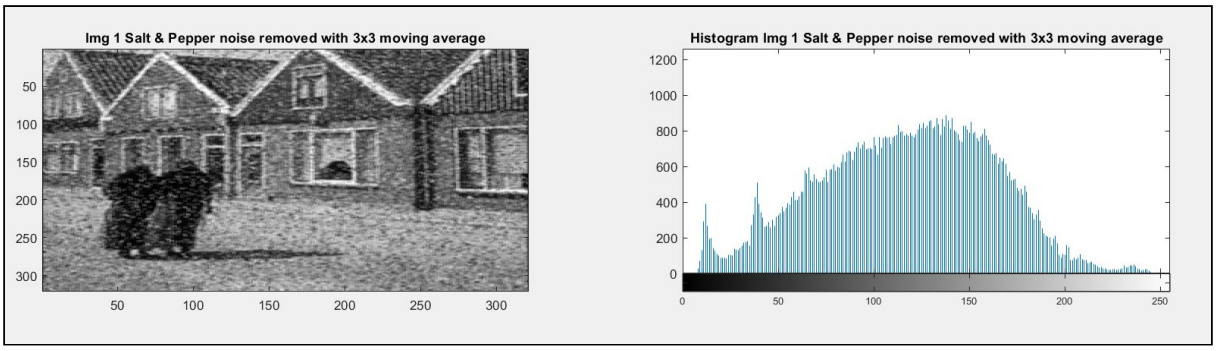

Figure 3.7 - image one with Salt & Pepper noise removed with moving average (7x7)

The results of second part of the experiment are shown in *figures 3.8-3.11*: the image is filtered using a neutral filter, a shifting filter, a blurring filter and a sharpening filter. Also in this case, the same procedure is carried out on both images, but this time only image 2 is shown.

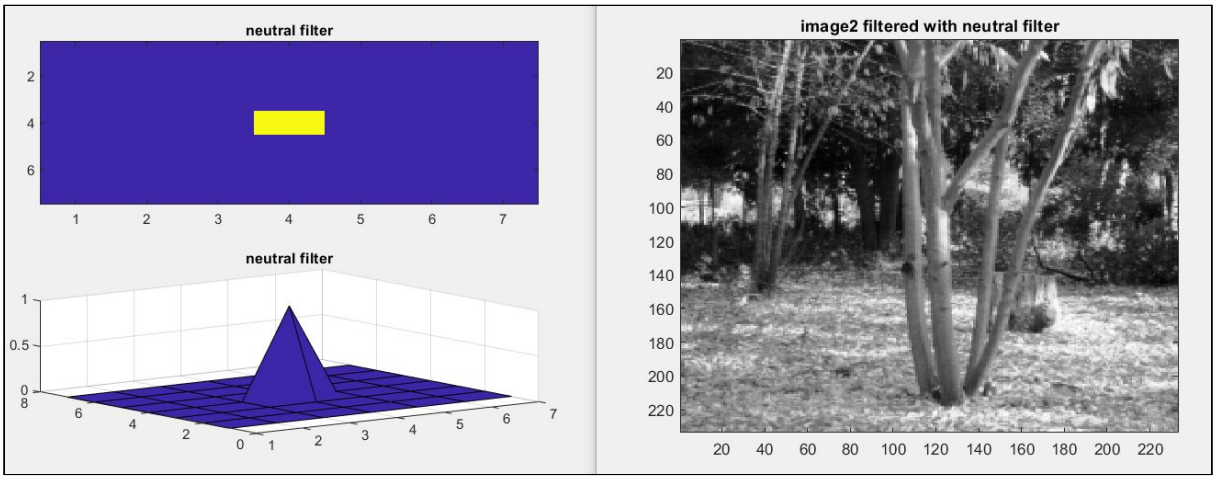

Figure 3.8 - image two filtered with neutral filter

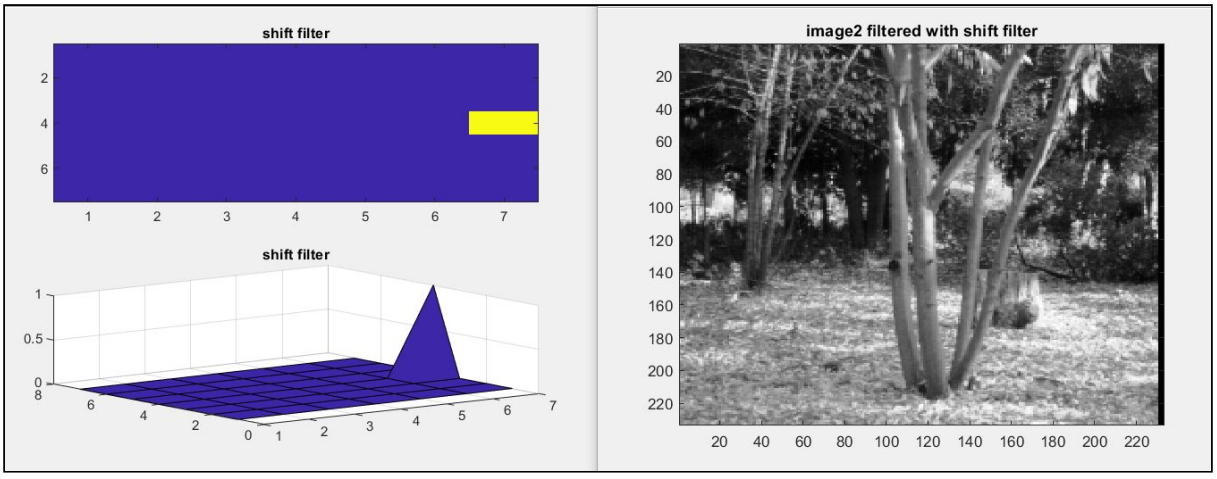

Figure 3.9 - image two filtered with shift filter

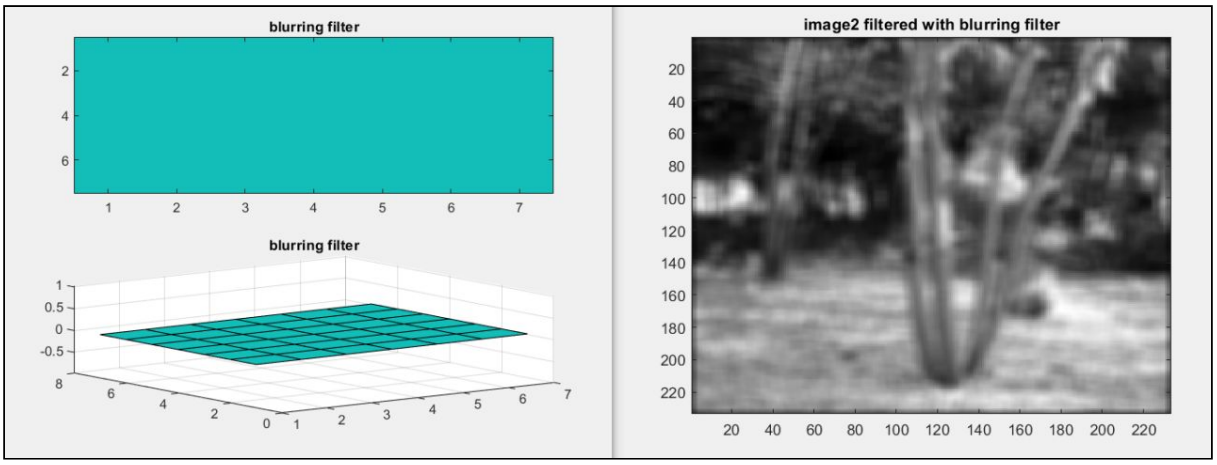

Figure 3.10 - image two filtered with blurring filter

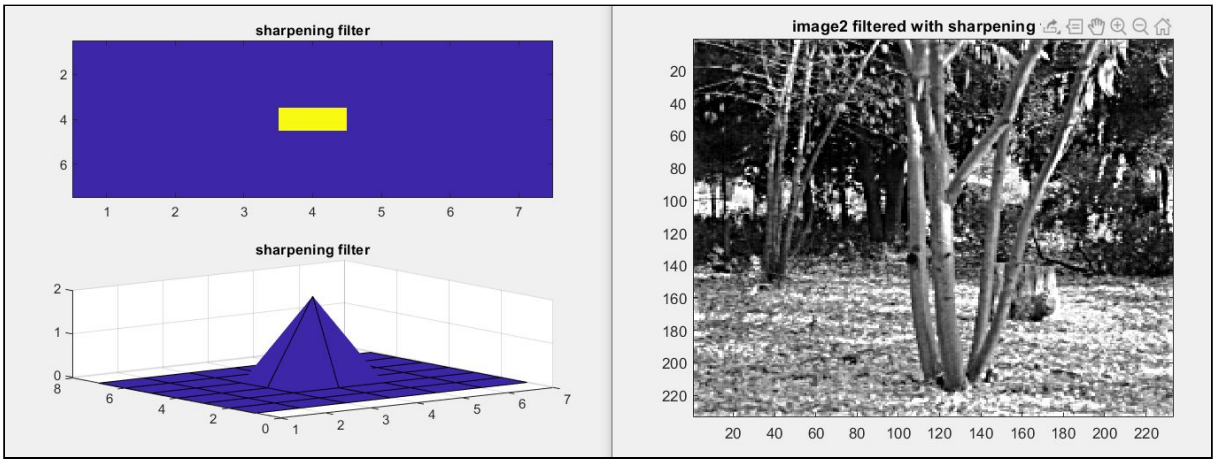

Figure 3.11 - image two filtered with sharpening filter

Finally, *figures 3.12-3.15* show the results related to the final part of the experiment.

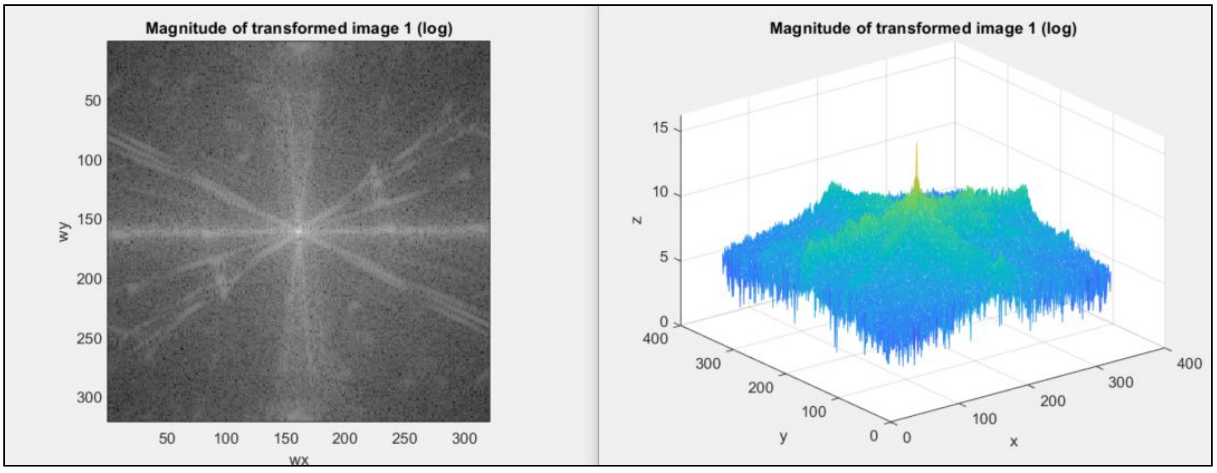

Figure 3.12 - magnitude of transformed image 1

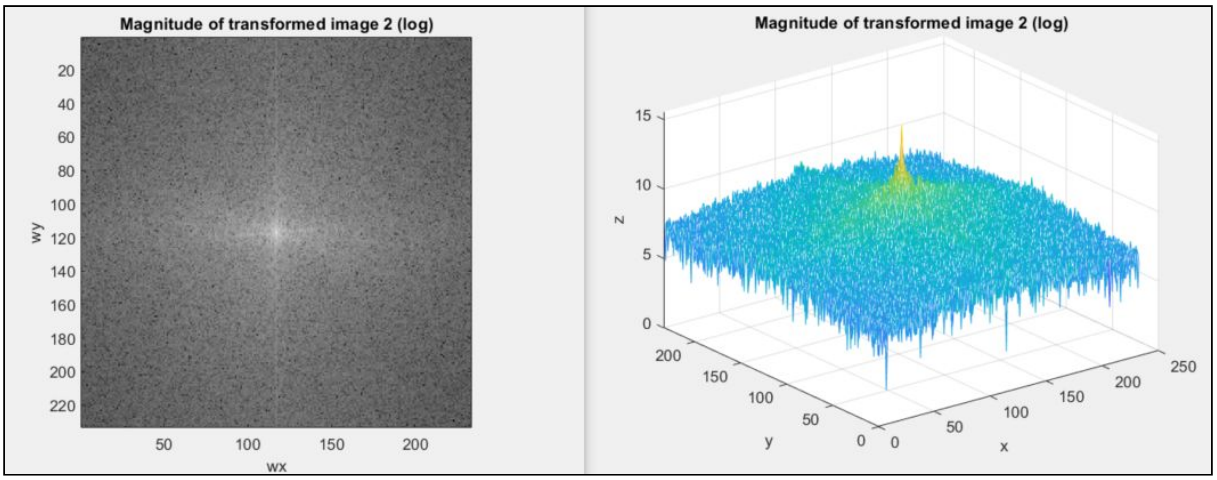

Figure 3.13 - magnitude of transformed image 2

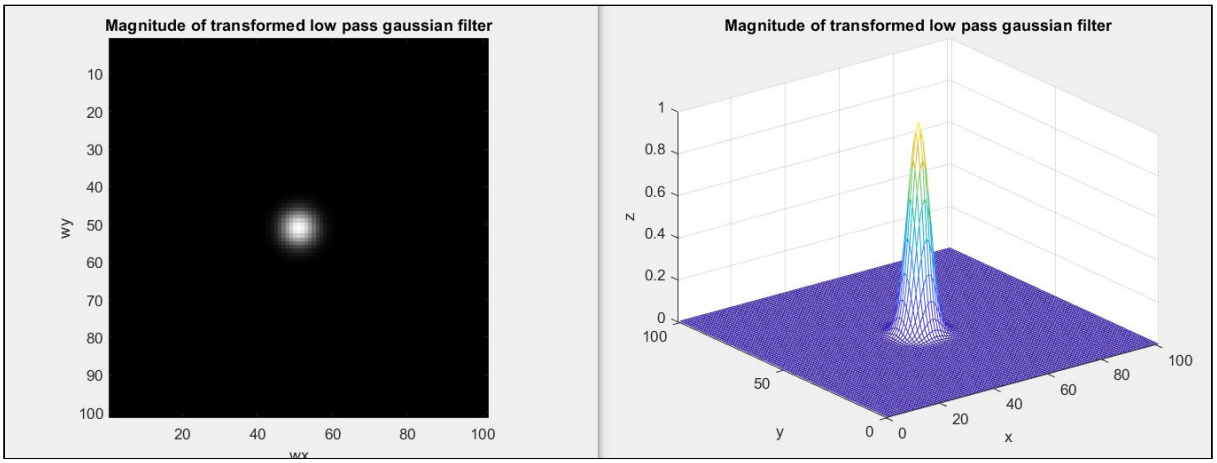

Figure 3.14 - magnitude of transformed low pass Gaussian filter

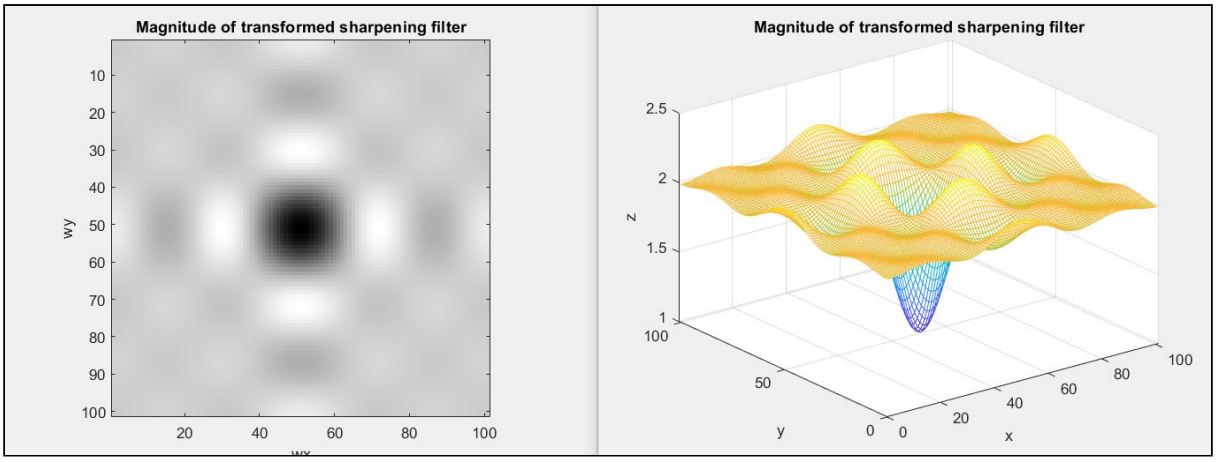

Figure 3.15 - magnitude of transformed sharpening filter

4. Conclusions

The results obtained in the first part of the laboratory show that, when an image is filtered in order to remove noise, not all the filters have the same effect; this depends not only on the filter *type* (i.e. Gaussian, blurring filter, Median filter) but also on filter characteristics like *standard deviation,* in case of a Gaussian filter, or like the *spatial support* in a generic case.
In particular in *figures 3.3-3.4* we can see how the spatial support of the moving average, that represents the size of the filter, determines the amount of the smoothing effect on the image: a bigger size allows to filter more noise, but also has the side effect of blurring the image.
So, the choice of the filter must be adapted to the particular situation and needs.
Also, in *figure 7*, it is evident that using a moving average filter, which performs the average of pixels in a neighborhood, is not a good choice for the nature of the Salt & Pepper noise, while a Median filter is a better choice. This can also be seen comparing the resulting image histograms with the original one.
In the final part, the results obtained performing the magnitude of the two-dimensional Fourier transform are coherent with the fact that all natural images have a similar magnitude transform (we know that the information about the energy at the different positions of the image is not in the magnitude, but in the phase).
Also the results of the magnitude of the two-dimensional Fourier transform of the two filters are as expected: they clearly emphasise and help visualise their properties and behaviours. In particular, *figure 3.14* clearly shows that Gaussian filter is a low-pass filter, since it retains the low frequency information while reducing the high frequency information.