Università
della
Svizzera
italiana

Faculty
of Informatics

Image and Video Processing **2023**

Students: Filippo Casari, Alessandro De Grandi

Report for Assignment 3

1. Theoretical exercise

Let us apply the Fourier transform to the first member :

$$\mathscr{F}(f(x,y)) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\; e^{-jux}\; e^{-jvy}\; dx\; dy$$

Now since by definition:

$$f(x,y) = f(x)\; f(y)$$

if we define γ as :

$$\gamma = \int_{-\infty}^{\infty} f(y)\; e^{-jvy}\; dy$$

which is a constant respect to $f(x)$, we get:

$$\mathscr{F}(f(x,y)) = \int_{-\infty}^{\infty} \gamma\; f(x)\; e^{-jux}\; dx$$

For the Constant Multiple Property of Integrals:

$$\mathscr{F}(f(x,y)) = \int_{-\infty}^{\infty} \gamma\; f(x)\; e^{-jux}\; dx = \gamma\; \int_{-\infty}^{\infty} f(x)\; e^{-jux}\; dx$$

By substituting γ with its definition:

$$\mathscr{F}(f(x,y)) = \int_{-\infty}^{\infty} f(y)\; e^{-jvy}\; dy\; \int_{-\infty}^{\infty} \gamma\; f(x)\; e^{-jux}\; dx$$

This is equivalent to:

$$F(u,v) = \mathscr{F}(f(x))\; \mathscr{F}(f(y)) = G(u)G(v)$$

2. Theoretical exercise

2.1. Point A

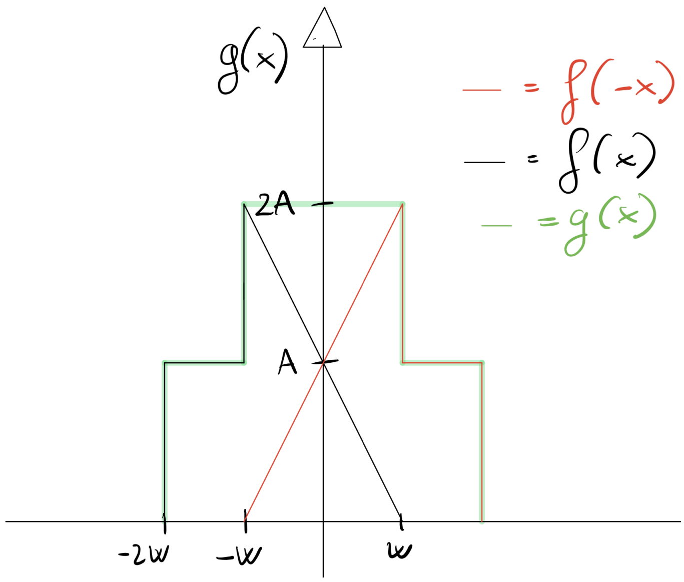


Figure 1: Function $g(x)$

2.2. Point B & C

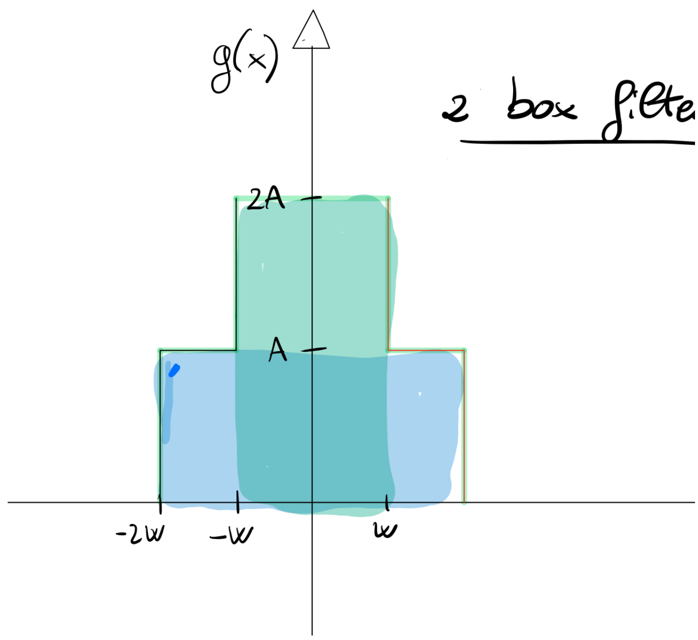


Figure 2: Box filters

Since we already know the Fourier function for box filters, we can write down the Fourier transform of $G(x)$:

$$G(\mu) = 2A\,\frac{sin(2\pi f W/2)}{2\pi f} \;+\; 2A\,\frac{sin(4\pi f W/4)}{4\pi f}$$

2.3. Point D

$$G(\mu) = A\ sinc(fW) + \frac{1}{2}\ sinc(fW) = \frac{3}{2} A sinc(fW)$$

3.

3.1. Bonus

F(w) is defined as :

$$F = \int_{-\infty}^{\infty} g(x) e^{-iwx} dx$$

we also know that :

$$\int_{-\infty}^{\infty} e^{\frac{-x^2}{2\sigma^2}} dx = \sqrt{2\pi}\sigma$$

Using integration by parts , we set:

$$g'(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-x^2}{2\sigma^2}}$$

and its integral:

$$g(x) = 1$$

We set as well f(x):

$$f(x) = e^{-iwx}$$

with the corresponding derivative:

$$f'(x) = -iwe^{-iwx}$$

Using integration by parts:

$$e^{-iwx} - \int_{-\infty}^{\infty} -iwe^{-iwx}$$

We know that the latter integral is equal to the Dirac function. Then we have the final solution which is a gauassian:

$$F(w) = e^{-iwx} + iw\delta(w)$$

4.

According with this, we found out that a rectangular function centered in 0 in Fourier domain is a *sinc* function in the spatial domain. Actually, a rectangular function is a particular case of the more general boxcar function. Indeed writing:

$$rect(w/W)$$

means that we have a rectangular function centered in zero with width of W like in our case. We can write the inverse Fourier transform of the rectangular function as:

$$\mathscr{F}^{-}(rect(w/W)) = W\ \frac{sinc(xW/2)}{\sqrt{2\pi}}$$

Since we have 2 boxes/rectangular functions in Fourier domain we have a sum of this 2 functions in spatial domain (as described above). We recall that a shift of w_0 in frequency domain is like multiplying by:

$$e^{iw_0 x}$$

in the spatial domain (reference here). Writing down all steps we got:

$$e^{iw_0 x}\ \frac{sinc(xW/2)}{\sqrt{2\pi}} + e^{-iw_0 x}\ \frac{sinc(xW/2)}{\sqrt{2\pi}} =$$

$$f(x) = (e^{iw_0 x} + e^{-iw_0 x})\ \frac{sinc(xW/2)}{\sqrt{2\pi}}$$

5.

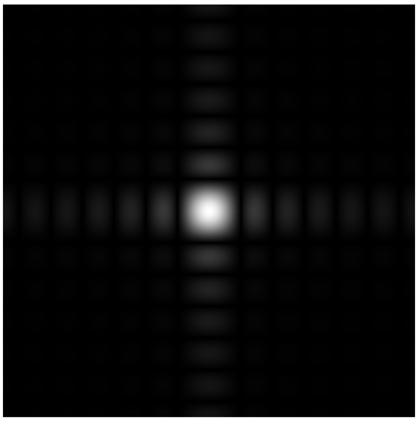
$$g(x) = h(x)\prod(x/s)$$

Where $\prod(x/s)$ is a box filter of height 1 and width S, centered in 0. So by multipliyng, all the values of the Gaussian inside the range $[-S/2, S/2]$ are unchanged, and outside the range they become 0.
From the convolution theorem we know that this multiplication can be expressed as a convolution in frequency domain, so:

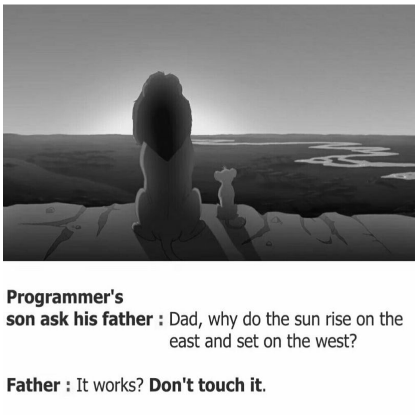
$$G(\omega) = H(\omega) * X(\omega),\ where X(\omega) = \mathscr{F}(\prod(x/s))$$

We used MATLAB to show the Fourier Spectrum (shifted to the center) of a Gaussian kernel with $\sigma = 25$, $G(\omega)$ with $kernelSizeCorrect = 4 * \sigma + 1$, $H(\omega)$ with $kernelSizeWrong = ceil(\sigma/2)$. By using these filters on an image we can see that the correct filter blurs the image as expected, and the wrong filter has a very similar effect as a box filter(We used imboxfilt with $size = kernelSizeWrong = ceil(\sigma/2)$ for comparison)

(a) $G(\omega)$
FT of correct Gaussian kernel



(b) $H(\omega)$
FT of wrong Gaussian kernel



(a) Original Image

(b) With correct Gaussian kernel

(c) With wrong Gaussian kernel

(d) With box filter

6. Gaussian Filtering

To prove experimentally that filtering using a convolution with a Gaussian kernel in space domain with σ_s is equivalent to filtering by element-wise multiplication in frequency domain with $\sigma_f = \frac{1}{2\sigma_s\pi}$. We filter with 3 different methods, 1 in space domain, and 2 in frequency domain, and compare the resulting filtered images using mse. To get consistent and equivalent results we apply the same padding that is used in the conv2 matlab function, before filtering in the frequency domain.

Filtering in space domain

To filter in space domain we simply create a Gaussian kernel using the fspecial matlab function, providing a kernel size of $4 * \sigma_s + 1$. Then use the conv2 matlab function with the 'full' parameter so it returns the result of the convolution that includes padding.

Filtering in frequency domain method 1

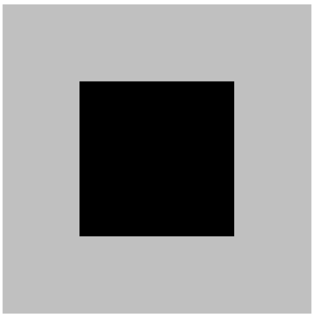
To filter in frequency domain we Fourier transformed the original padded image and the same kernel we used in space domain. Using the fft matlab function providing the image size allows to obtain a filter that is the same size of the image to then element-wise multiply. To obtain the filtered image we inverse Fourier transfomed the result using the ifft function.

Filtering in frequency domain method 2

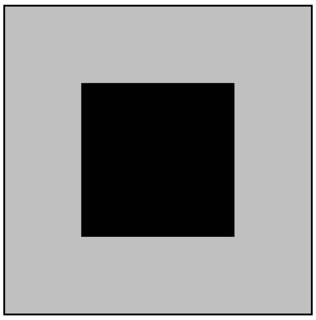
We created an equivalent Gaussian kernel using $\sigma_f = \frac{1}{2\sigma_s \pi}$, we need the kernel to be the same size of the padded image to be able to element-wise multiply in frequency space, so we used fspecial providing the image size, and $\sigma_f * imagesize$. We can then directly element-wise muliply this kernel to the fourier transformed image. Then with ifft we obtain the result.

Comparing results

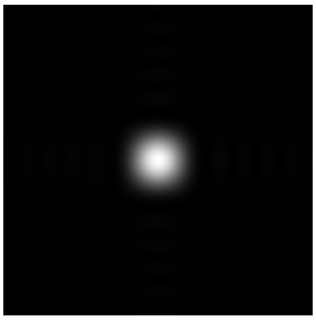
We compared the resulting images calculating the mse of the differences:
$mse = 8.7e - 05$
The results are the same, showing that these three methods are equivalent.



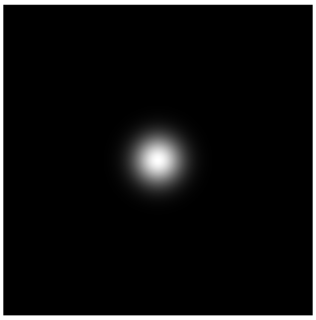
(a) Original image



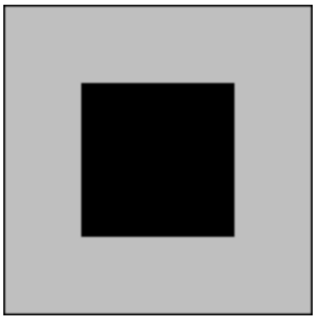
(b) Original Image Padded



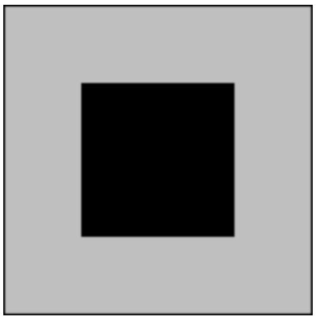
(a) $\sigma_s = 3$ Gaussian filter Freq domain method 1



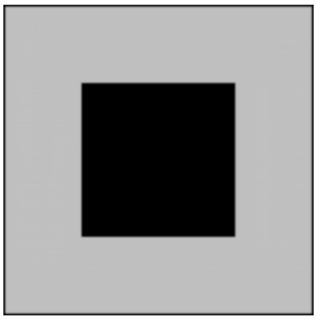
(b) σ_f Gaussian filter Freq domain method 2



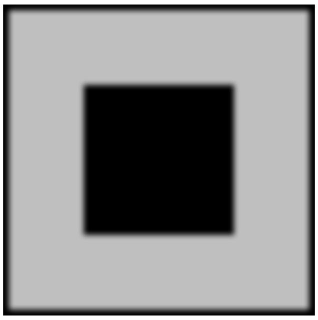
(a) Filtred $\sigma_s = 3$
Space domain



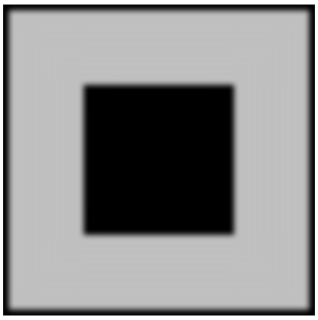
(b) Filtred $\sigma_s = 3$
Freq. domain method 1



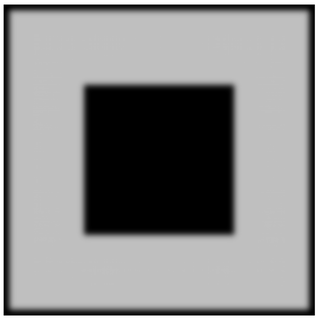
(c) Filtred $\sigma_s = 3$
Freq. domain method 2



(d) Filtred $\sigma_s = 10$
Space domain



(e) Filtred $\sigma_s = 10$
Freq. domain method 1



(f) Filtred $\sigma_s = 10$
Freq. domain method 2

6.1. BONUS: Execution time vs σ_s

Figure 8 shows that filtering in spatial domain is more efficient with smaller values of σ_s, but increases linearly, it is related to the size of the kernel. The efficiency of filtering in frequency domain just depends on the size of the image, which in this case is constant, so it becomes more efficient with higher values of σ_s.

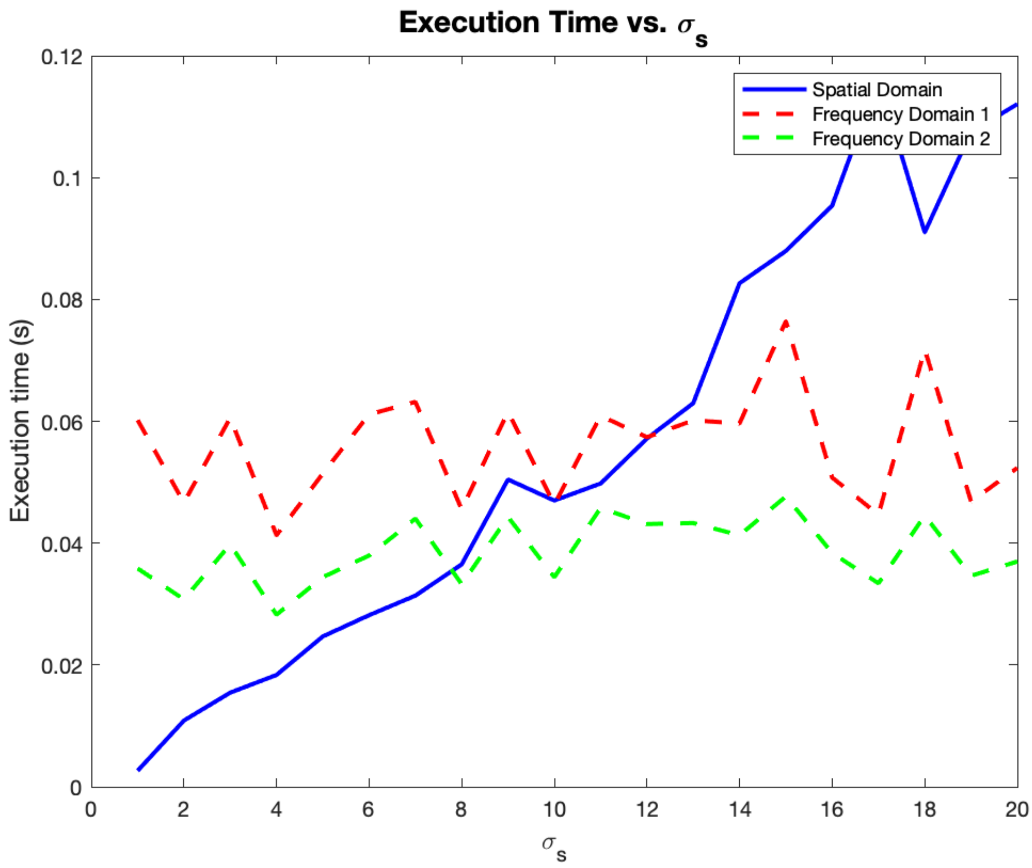
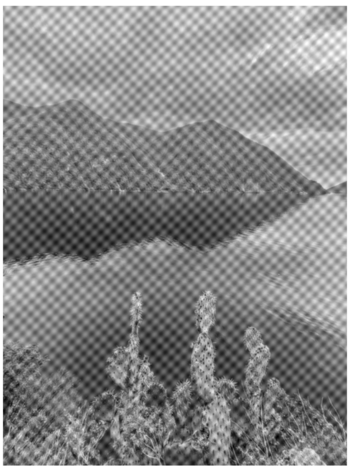


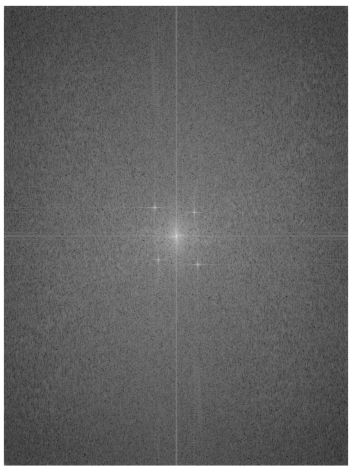
Figure 8: Time vs σ_s

7. Image Restoration

Observing the Fourier Spectrum of the original San Domenico image we can see 4 white dots that correspond to the frequencies that we should filter out from the image. This can be done using Notch filtering, to do it we simply implemented a mask with black circles and element wise multiply it to the Fourier spectrum.

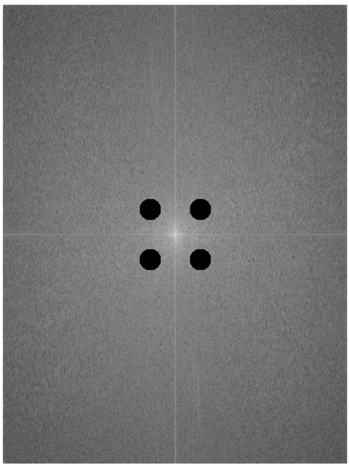


(a) Original image



(b) Fourier Spectrum

(a) Result of Notch Filtering



(b) Fourier Spectrum Notch Filtering

8. Bonus