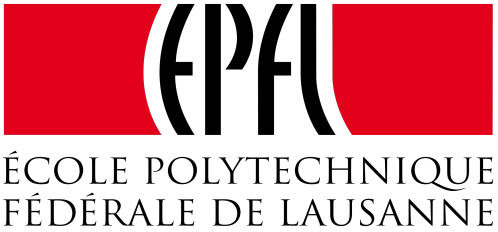

EE-550 Laboratory 1

Getting started with Matlab, Quantization, Sampling, Filtering, 2DFT and Weber's law

Image and Video Processing

Academic Year 2020/21

Gloria DAL SANTO
320734

Exercise 1: Images and color tables

1-2. The images used in this exercise were "trees.tif", which is in *indexed* format, and "lena.tif", with *truecolor* format. To show them in gray level I applied rgb2gray function on the colormap matrix of "trees.tif" and on the matrix correspondent to "lena.tif". This function makes a weighted sum of the R, G and B components where the coefficents are those used to calculate the luminance. To obtain their negative I subtracted the values of the same matrices from the maximum value of their corresponding format (1 for *indexed* and 255 for *truecolor*). Figures 1 and 2 illustrate the obtained images.

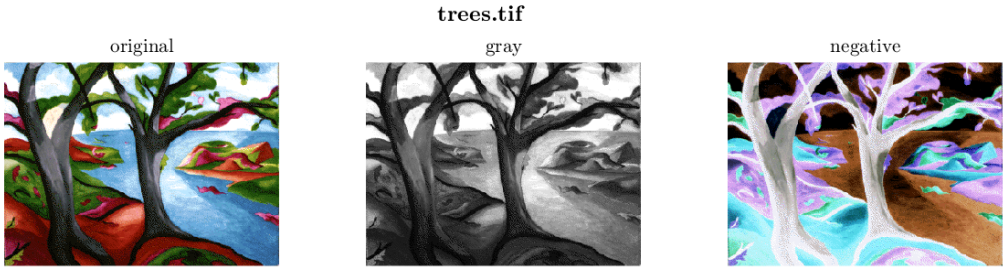

Figure 1: Original, gray and inverted versions of image *trees.tif*

Figure 2: Original, gray and inverted versions of image *lena.tif*

3. To modify the color table of "trees.tif" through a gamma correction I raised the colormap to the power of γ. This operation allows to modify the color values directly, without the need of scaling them. Being the RGB matrix elements in the range $[0\,, 255]$ I preferred to use imadjust function which maps the values according to the argument γ using the default contrast limit $[0\,1]$. In Figure 3 and 4 it can be observed that the gamma correction modifies the luminance through a non-linear relation that gives more weight to low pixel values in the case $\gamma > 1$, resulting in a darker image, while for $\gamma < 1$ it gives more weight to high pixel values resulting in a brighter image.

Figure 3: Gamma correction in *trees.tif*

Figure 4: Gamma correction in *lena.tif*

4. To create the $8x8$ chess board image I first generate a $8x8x3$ matrix with the numeric values corresponding to the *truecolor* format and then convert them into $\mathrm{uint8}$ data type to obtain the *truecolor* image. To get the image in *indexed* format I used the $\mathrm{rgb2ind}$ function that converts the image using the minimum variance quantization with the two colors.

Exercise 2: Image quantization

Figure 5 shows the image "lena.tif" quantized for different quantization step sizes. It can be noticed that when the number of gray levels is equal to 16 ($\Delta = 16$) the phenomenon of false contours appears and becomes even worst for lower number or levels. On the other hand, the images with 128 and 64 gray levels do not appear different from the original one.

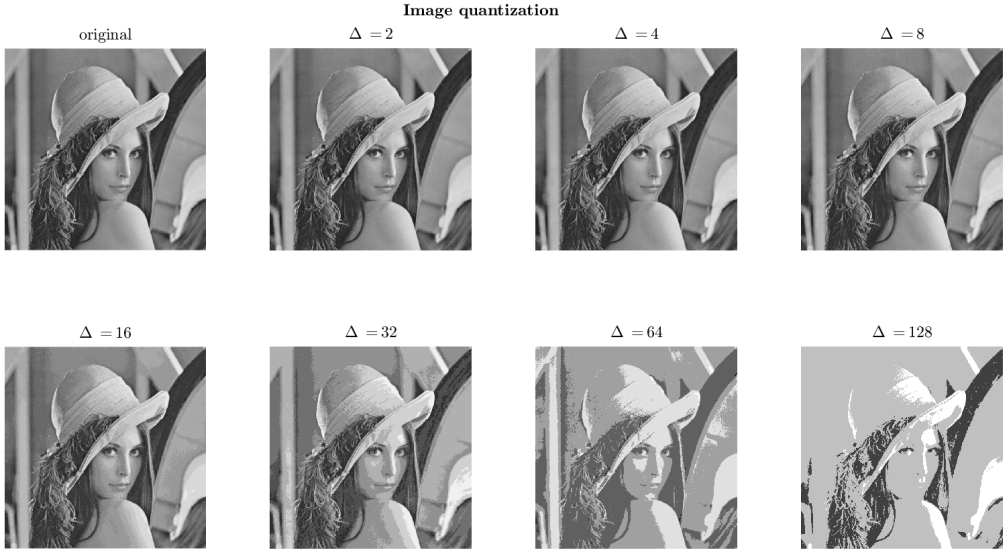

Figure 5: Effect of different quantization steps

Exercise 3: Filtering

The first filter proposed in this exercise is depicted in Figure 6 and it can be seen that it is a low pass filter. The output of the convolution between the filter and the image (Figure 8) is blurry and has less defined contour compared to the original image. Applying the convolution between the resulting image with the second filter, which is a high pass filter (Figure 7), the image gained more details and

sharper contours, compensating the effects of the low pass filter as it can be seen in Figure 9. The high pass filter caused some changes also in the intensity values in such a way that the image appears paler.

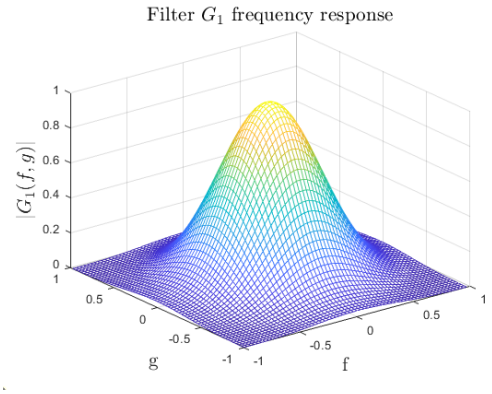

Figure 6: Frequency response of $g_1(k,l)$

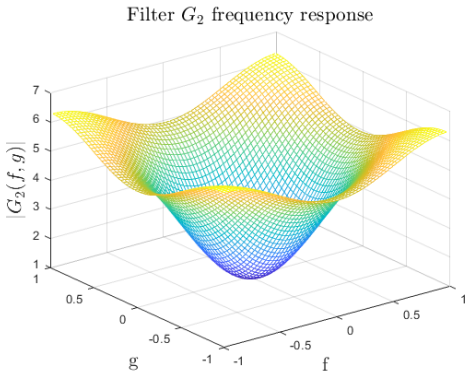

Figure 7: Frequency response of $g_2(k,l)$

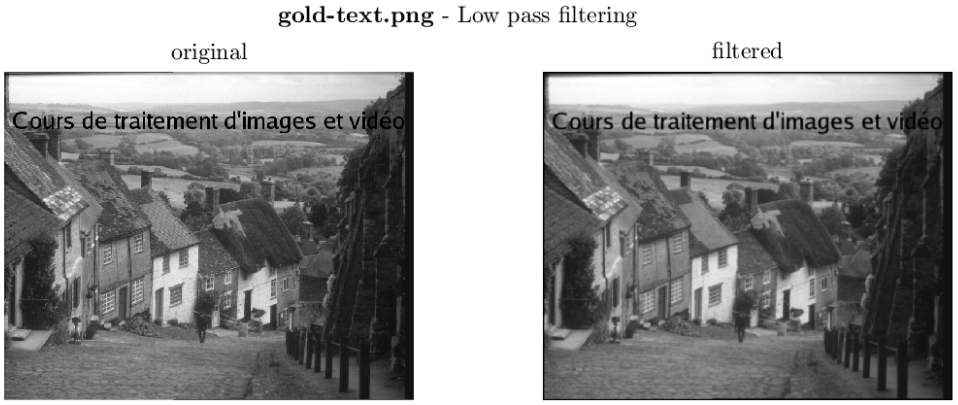

Figure 8: Original and LP filtered image "gold-text.png"

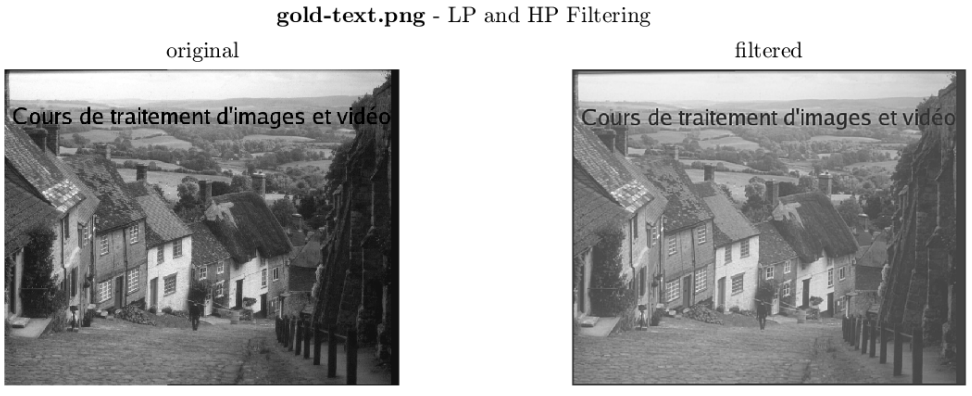

Figure 9: Original and filtered image "gold-text.png"

To compute the convolution I have directly used the conv2 function, with the additional argument "same", to obtain only the central part of the convolution having the same size as the image.

Exercise 4: Correlation

To implement the correlation between the filtered "gold_text.png" and "g_letter.png" I used the $\mathrm{conv2}$ function. To do so I first inverted the entries of the second image and then I computed its conjugate while leaving the filtered image unchanged. In this way the output of the function is equivalent to the output of the correlation of the two images. For the frequency domain I first computed the Fourier transform by means of the combination of functions $\mathrm{fftshift}(\mathrm{fft2}(\cdot))$ in order to have the zero frequency in the centre of the plot and then I used the same procedure as for the spatial domain.
Figure 11 shows the two correlation functions of the images without noise. The peak that can be seen in the spatial domain corresponds to the maximum correlation, or similarity, between the two images and indeed it has the same coordinates as the letter "g" in "gold_text.png": bottom right corner at $[109, 504]$. Observing the results in frequency domain, it can be deduct that the "g_letter.png" has very low frequency components since the peaks are located around the coordinates relative to zero frequency. With additional noise (Figure 10) the absolute peak in the spatial domain becomes less distinguishable as the standard deviation increases. Experimentally, I obtained that with $\sigma > 25$ the true maximum of the spatial correlation is overcome by other noisy values and therefore the position of the letter "g" can not be identified correctly.

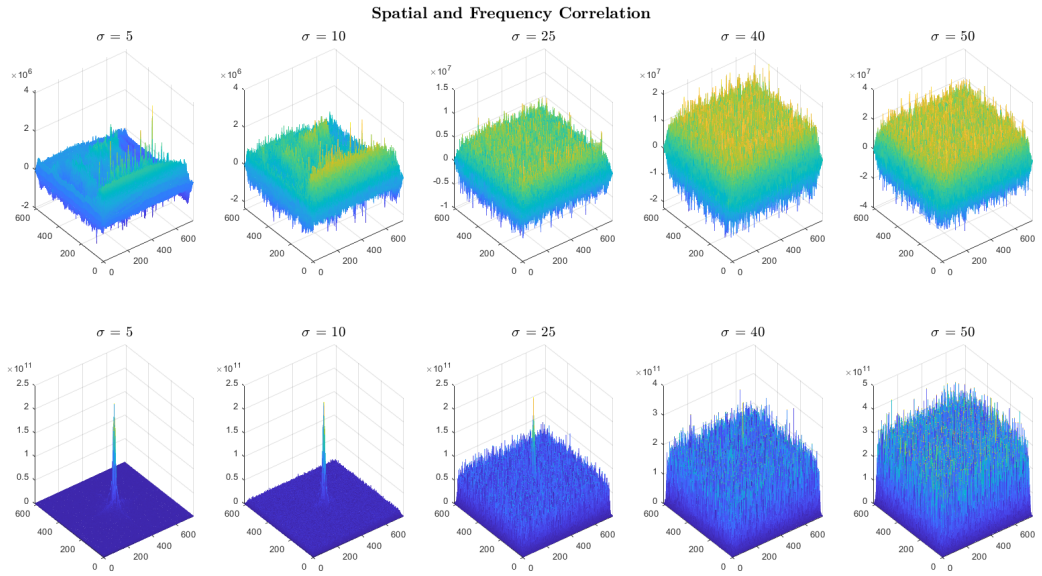

Figure 10: Spatial and frequnecy correlation - noisy scenario

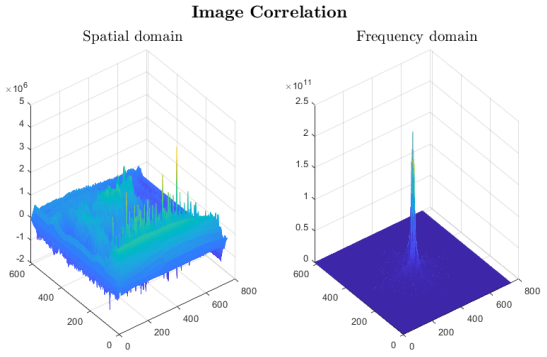

Figure 11: Spatial and frequency correlation - noiseless scenario

Exercise 5: Resampling

The downsampling operation (Figure 12) led to the appearance of additional frequency components that interfere with the original image and which result in the Moiré patterns. Moreover, it can be noticed that in the case of a resampling factor of 4 the numbers in the image appears almost indistinguishable.

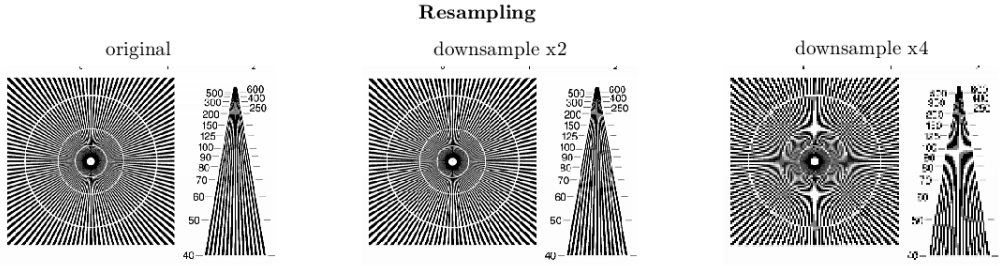

Figure 12: Effect of downsampling operation

Exercise 6: Phase and magnitude of the 2DFT

In this exercise I used the $\mathrm{fft2}(\cdot)$ function to perform the 2D Fourier transform. This function is based on the same 2DFT formula but it ensures the highest computational speed, in particular for matrices with dimensions of length 2^n, as in this case.

1. In class we have seen that for all real signals the real part of its 2DFT is always an even function, while the imaginary part of the 2DFT is odd. In Figure 13 it can be seen that taking the inverse 2DFT of the two Fourier transforms, an even and an odd function, I obtained again one even image and one odd image respectively, proving also the other symmetry property related to the Fourier transforms of real functions. The images in the figure are related by the following equation:

$$x(k,l) = F^{-1}\{Re[F\{x(k,l)\}]\} + F^{-1}\{Im[F\{x(k,l)\}]\}$$

2. Figure 14 shows that the magnitude of the Fourier transform apparently does not give much information about the structure of the image, whereas from the phase we can directly retrieve the information about the edges and contours of the image which are visible as darker or brighter lines.

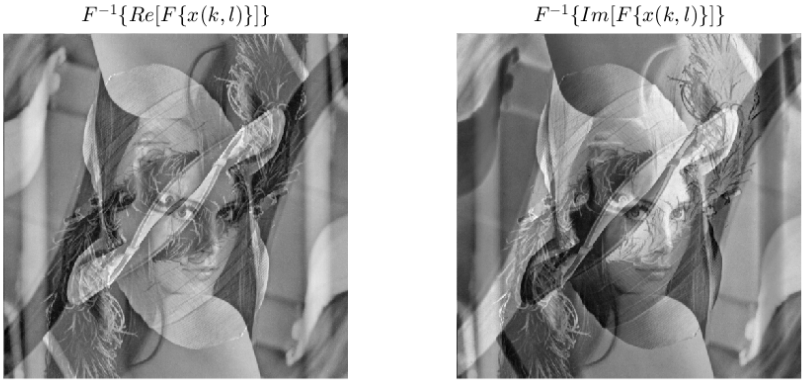

Figure 13: Real and imaginary parts of the DFT

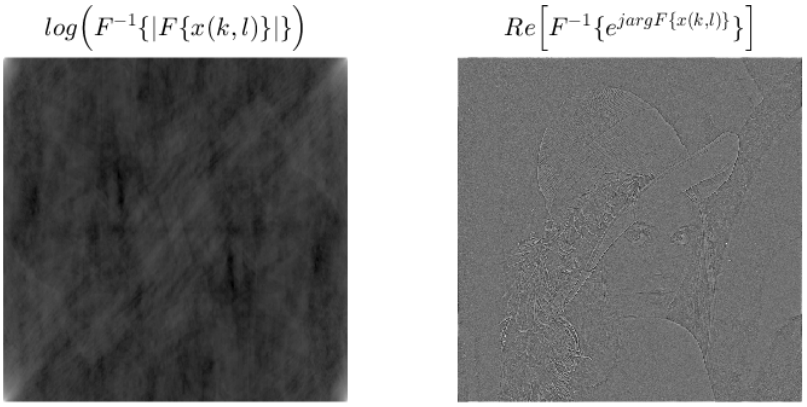

Figure 14: Phase and magnitude of the DFT

Exercise 7: Weber's law

The values of the Weber constant that have been found experimentally are depicted in Figure 15. It can be noticed that for lower and higher values of L_1 it became increasingly difficult to perceive the intensity difference and thus larger steps ΔL were needed. At lower L_1 the constant takes its absolute maximum values which means that, no matter the background, differences between darker intensities were harder to detect. However, with the brighter background ($L_b = 200$) the difference between darker intensities took larger ΔL to be perceived with respect to the result obtained with the darker background. This trend is inverted if greater values of L_1 are considered.
The average Weber constant is $C_w = 0.0182$ which is consistent with what we have seen in class.

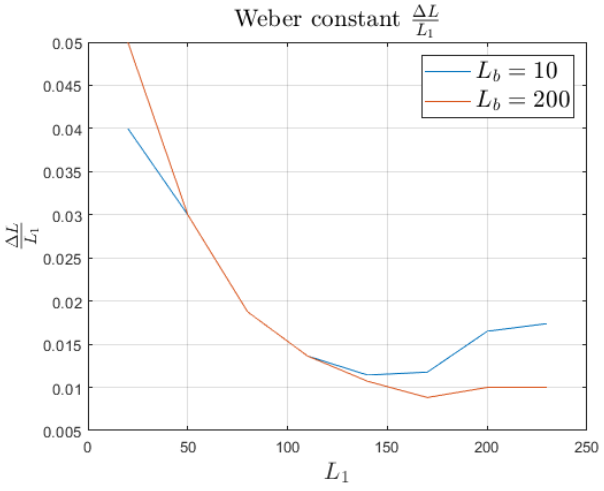

Figure 15: Weber constant for $L_b = 10$ and $L_b = 200$