UNIVERSITÉ DE GENÈVE
FACULTÉ DES SCIENCES
Département d'informatique

Imagerie Numérique 2023 Automne

December 8, 2023

TP Class N°6 - Spatial filters

Instructions :

- This TP should be completed and uploaded on Moodle before **Thursday 21 December 2023, 23h59**.
- The name of the file you upload should be **TP6_name_surname.ipynb**.
- If you need to include attached files to you TP, please archive them together in a folder named **TP6_name_surname.zip**.

Exercise 1. Explain the code *(0.25 point)*

Run the following code. Explain the difference in the produced results. Make conclusion.

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt

def plot(I, Titles, dr=1, dc=1, cmap="gray"):
    fig = plt.figure()
    for i in range(len(I)):
      plt.subplot(dr,dc,i+1)
      plt.imshow(I[i], cmap=cmap)
      plt.title(Titles[i])
      plt.axis("off")

    plt.show()
```

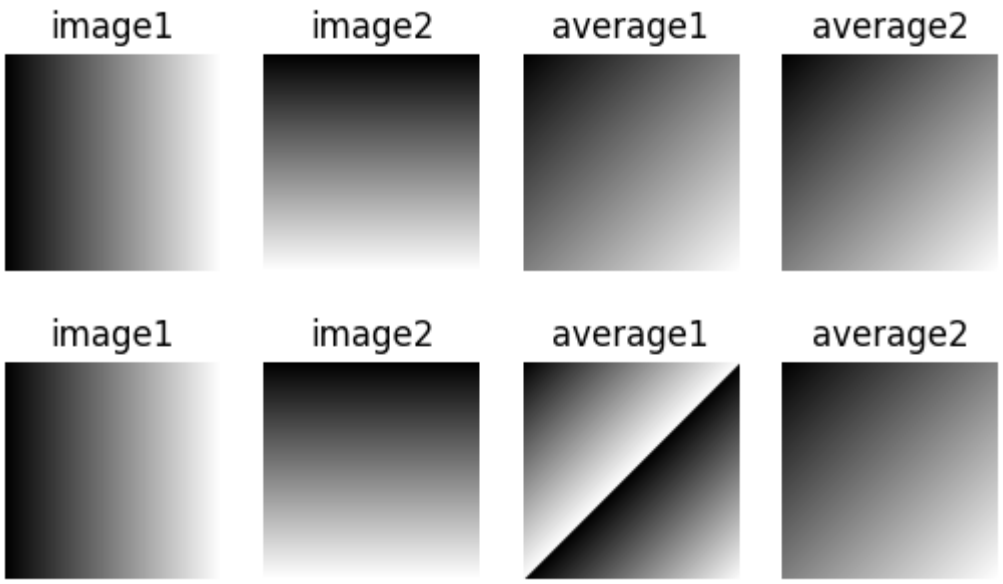
In [2]:
```
image1 = np.arange(0, 255)*np.ones((255, 255))
image2 = np.transpose(np.copy(image1))
av1 = (image1 + image2)/2
av2 = image1/2 + image2/2

plot([image1, image2, av1, av2],
     ["image1", "image2", "average1", "average2"],
     dc=4)

image1 = image1.astype(np.uint8)
image2 = image2.astype(np.uint8)
av1 = (image1 + image2)/2
```

```
av2 = image1/2 + image2/2

plot([image1, image2, av1, av2],
     ["image1", "image2", "average1", "average2"],
     dc=4)
```



Exercise 2. Convolution and correlation *(0.75 point)*

Let there be the functions:

$$
\begin{aligned}
x(n) &= \delta(n-1) + \delta(n-2) + \delta(n-3) \\
h_1(n) &= \delta(n) + 2\delta(n-1) + 3\delta(n-2) \\
h_2(n) &= \delta(n+2) + 2\delta(n+1) + 3\delta(n) + 2\delta(n-1) + \delta(n-2)
\end{aligned}
$$

(a) Convolve manually $x(n)$ with $h_1(n)$ and $h_2(n)$.

(b) Correlate manually $x(n)$ with $h_1(n)$ and $h_2(n)$.

Exercise 3. Low Pass Filtering *(1 point)*

(a) Program a function that performs image filtering with a square filter of arbitrary size.

> **Important:** try to program with a min amount of loops. Ideally without any loops.

(b) By using your function, perform the filtering of zero padded grayscale image *cameraman.jpg* with an average/box filter:

$$h = \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Hint: for zero padding the corresponding *numpy* function could be used.

(c) By using your function, perform the filtering of zero padded grayscale image *cameraman.jpg* with a Gaussian filter:

$$\mathbf{h} = \frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$

(d) How could the average/box and Gaussian filters to be modified to increase the blurring effect, i.e. remove more high frequency components?

Exercise 4. Border effects *(1 point)*

Generate an image (as illistrated in Figure 1) of size 512 $\times$ 512.

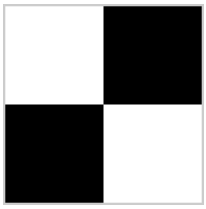


Figure 1.

Using the *numpy* padding function and the box filter of size 31 $\times$ 31 (use your implemetation from the exercise 1):

(a) investigate the influence of tree types border padding:

- zero padding
- periodic extension
- symmetric flip or mirror reflection

(b) Visualize the original, padded and filtered images. Explain the effects that one can observe.

(c) What are the advantages and disadvantages of each type of border extension?

Exercise 5. The Laplacian filter *(1 point)*

The Laplacian filter kernel:

> $$\nabla^2 f(x,y) = f(x-1,y) + f(x+1,y) + f(x,y-1) + f(x,y+1) - 4f(x,y)$$

(a) Program and test (apply to the grayscale image *cameraman) the Laplacian filter based on the equation (1) and 3 $\times$ 3 neighbourhood.

> **Hint:** you can represent the equation (1) in a matrix form and use your implementaion from the exercise 1.

(b) Visualise the original and filtered images. Comment the effect of filtering.

Exercise 6. Mechanism of sharpening *(1 point)*

Take the grayscale image *cameraman* A. Blur this image with a box filter of size 3 $\times$ 3 B.

(a) E_1: defining the optimal parameters perform the sharpening of B using the Gaussian filter.

(b) E_2: defining the optimal parameters perform the sharpening of B using the Laplacian filter.

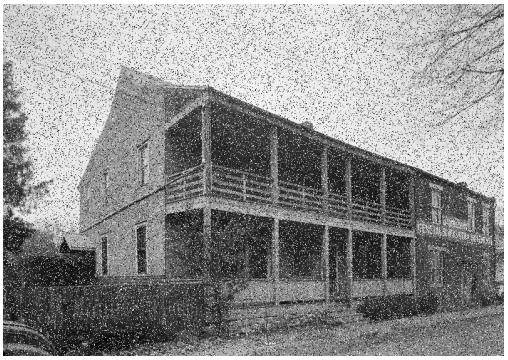
> **Hint:** try to find the parameters that minimise the *mse* between the original A and enhanced images E_1 and E_2.

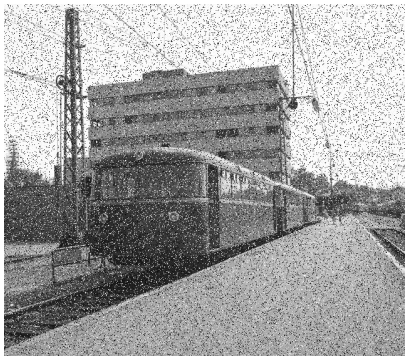
(c) Display blurred B and enhanced images E_1 and E_2. Comment the quality of the enhanced images based on the visual quality and based on the *mse* between the original A and enhanced images E_1 and E_2.

> **Hint:** for this exercise use the formulas from the Theme 6 page 56.

Exercise 7. Order statistics filtering *(1 point)*

You are given two noisy images *tp6_005* and *tp6_006* corrupted by "impulse" noise.





(a) tp6_005 (b) tp6_006

Figure 2.

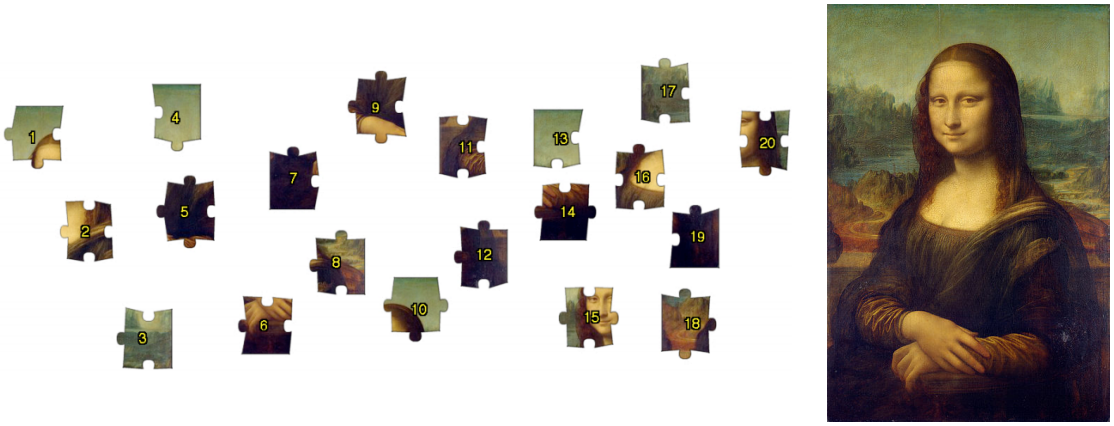
(a) Defining the optimal parameters (filter size) by yourself try to remove the noise with:

- averaging filter
- median filter

(b) Visualise the images before and after filtering. Report the used parameres of filtering. Comment the denoising efficiency based on the visual image quality.

Exercise 8. Solving a Jigsaw Puzzle *(1 point)* Non obligatory, bonus exercise

The goal of this exercise is to use image processing to automatically solve a jigsaw puzzle. The image *tp6_003* (see Figure 3.a) shows the pieces of a jigsaw puzzle. The reference image *tp6_004* is shown in Figure 3.b. Note that when the puzzle pieces are assembled together, the completed puzzle has potentially a different area than the reference image.



(a) Puzzle image tp6_003 (b) Reference tp6_004

Figure 3.

Implement an image processing algorithm to automatically locate where each jigsaw piece fits in completed puzzle. Clearly describe the steps of your algorithm, showing intermediate results for clarification if helpful.

> **Hint:** for the cross-correlation you might use *scipy.signal.correlate2d(.)*.
>
> **Important:** for cross-correlation both images should be normalized (zero mean, unit variance).

In [ ]: