TP2: Basic Cryptography and Watermarking

by Martino Ferrari

1 Encryption

Exercise 1

After implementing a function to permutate an grey-scale image we were asked to display both the image and the histogram of it.

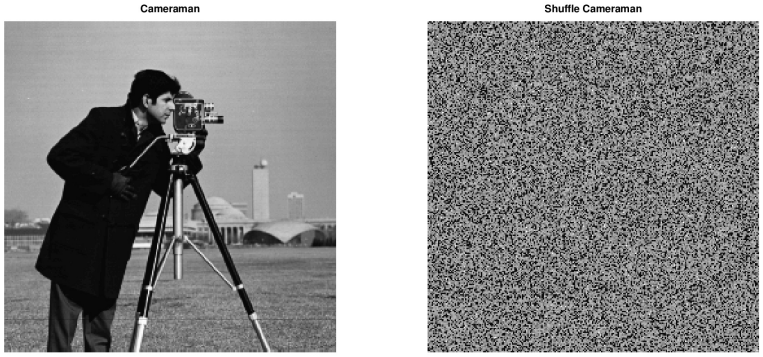

Figure 1. `cameraman.bmp` and a random permutation of it

How previsible even if the two pictures are totally different their histograms are exactly the same, as the values of the pixels is preserved (not theirs position however).

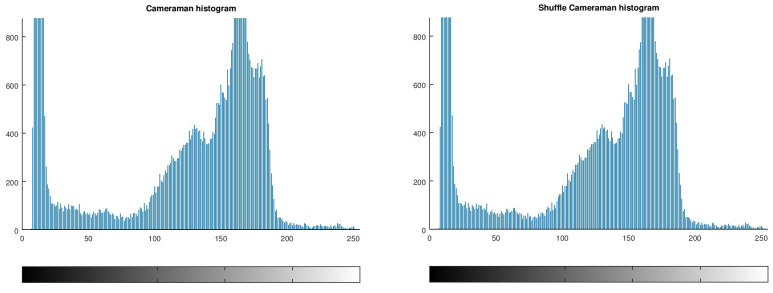

Figure 2. Histogram of `cameraman.bmp` and his permutation

Exercise 2 and 3

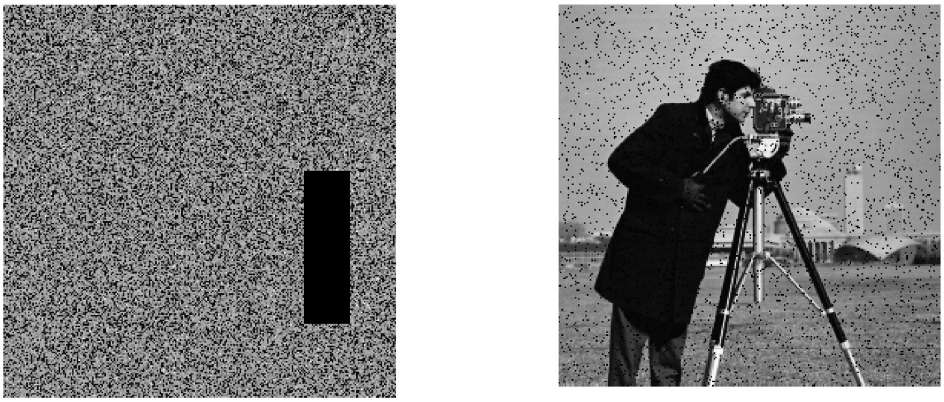

Figure 3. Block loss in the shuffle image and its reconstruction

After implementing a simple function simulating the loss of one block of the image I applied this function to the permutated `cameraman.bmp` image (with a loss of a block of size $50 \times 100\,\text{px}$).

After reconstructing the original image using the permutation vector it possible to see how the black pixels are now distribuited randomly in the picture (as the pixel where shuffled by the original permutation). In this way the result of the loss is less desctructive than if the same loss were heppen to the original image.

Exercise 4

In this exercise we will add some uniform distributed noise to the image `cameraman.bmp` with different intensities (± 1, ± 5, ± 10, ± 15) and we will see how this impact both the image and its histogram.

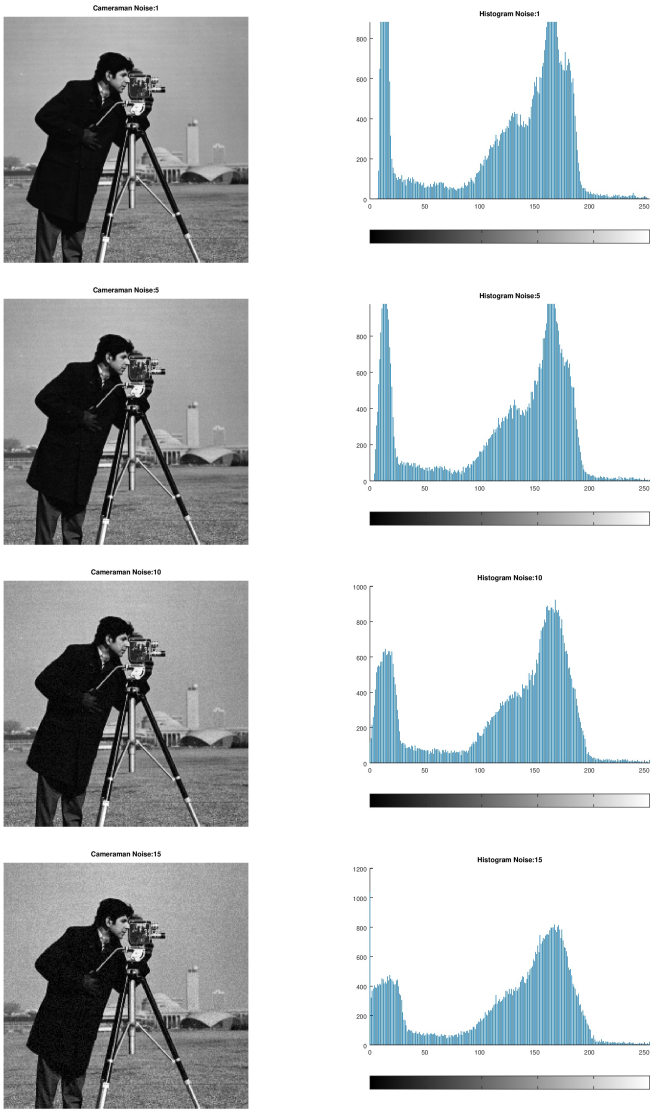

Figure 4. Noisy `cameraman.bmp` and its histogram

In figure 4 it's possible to see that the image change a little (even if with noise between ± 15 the degradation is visible) while the histogram change visibily.

In particular more the noise is strong more the instogram become flat, that's beacouse the histogram of a uniform distribution is flat and more the noise is intense more the the histogram will resamble to the noise one.

This shows how the histogram is not so semantically relevant as in the exercise 1 we had totally different images (semantically) with the same histogram while here we have the same images but with different histograms.

As previsible the PSNR decrease with the intensity of the noise as shown in figure 5.

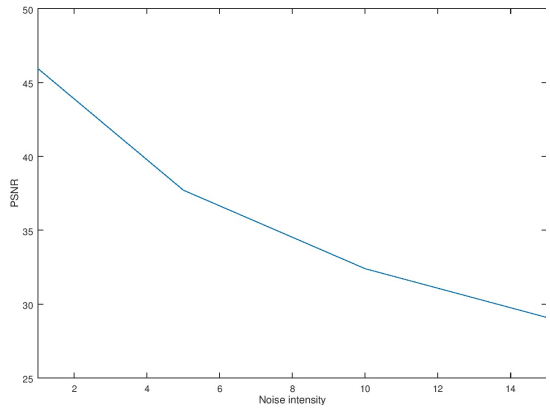

Figure 5. PSNR vs noise intensity

2 Classical Cryptography

After modifing the given script I obtained the following results:

original 1px difference gaussian noise

Figure 6. different decripted images

The script takes the `cameraman.bmp` picture, then encrypts it using the AES algorithm and then decript it.

The original results is shown in the left picture of figure 6.

The central picture is the decription after modifing a single pixel of the cripted image.

Finally the rigth one is the result after adding a gaussian noise $\mathcal{N}(0, 2)$ to the cripted image.

As the AES algorithm is a classical cryptography algorithm when a single bit of the crypted message change the decripted result is totally different.

However the given script crypt every windows $[4 \times 4]$ indipendently, that's why when forcing a single pixel to 0 only 16 pixels change.

However adding a (weaker) gaussian noise to all the crypted message results in a totally random image.

3 Basic Data Hiding

In this last exercise we will hide one image in another with the same resolution, to do so we will exploit the less significant bits of the *cover* image to carry the more significant bits of the *secret* image. Moreover we will use the fact that the human eye is less sensible to the blue to hide more data in this components.

Figure 7. *cover* image and *secret* image

In particular we will hide the data in this way:

red	$c_{r,7}$	$c_{r,6}$	$c_{r,5}$	$c_{r,4}$	$c_{r,3}$	$s_{r,7}$	$s_{r,6}$	$s_{r,5}$
green	$c_{g,7}$	$c_{g,6}$	$c_{g,5}$	$c_{g,4}$	$c_{g,3}$	$c_{g,2}$	$s_{b,7}$	$s_{b,6}$
blue	$c_{b,7}$	$c_{b,6}$	$c_{b,5}$	$s_{g,7}$	$s_{g,6}$	$s_{g,5}$	$s_{b,5}$	$s_{b,4}$

Table 1. color structure in the stego image

The stego image and the extracted hidden image is shown in figure 8:

Figure 8. Stego image and extracted hidden image

How it possible to see both the stego image and the extracted secret image are well conserved. Some degradations are visibile in both as visible in figure 9:

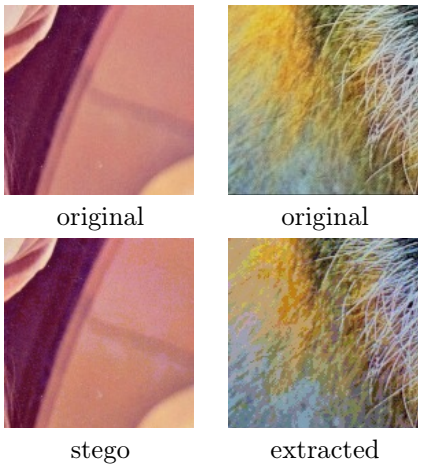

Figure 9. Difference between the original images and the stego and extracted images

However this differences are minors and the result can be improved using some image adaptative algorithm to adapt the color structure to the *cover* image and by hiding more green components of the *secret* image.