Computer vision course

Lab 2 - Filters and histogram equalization

Pietrobon Andrea

27 maggio 2023

1 | Task 1

In the first activity I had the chance to understand the behavior of the cv:cvtColor() function and I discovered that in OpenCV an RGB image is saved in a variable of type Mat in three inverted channels, such as BGR (the first blue channel, the second green and the third red). I used the COLOR BGR2GRAY parameter to transform the image into a 1 channel grayscale.

Figura 1.1 Grayscale image obtained from the original RGB.

2 | Task 2

The second task was slightly more complicated. I ran into some initial difficulty as the image didn't change (don't know why yet). Next, applying maxfilter (2.1) and minfilter (2.2) were the hardest spots. To define these functions, I had to think about how to define a kernel (filter), take a part of the whole image (with the same kernel size), calculate the maximum pixel value of all filters, and replace it with the average value in the nut. Also, one mistake that cost me a lot of time was to use the same modified image to compute subsequent kernel values. This meant that every image was ruined by the filter. Once this small but annoying problem has been fixed, we still need to figure out how to apply the filter to the last row and last column of the image. (I thought about adding rows and columns to the image, but I'm sure that's not the correct implementation). In any case, as you can see in the image below, the maxfilter was the best filter to remove cables from the background, and the perfect kernel size not to corrupt the image too much was 5. The minfilter, however, put highlight the dark details even more.

Figura 2.1 Grayscale image with Max Filter and kernel size 5.

Figura 2.2 Grayscale image with Min Filter and kernel size 5.

3 | Task 3

Not too difficult task. I found the functions on the openCV documentation very easily and I implemented them by increasing the ksize to see the differences well as shown in (3.1) and in (3.2).

Figura 3.1 Grayscale image with Median Filter and kernel size 15.

Figura 3.2 Grayscale image with Gaussian Filter and kernel size 15x15.

4 | Task 4

Implementation-level simple task. Once you found an example on the official openCV documentation, you just had to apply the steps to the letter. The most ”complicated” part was actually understanding the various steps and what they were going to do to recreate the histogram. In the images below we can see the original image in the (4.1) and its histogram in the(4.2).

Figura 4.1 Original Image.

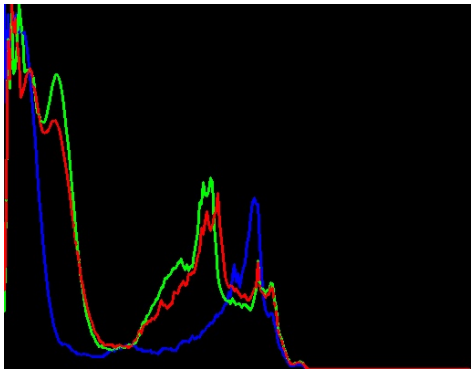

Figura 4.2 Histogram of the Image with 256 bins.

5 | Task 5

To complete this last task it was sufficient to use cv::equalizeHist() on the grayscale image and then slightly modify the code of task4 by removing the extra channels. I did not find any particular difficulties in carrying out this task.The equalized image and the corresponding histogram are shown below on 5.1 and 5.2.

Figura 5.1 Equalized Image.

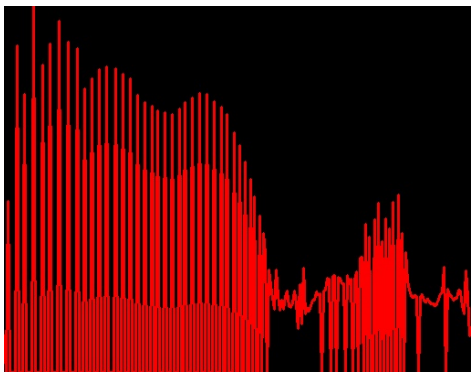

Figura 5.2 Histogram of the Image.