CP467: Assignment 2 Report

Duc Minh Nguyen - 203009140

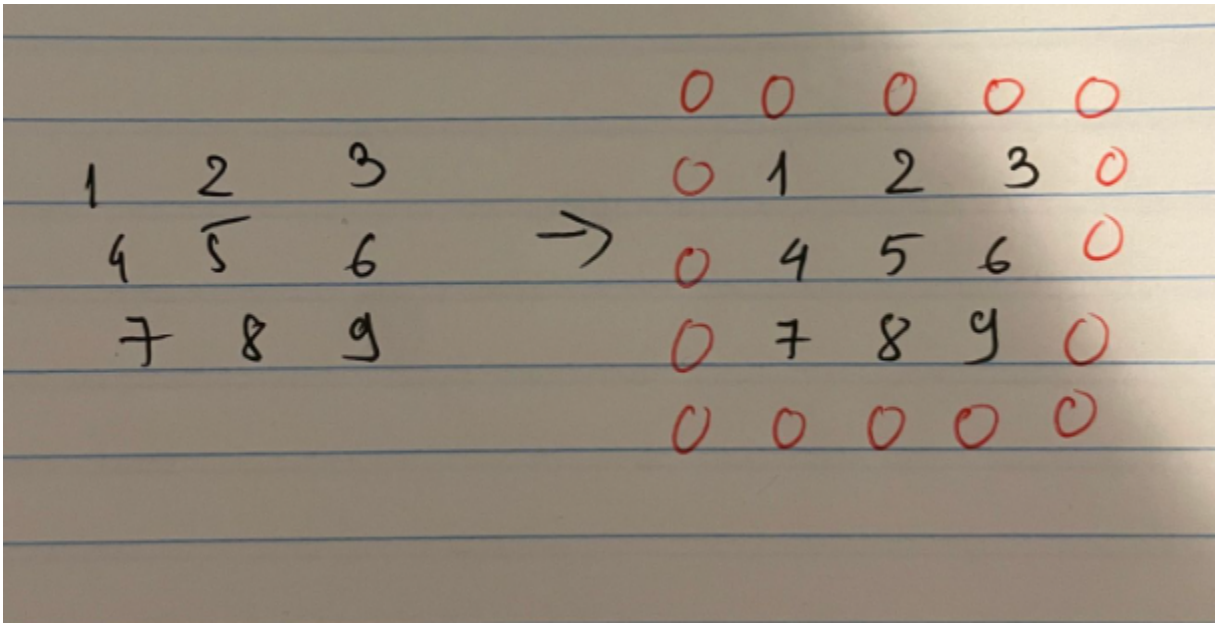
Overview

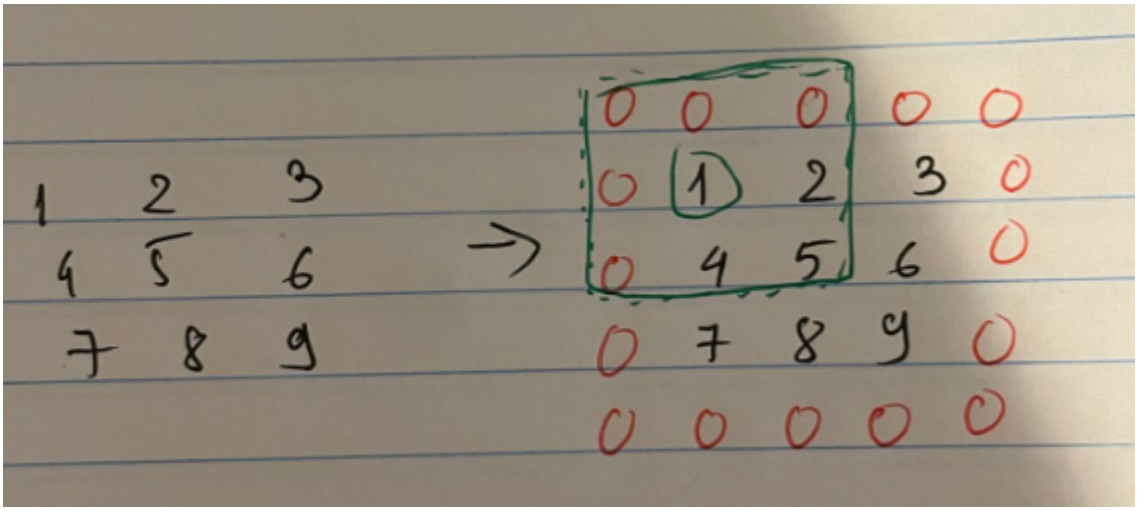
I. Task 1

1. Average Smoothing Filter (filter size 3*3)

Brief theoretical:

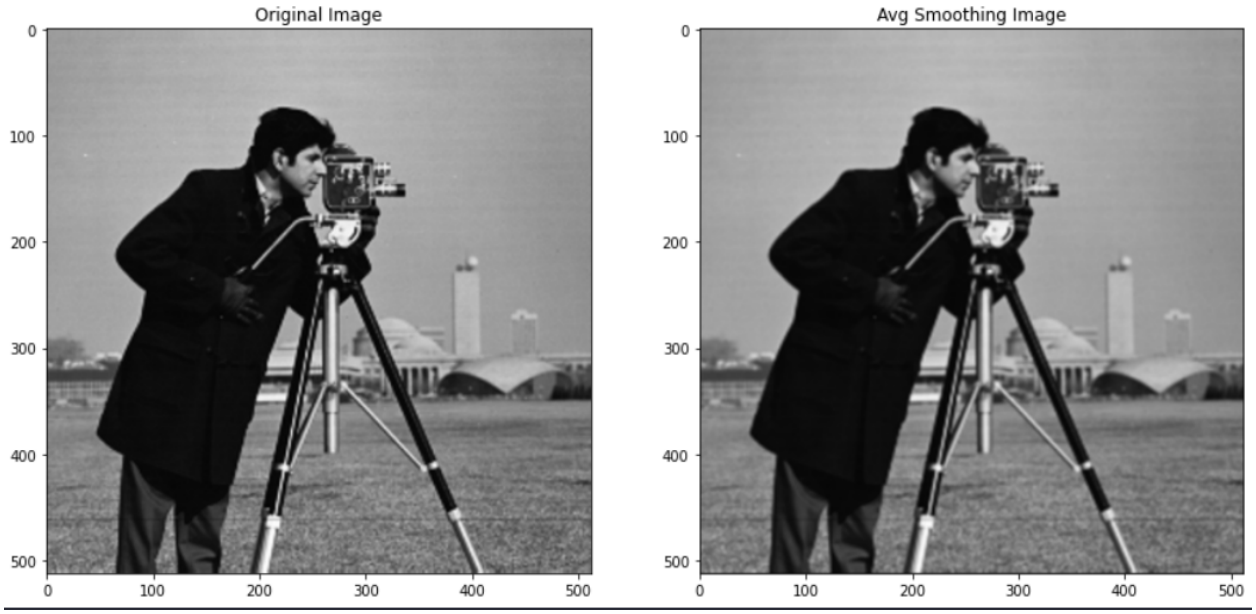
- The filter applying the filter size (n * n) and move one by one to calculate the pixel, first step was to pad the image by adding another line
- Then the filter will keep moving through the image

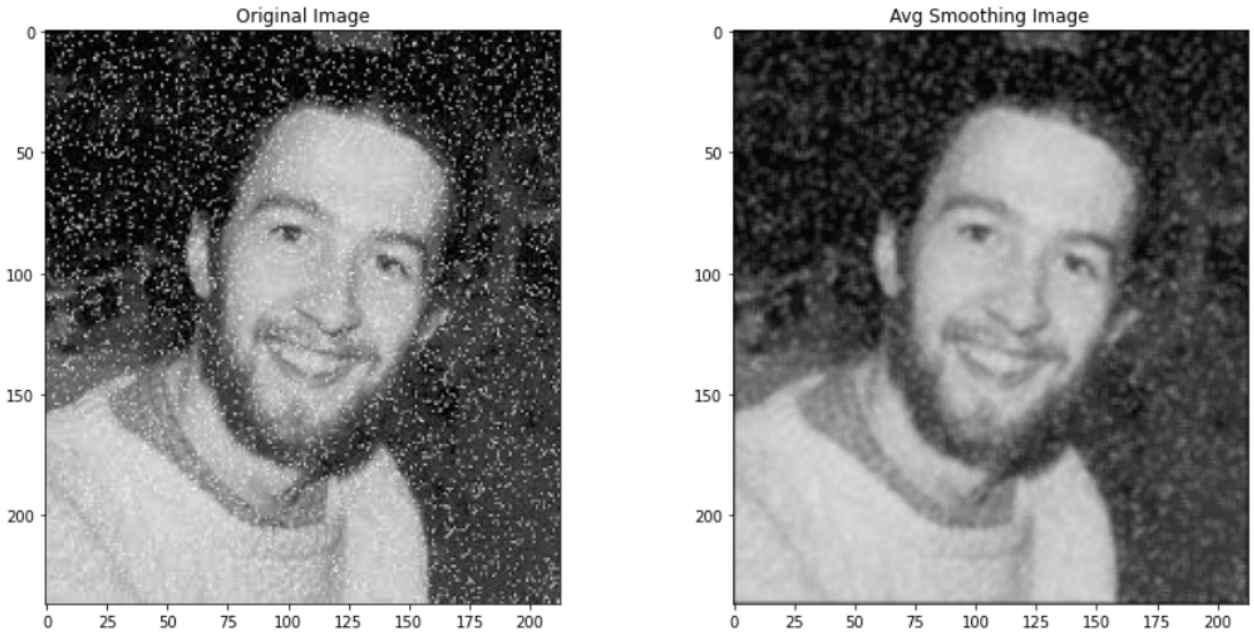




Quality result:

- The image is more smooth now, the noise is significantly reduce, however the quality of the image is more blurry, because the filter is averaging the surrounding pixels
- Cameraman image is a bit hard to visualize, so I install an image online that have noise

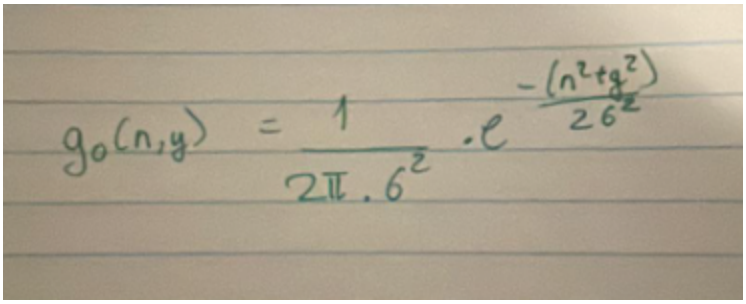




2. Gaussian Smoothing Filter (filter size 7*7)

Brief theoretical:

- The Gaussian filter work based on the mathematical formula to generate a filter and apply to the image

- 

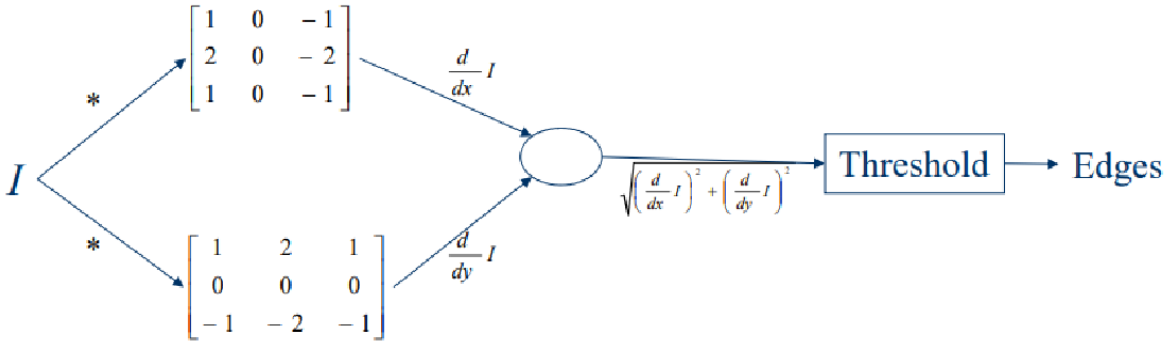
Quality result:

- The image is more blurry than the average smoothing filter , however the different is not too significant

3. Sobel Sharpening Filter (filter size 3*3)

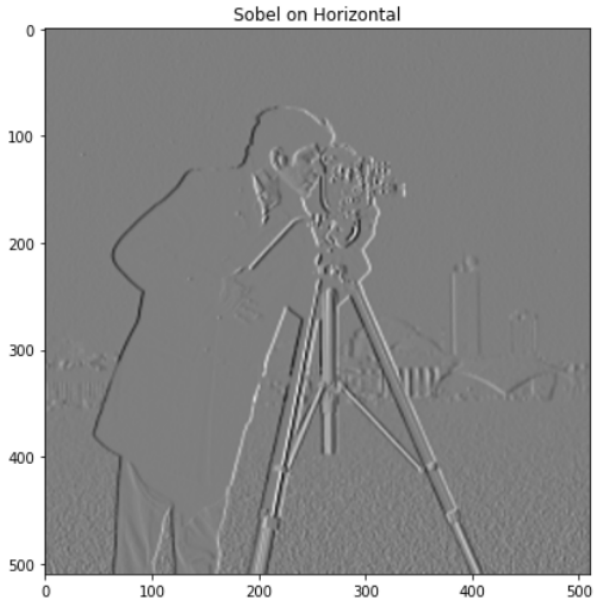
Brief theoretical:

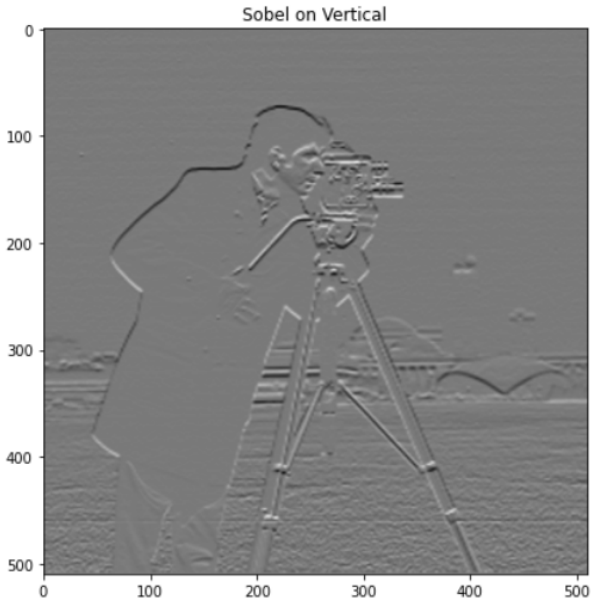
- Sobel sharpening work based on two filter array to horizontal and vertical, the following is taken from lecture slide of two array



Quality result:

- After applying sobel sharpening, we can see that image is now enhance in edge and more fine details
- We can see that the cameraman is highlighted from the background and it feels like that the sobel apply horizontally gives us more pop out image and sobel apply vertically give more pop in the object. However, this is just a personal perspective





II. Task 2

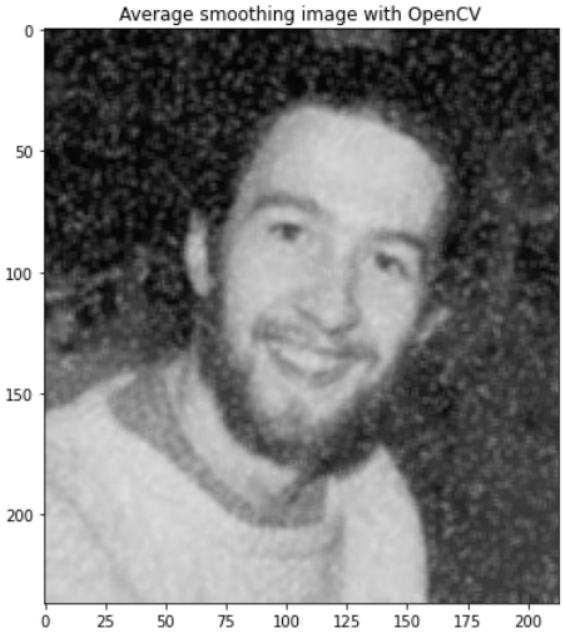
1. Sobel Sharpening Filter (filter size 3*3)

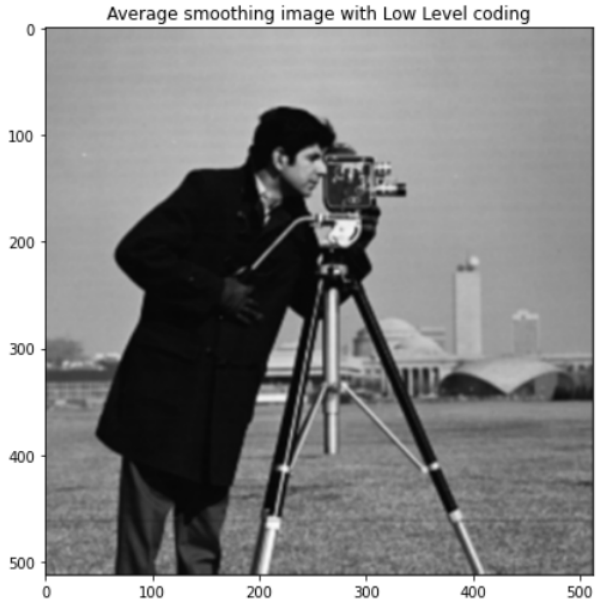
Brief theoretical:

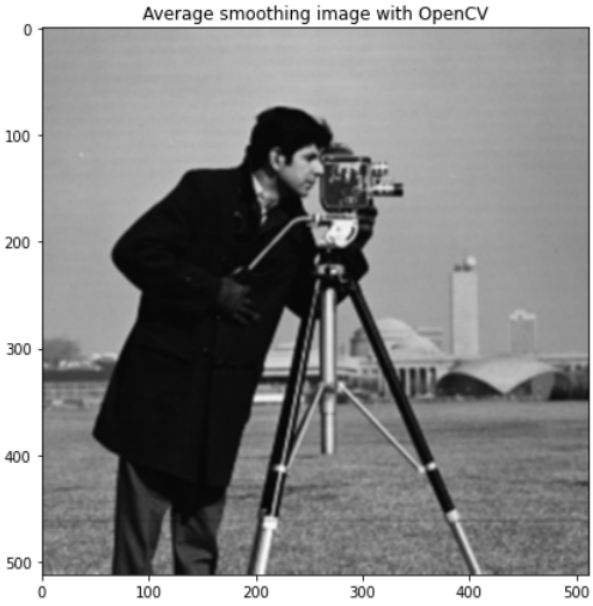
- The built in function from OpenCV is straightforward as the used is implemented and documented by OpenCV open source

Quality compare with question 1.a:

- The differences between OpenCV built in function and coding from scratch is really little that it needs a lot of focus to distinguish and it is not too recognizable
- However, I think OpenCV produce a bit smoother images, this is due to the fact that OpenCV have well-established function and optimized for many different image processing







2. Gaussian Smoothing filter with OpenCV

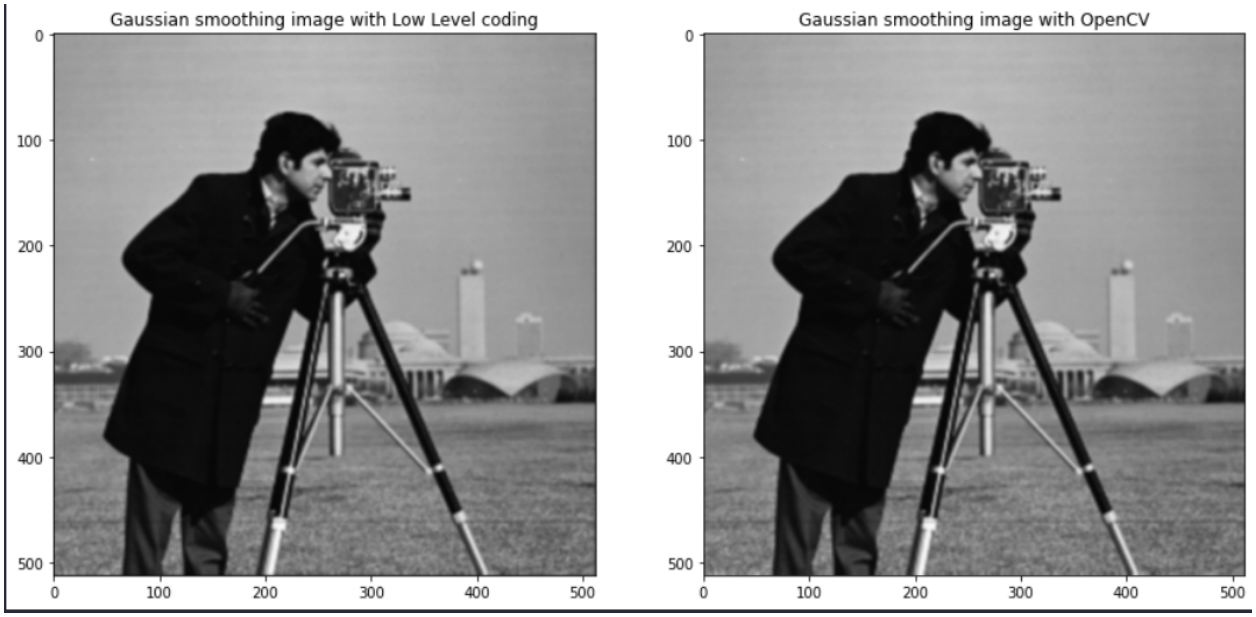
Brief theoretical:

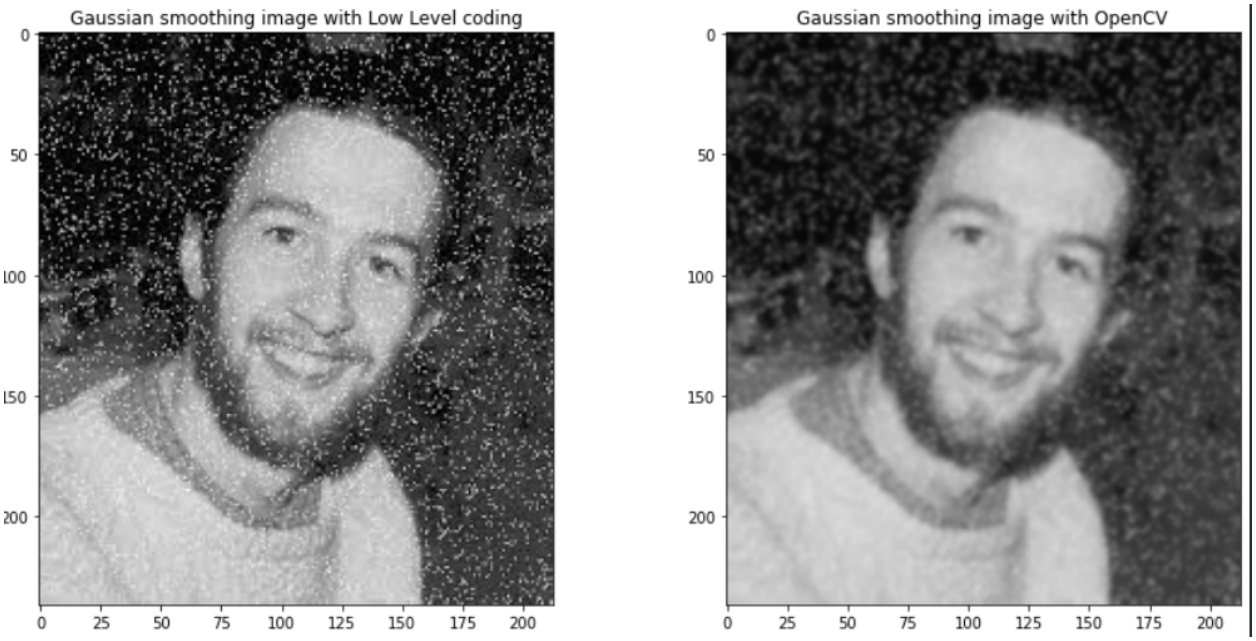
- The built in function from OpenCV is straightforward as the used is implemented and documented by OpenCV open source

Quality compare with question 1.b:

- For this task, I would not think the two method from OpenCV built in function and implemented from scratch would produce different quality image, as much as I can observed

- The fact that I think the images should not have too much of a difference is that the formula is the same for both built in function and coding from scratch





3. Sobel Sharpening filter with OpenCV

Brief theoretical:

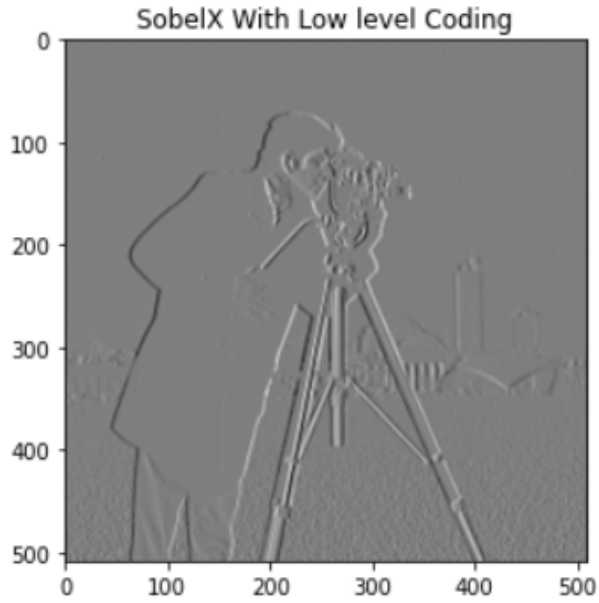
- The built in function from OpenCV is straightforward as the used is implemented and documented by OpenCV open source

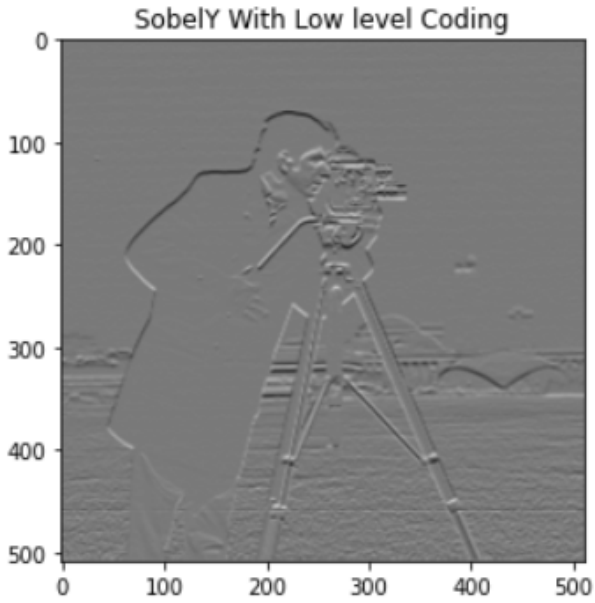
Quality compare with question 1.c:

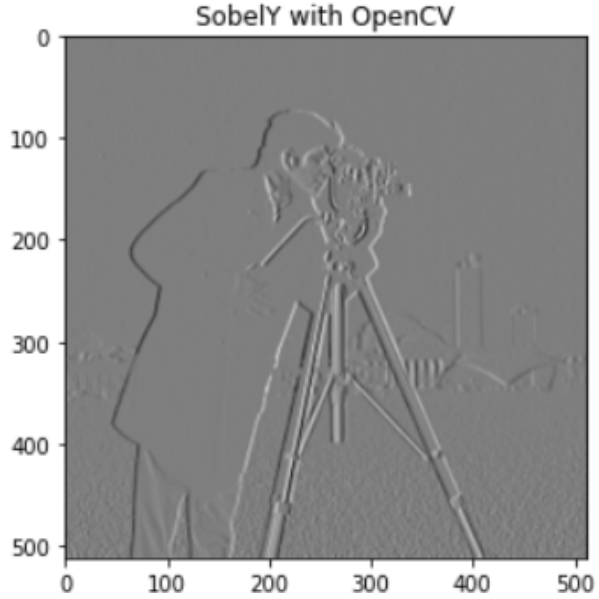
- For this task, I plotted the four images all together so we can observed them easily

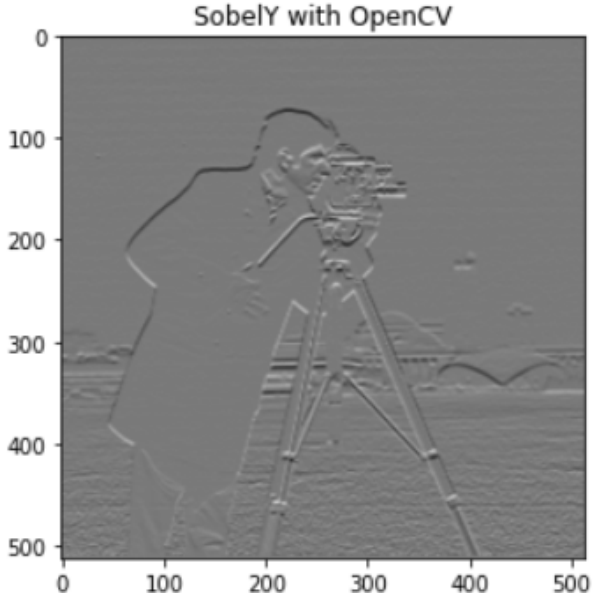
- After observation, I believe that two image should not have any differences, again, I think this result based on the fact that we have two filters that are already re-defined and we can change the kernel size if we wanted to

---









4. Overall observation

- In conclusion, I think that Task 1 and Task 2 does not have that much of a differences and the quality of the image may varies, however they are quite the same

III. Task 3

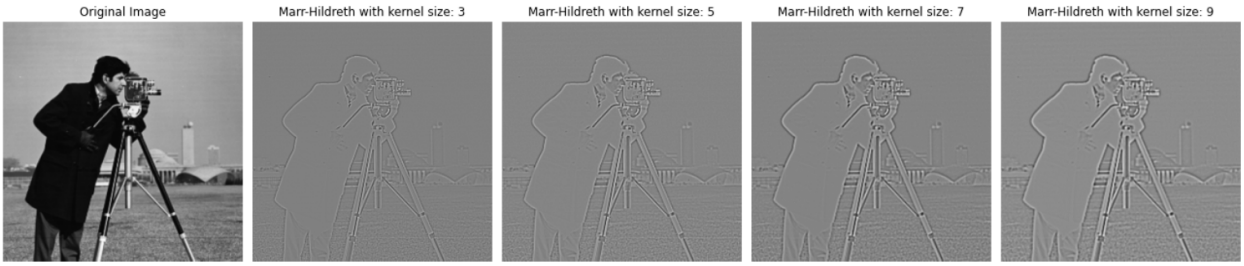
1. Marr-Hildreth

Brief theoretical:

- The built in function from OpenCV is straightforward as the used is implemented and documented by OpenCV open source
- The Gaussian Smoothing is followed by the computation of laplacian. Which allow the reduction of noise and still preserving the edges

Quality:

- For this question, the images generated from the built in function with different kernel size (3,5,7,9) is generated and we can clearly see the edge
- After running different kernel size, the result images seems to be more bright if we apply a higher kernel size
- The main factor for this is larger kernel size tends to average out dark objects or edges with varying intensity level



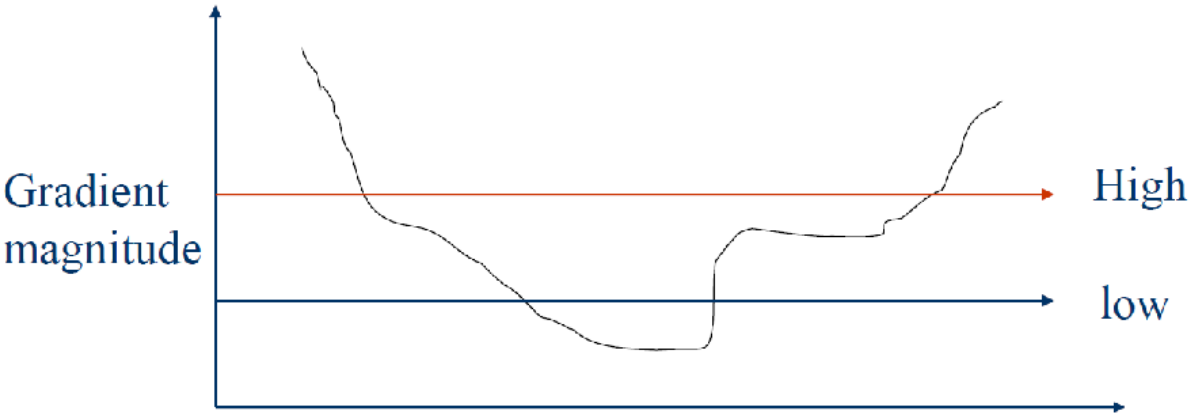
2. Canny Edge detector

Brief theoretical:

- The built in function from OpenCV is straightforward as the used is implemented and documented by OpenCV open source
- It used to detect edges boundaries within an image

Quality:

- For this question, the images generated from the built in function with different threshold value set, each of them have a different noise level
- For the threshold set of (100,200) they have the least noise around the camera man's legs. However, the building in the back is missing from the images
- Overall, threshold level 100,200 would give more detail in cameraman and less noise but it would not fully capture the background]
- The factor that play huge role in the differences is the high and low threshold, the week edge may include some edges that are some part of the background (like the images bellowed, from lecture's slide)



IV. Task 4

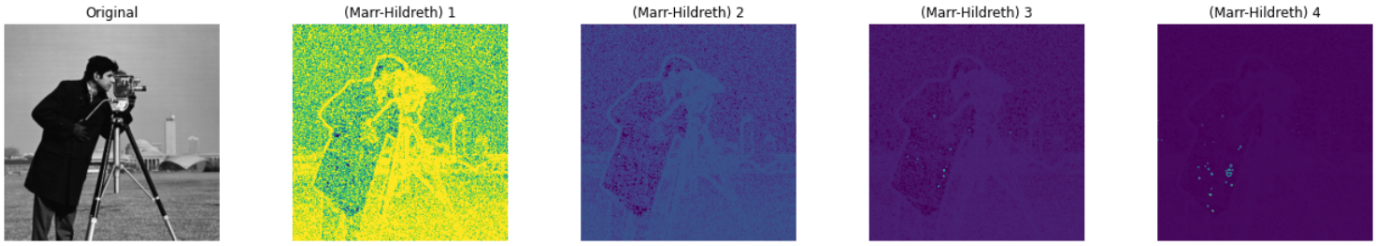
- For this task, I used the formula for 8-neighborhood, the reason for this is that the image we're working on might have small gaps between edges, therefore, for the purpose of bridging those gaps I would prefer 8-neighborhood
- Also, the algorithm is also mentioned in the slides (slide 5. Edge Detection)

$(x-1,y+1)$	$(x,y+1)$	$(x+1,y+1)$
$(x-1,y)$	(x,y)	$(x+1,y)$
$(x-1,y-1)$	$(x,y-1)$	$(x+1,y-1)$

8-neighbourhood

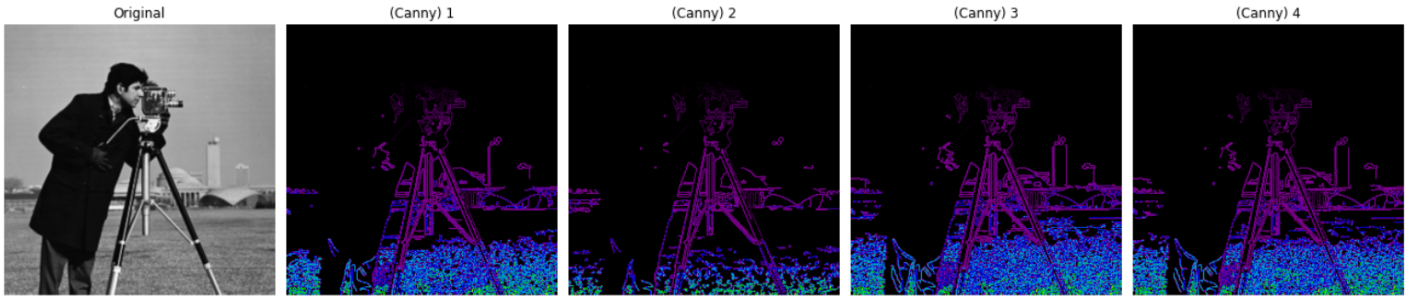
1. Marr-Hildreth group adjacent pixels

- Images seems to be clearer when we apply smaller kernel size to the image, moreover as we use large kernel size, the image tends to be a bit blurrier so in my opinion, that explains why image don't have any connected edges
- I would choose picture with kernel size 3, since it is most obvious to observe edges



2. Canny group adjacent pixels

- Images that apply group adjacent in canny seem to be more precise and shape better, however the cameraman head is missing from the images. I think this is because the pixels portions of the head is broken or week so that the connect method might not connect them effectively
- I would choose images with threshold set (100,200) because it would have the least noise and still have decent connected edges



IV. References

- Zia Ud Din. (n.d.). 5. Edge Detection [Review of 5. Edge Detection].