COMP9517 Computer Vision
Report of Assignment 1

Edited by Pinheng Chen (z5383372)

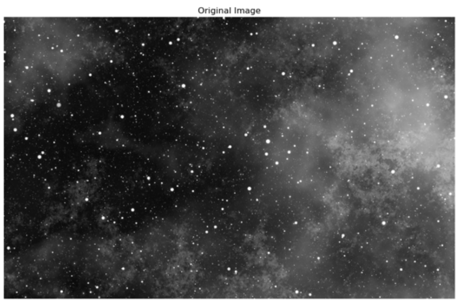
Environment:
Python(v3.9.13), Numpy(v1.23.1), Opencv(v4.6.0), Matplotlib(v3.5.2).

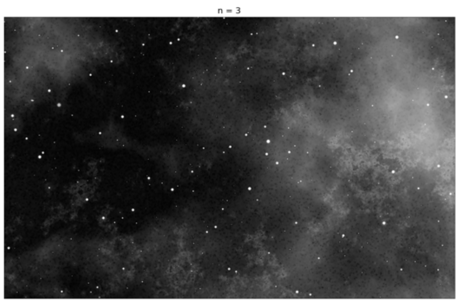
Goal:
The assignment is going to realize a function that is able to subtract the background of an image.

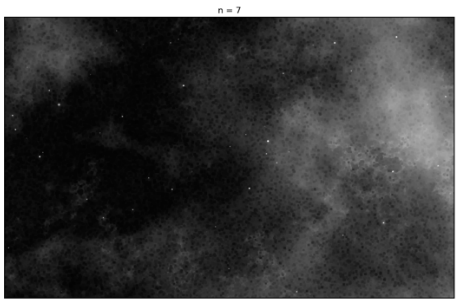
Approach:
I used Numpy, Opencv, and Matplotlib to realize the function in this assignment. Numpy is mainly used to manipulate arrays, Opencv is mainly used to manipulate images, and Matplotlib is mainly used to visualize images in jupyter notebook. The filters used in this assignment are minimum filter and maximum filter.

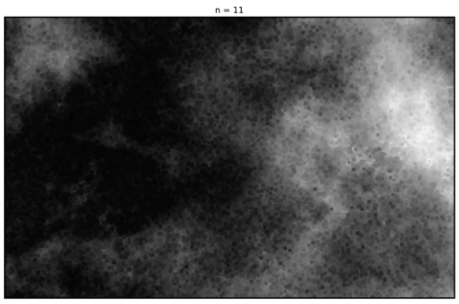
Steps:
Step 1: Use opencv read the original image I.
Step 2: Use min_filter (or max_filter) to get the background image A.
Step 3: Then use max_filter (or min_filter) to get the background estimate image B.
Step 4: Subtract image I with image B, we will get the background subtracted image O.
(I will explain this part later in more details)
Step 5: Output the result image O in a png format file.

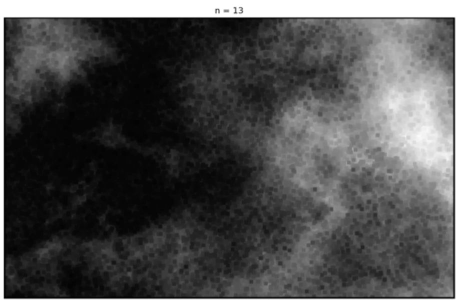
Task 1
Below is some values I have tried when choosing the value of N to generate image A. (see next page) Through these images (They can be found in the .zip file of the assignment), it is clear to see that N=17 is the smallest value of n that causes the stars in image I to visually disappear altogether in image A. As it is known to all that in gray scale, 0 represents dark, and 255 represents bright. And the min_filter is going to assign the minimum value in the kernel range to the center. So when appling the min_filter, if we chose the larger value of N, we are more likely to get an image that stars (extreme bright pixels) disappear altogether. When the choosing the values of n which are larger than 17, the number of dark pixels will increase, and these "outstanding" dark pixels will become the noise which will affect the quality of the result image. That is the reason why I chose 17 as N's value.

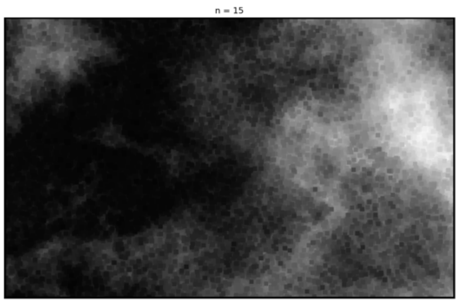


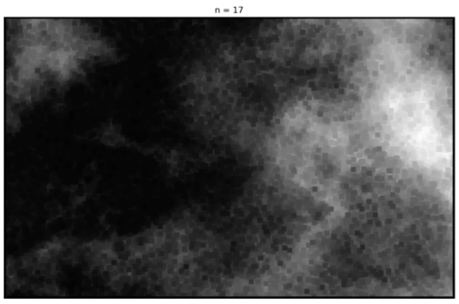


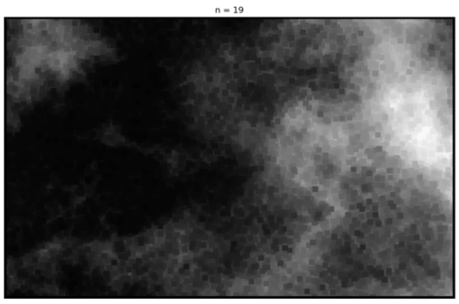




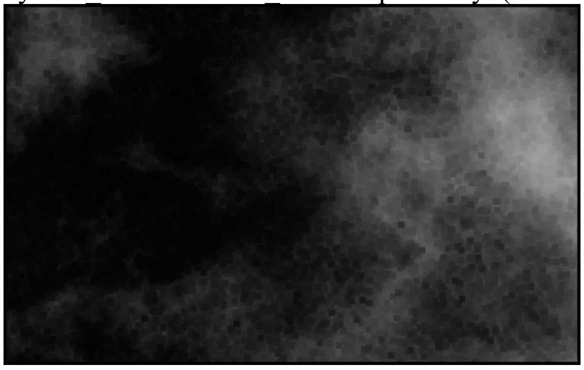


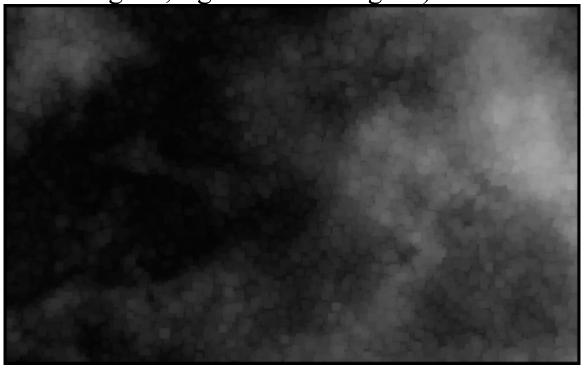






Then, I applied max_filter with n = 17. Below are the image A and image B which had been filtered by min_filter and max_filter respectively. (Left side is image A, right side is image B)





Task 2

Then, subtract image B from image I from pixel by pixel, we can get the following image O.

Comparing with the original image I, it is clear to see that the background has become darker and not so outstanding as pre-processed.

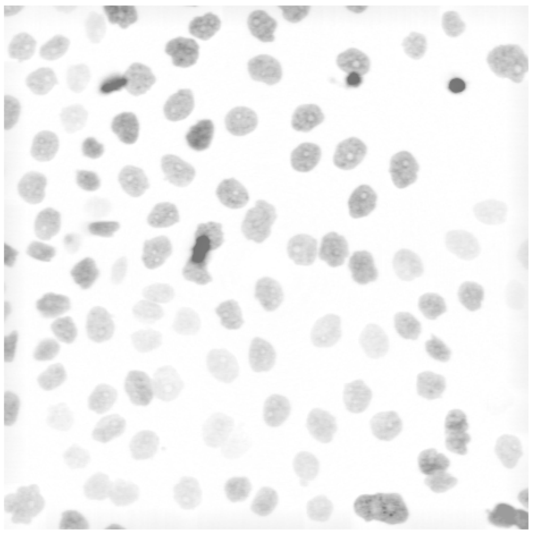
Task 3

In order to process more different images, some extensions should be added into my algorithm. Firstly, the size of kernel, named N (it is named n in my jupyter notebook), can be adjusted by user manually. And before processing image pixel by pixel in a loop, I used image.shape to get the numbers of rows and columns. Also, there is a parameter named M can determine the order of applying min-filtering and max-filtering operations. Further more, there is a parameter called color can determine the empty array (image) will be filled with 0 or 255.

As for the reason why using M=0 or M=1 for 2 different images, following is the explanation. At first, we know that 0 represents dark (black) and 255 represents bright (white) in gray scale image. For image I (Stars.png), the object is stars which is bright (value is close to 255) in the image and the background is dark (value is close to 0). In order to get the dark background image, we should eliminate stars visually through changing the larger value with smaller value (let white points become black). So we should use M=0 when processing image I. However, for image X (Nuclei.png), the object is cell nuclei which is dark (value is close to 0) in the image and the background is bright (value is close to 255). In order to get the bright background image, we should try our best to eliminate cell nuclei through changing the smaller value to larger value (let black blocks become white blocks). Hence, we should use M=1 when processing image X.

When M=1, the subtraction requires adding 255 because in the process of changing dark to bright, subtraction will generate negative values and these vegetive values will result in wrong output image. So, adding 255 is in order to prevent the value becomes negative after subtraction.

Following are background image B and output image O of Nuclei.png. (Left side is background image B, and right side is output image O)



These are background and result images that generated by different values of N (see next page). For background images, N verifies from 27 to 43, and for result images, N verifies from 27 to 131. It seems that when N is larger than 39, the result images will lose more details and the image scale will become very small, the objects in the margin area will be replaced with white padding. And the background images are still similar to the background image of N = 39. So I chose N = 39 eventually.