Assignment Report

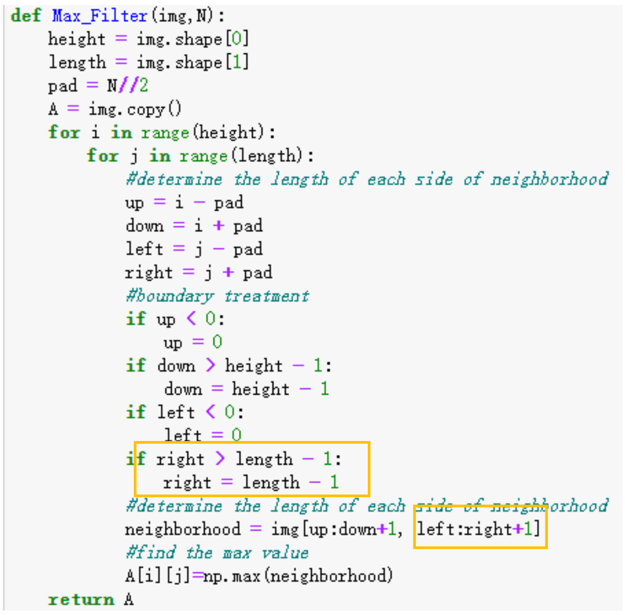
Guanzhu Hou z5325417

Task 1: Background Estimation

1. Explanation of the chosen implementation approach.

   In this task, we will make an accurate estimate of the background of the image by getting rid of the shading.

   As picture1-1 shows, firstly, we create a matrix A with the same height and length as the input particles.png. Then, we determine the length of each side of the neighborhood where the center of this N*N filter window is [i,j]. After that we consider the boundaries to prevent the area of the filter window from exceeding the input image. Finally, with the help of np.max(), we find the max value in the filter window.



```
def Max_Filter(img,N):
    height = img.shape[0]
    length = img.shape[1]
    pad = N//2
    A = img.copy()
    for i in range(height):
        for j in range(length):
            #determine the length of each side of neighborhood
            up = i - pad
            down = i + pad
            left = j - pad
            right = j + pad
            #boundary treatment
            if up < 0:
                up = 0
            if down > height - 1:
                down = height - 1
            if left < 0:
                left = 0
            if right > length - 1:
                right = length - 1
            #determine the length of each side of neighborhood
            neighborhood = img[up:down+1, left:right+1]
            #find the max value
            A[i][j]=np.max(neighborhood)
    return A
```

Picture 1-1 The code of Max_Filter

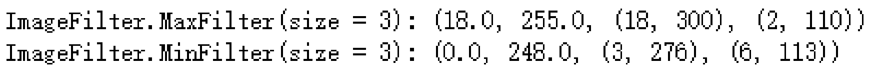
   By changing np.max() to np.min(), we get the function Min_Filter().

2. Discussion of intermediate results and experiences.

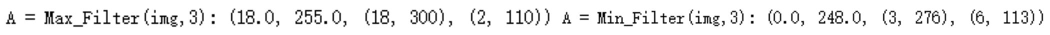
   Here, we take the right point as an example to introduce how we determine their values in code. By drawing a picture, we can directly recognize the right-hand maximum of the filter window is the length of the input image, which means in the matrix, the serial number of the right point cannot exceed the maximum value length-1.

   As it is given in Assignment Specification, N is an odd, so the length from the center to the left and the right are the same. And the length from the right to the left in the code is “left:right+1”, which is (right+1)-left= (i+pad+1)-(i-pad) = 2pad+1=N.

   After comparing the result of Min_Filter and Max_Filter with ImageFilter in pictures 1-2, we find the global minimum and maximum are the same, which determined that the filters can handle the image correctly.



```
ImageFilter.MaxFilter(size = 3): (18.0, 255.0, (18, 300), (2, 110))
ImageFilter.MinFilter(size = 3): (0.0, 248.0, (3, 276), (6, 113))
```



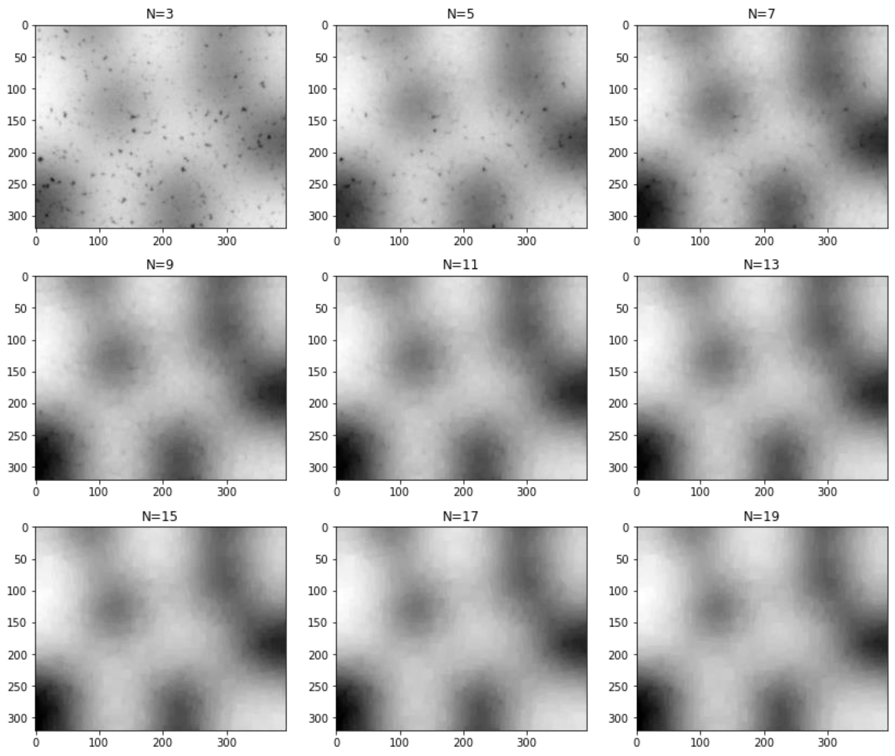
```
A = Max_Filter(img,3): (18.0, 255.0, (18, 300), (2, 110)) A = Min_Filter(img,3): (0.0, 248.0, (3, 276), (6, 113))
```

Pictures 1-2 Checking the correctness with ImageFilter

3. Discussion of the best value for parameter N.

1). What is the smallest value of N that visually causes the dark particles in I to disappear altogether in image A?

Plotting the experiment results from N=3 to N=19 in picture 1-3, the smallest value N is 15.



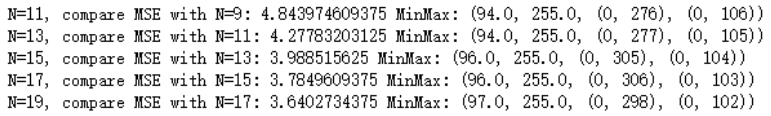
Picture 1-3 Results of Max_Filter from N=3 to N=19

2). Why this value of N, and not smaller values, causes the particles to disappear?

From the picture 1-3, there are still dark particles after filtering with smaller windows. What's more, max filtering forces points with distinct intensities to be the maximum(brightest) among their neighborhoods, thus eliminating isolated intensity spikes. That is too small a window cannot effectively get a clean background, so we need to choose a relatively big window size N.

3). What is the effect of taking larger values than this smallest value?

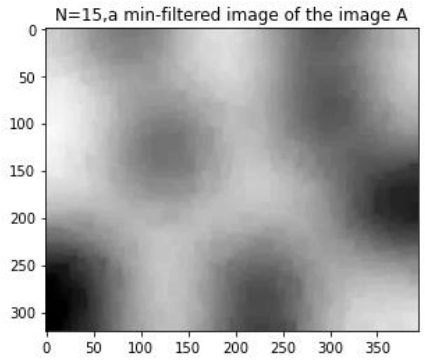
From the picture 1-3, there is almost no difference after adding N. From the point of view of the Mean Square Error (MSE) in the picture 1-4, it decreases more slowly after N=15, which means it is more similar to the previous one after N=15.



```
N=11, compare MSE with N=9: 4.843974609375 MinMax: (94.0, 255.0, (0, 276), (0, 106))
N=13, compare MSE with N=11: 4.27783203125 MinMax: (94.0, 255.0, (0, 277), (0, 105))
N=15, compare MSE with N=13: 3.988515625 MinMax: (96.0, 255.0, (0, 305), (0, 104))
N=17, compare MSE with N=15: 3.7849609375 MinMax: (96.0, 255.0, (0, 306), (0, 103))
N=19, compare MSE with N=17: 3.6402734375 MinMax: (97.0, 255.0, (0, 298), (0, 102))
```

Picture 1-4 MSE as well as the Min&Max values from N=11 to N=19

4) Background estimate image B included



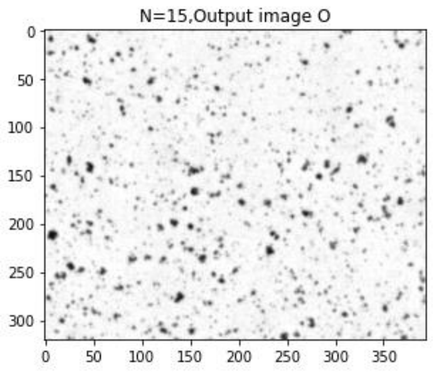
Picture 1-5 image B N=15

Task 2: Background Subtraction

1) Explain in your report what is theoretically the most negative value we can expect when subtracting two 8-bit/pixel images.

Min-Max=0-255=-255

2) Output image O included

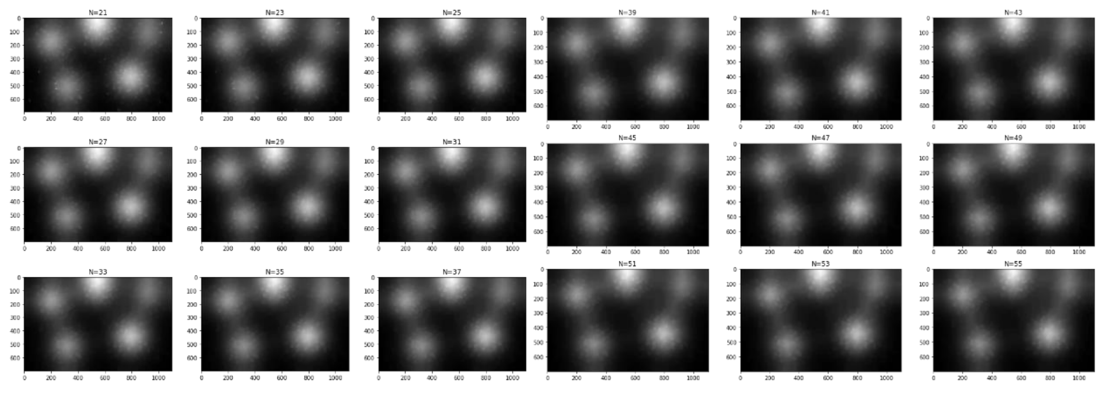


Picture 2-1 Output image N=15

Task 3: Algorithm Extensions

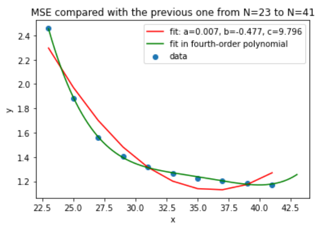
1) Discussion of the best value for parameter N

To find the good value of N, first apply Min_Filter only like what we have done in Task 1, and take the smallest value which can remove all the cells as the good value.



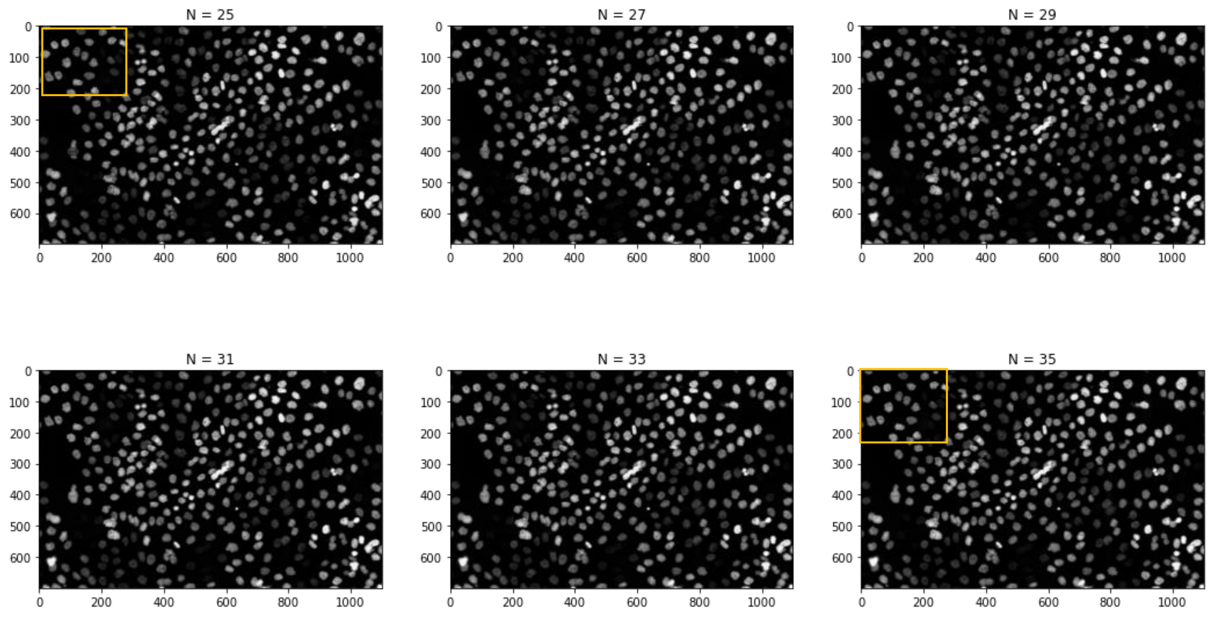
Pictures 3-1 Results of Min_Filter from N=21 to N=55

Observation of pictures 3-1 by naked eyes alone shows that there is almost no difference in the results after N=31. Then from N=23 to N=41, calculate the MSE value between them and the previous Min_Filter result, and fit the curve. The red curve is a quadratic polynomial, and the curve rises after N>37, which is not very accurate. Green is a fourth-degree polynomial, which shows that the data are convergent, though not robust enough. In short, take N equal or bigger than 27, for the curves decrease more slowly after N=27 and the larger window area does not make it much different from the previous window area.



Picture 3-2 MSE compared with the previous one from N=23 to N=41

Another way to find a good value is to analyze the final results after subtracting the background image from the original.



Picture 3-3 final results from N=23 to N=41

Compared with the original cells.png, we could distinguish light-colored objects in dark backgrounds since N=19 or maybe smaller N. But take the cells in the orange box as an example, we could observe the cell morphology more clearly instead of the dark dots with larger N, which is undoubtedly more conducive to scientific work. At the same time, in the central area of the picture, due to the dense number of cells, it is not difficult to distinguish the number of cells with larger N, and it is convenient for counting the total number of the cells. In summary, although we have met the requirement of removing background shadows with smaller N, a larger N is used to facilitate better use of this picture in practice.

Through these two experiments, we choose N=31, of which MSE value drops relatively slowly, and the central area is clearer.

2) Explanation of the algorithm extension approach

In the main code, we need to enter the values of M and N on the keyboard. If we enter M=0, then the code begins to process particles.png and M=1 for cells.png.

For particles.png, we recommend N = 15 and the code will execute Max_Filter, Min_Filter and then Subtraction with correction. Finally, it will display and save the process picture B and the output picture O.

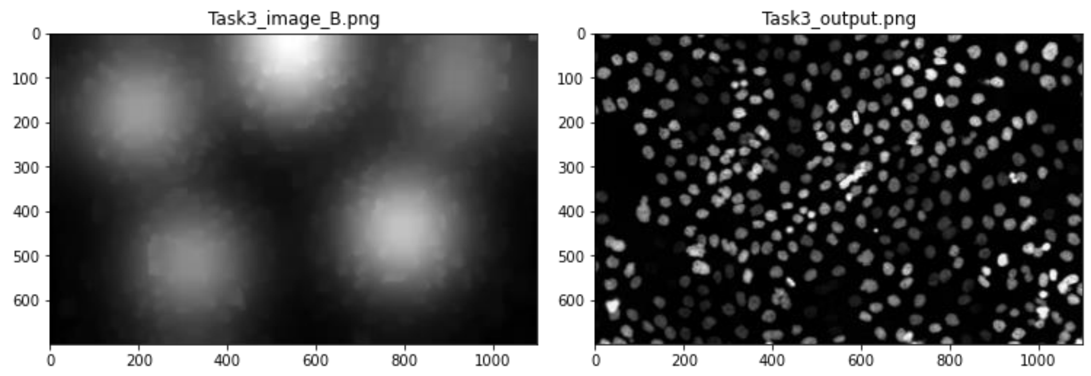
For cells.png, we recommend N = 31 and the code will execute Min_Filter, Max_Filter and then Subtraction without correction. Finally, it will display and save the process picture B and the output picture O.

3) Explanation why M = 0 or M = 1 for the two different images

For particles.png when M=0, the objects are dark while the background is bright. So if we want to extract its background, we will use Max_Filter first and get image A, which will increase the value of the dark objects to their brightest neighbor and make these dark objects disappear. Then because of the higher gray values in A than the actual background values in particles.png, we apply Min_Filter to make it closer to the actual background.

For cells.png when M=1, the objects are bright while the background is dark. So if we want to extract its background, we will use Min_Filter first and get image A, which will decrease the value of the bright objects to their darkest neighbor and make these bright objects disappear. Then because the lower gray values in A than the actual background values in cells.png, we apply Max_Filter to make it closer to the actual background.s

4) Background image B and output image O included



Picture 3-4 Background image B and output image O