# COMP9517: Computer Vision
# 2022 Term 2
# Assignment Report

Yifei Yue z5392319

**Task1**

**Goal and approach**

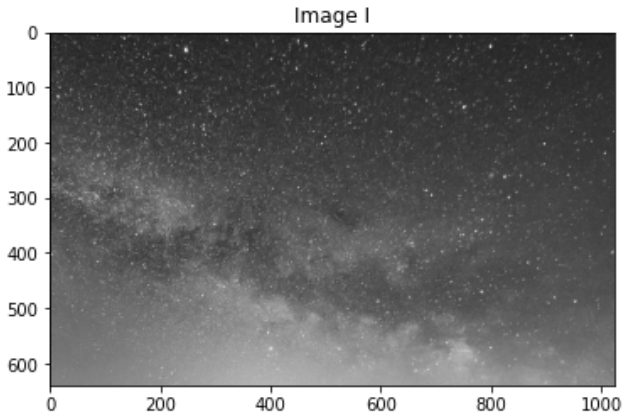
In task 1, for a given image, our goal is to obtain an estimated background image of the given image through a series of image processing methods. So that the objects in the original image can be obtained later by subtracting the estimated background image from the original image.

**Our approach has four steps:**

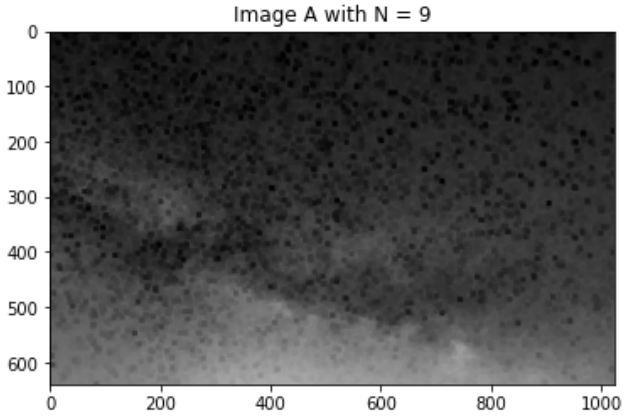
1. Create an image A of the same size as the input image I.
2. In the given Milkyway image, the color of the objects is bright, and the value of object pixels are larger. The color of the background is dark, and the value of background pixels are smaller. Since the pixels that make up the objects have higher pixel values, then we can firstly use min-filtering so that the objects can be removed from the original image by updating the value of each pixel to the minimum gray value in a neighborhood centered around that pixel. The image A obtained is the background image with the objects removed from the original image.
3. Create an image B of the same size as image A
4. To enhance the shading details in the obtained image A, as we just update the value of each pixel to the minimum gray value in a neighborhood centered around that pixel. Then we use max-filtering to update the value of each pixel to the maximum gray value in a neighborhood centered around that pixel. The image B obtained is the background image with shading details enhanced from the initial background image A.

**Intermediate steps:**

We first read the Milkyway.png as image I, get the shape of it and create an Image A with the same shape. Each pixel is represented by 0 initially. The original image I is shown below:

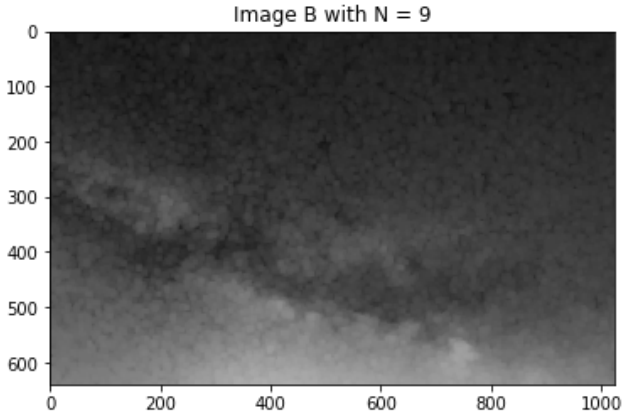


Then we do min-filtering on image I, due to the neighborhood is a square of N x N pixels, we can pad the edge of image I with a circle of pixels with a width of N // 2, so that each edge pixel in the image can be the center pixel of its neighborhood. The pixel value of the pixels we padded is 255, because we choose the minimum gray value in a neighborhood centered around that pixel. Therefore, the padded pixel values of the boundary will not affect the update of edge pixels in image I. When each pixel of image I is updated, we obtain the image A. The resulting image A which is the min-filtered image of the image I is shown below

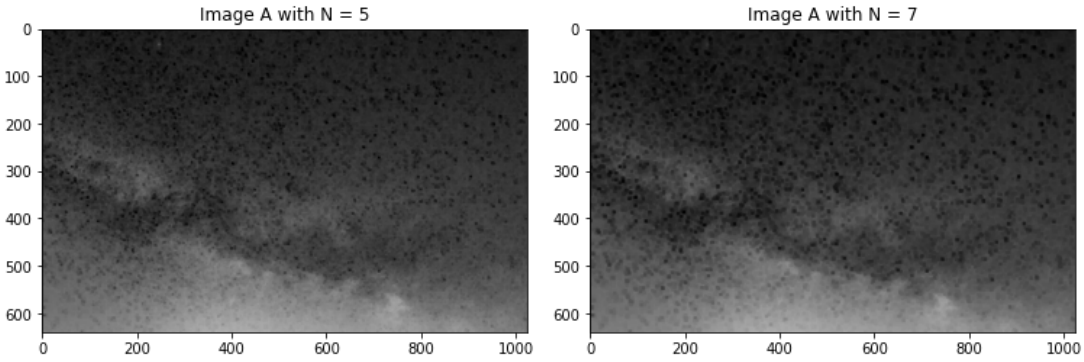



Then, we get the shape of the image A and create an Image B with the same shape. Each pixel is represented by 0 initially.

At last, Image A is padded in the same way. Since we need to find the maximum gray value of each pixel in a neighborhood of the save size centered around each pixel, when we pad 0 around the boundary of image A, these padded pixels will not affect the update of the edge pixels in the image. When each pixel of image A is updated, we obtain the image B. The resulting image B which is the max-filtered image of the image A is shown below.

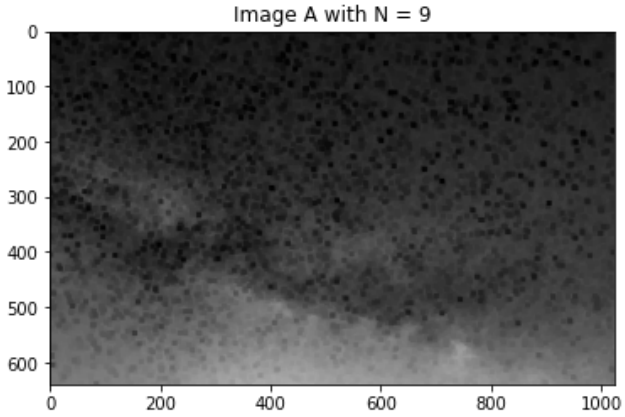



9 is the best value of parameter N in this task. We tried N from 3 to 11 in turn, and found that when N is 7, there is still a bright spot or a star in the upper left corner of the image, and when N is 9, all the stars in the image disappear, so the minimum value of N is 9.



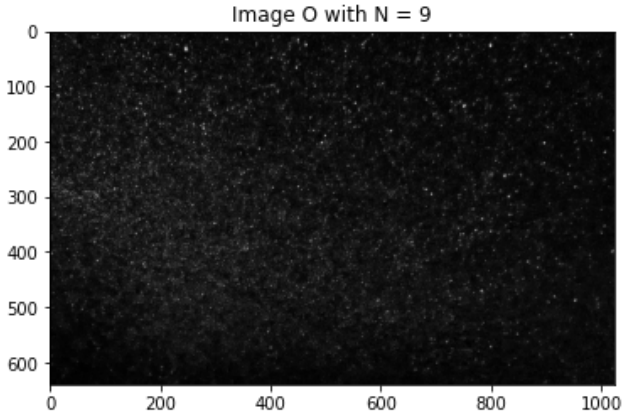

**Discussion of the best value for parameter N**

When N is less than 9, the square of N x N is likely to be smaller than the largest star in the image. Therefore, during the min-filtering, the pixels in the neighborhood we compared could be all inside the largest star, so the new pixel values obtained by min-filtering are still large, so the star is not removed. When selecting a larger value of N, it can ensure that more background pixels are considered while processing the objects, so the selected center pixel is more likely to be replaced by the pixel value in the background. Because the value of the smallest N that can make all the stars disappear is 9, then we think 9 is the best value of N.




**Task 2**

Therefore, we can compute the output image O from Image I and Image B, which is shown below.



**Task3**

**The implemented algorithm extensions:**

To enable the whole task to process input images of any size, and allow the user to set the size N of neighborhood and choose the order of the min-filtering and max-filtering operations by M. We define the following four functions, min_filtered, max_filtered, bgm_sub and get_objects. min_filtered and max_filtered have two input parameters which are img and N, which requires an input image of arbitrary size and N which is the size of neighborhood we will use, they return a image after doing the corresponding filtering. bgm_sub has three input parameters which are img1, img2 and M, where img1 should be the image I which is the input image, and img2 should be the image B which is the background image, and M is the order of the min-filtering and max-filtering operations which will be explained below. It returns an image which is subtracting img2 from img1 pixel by pixel. We use get_objects function which calls the previous three functions to do the whole task. It requires an image of arbitrary size and N which is the size of neighborhood we will use, and M which is the order of the min-filtering and max-filtering operations, and it returns a image of objects which is the given image with background removed.

**Why M = 0 and M = 1 for the two different images?**

For such images with dark background and bright objects, we set M to 0, because we need to do min-filtering first, to remove all bright objects, and then do max-filtering to enhance the shading details in the dark background.

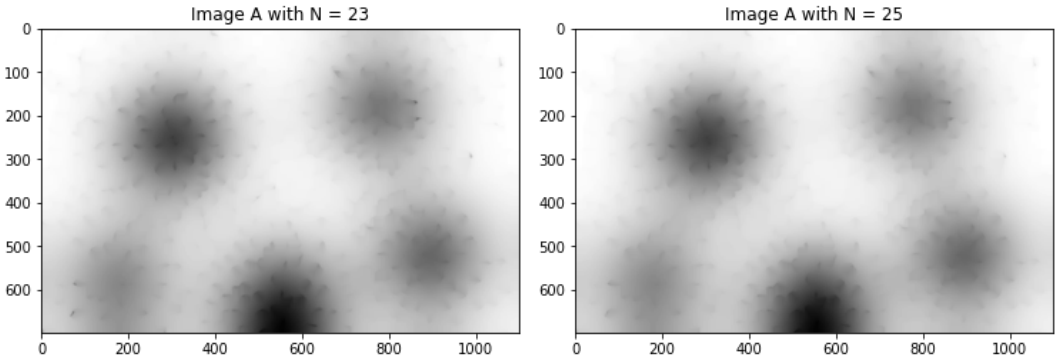
For such images with bright backgrounds and dark objects, we set M to 1, because we need to do max-filtering first, to remove all dark objects, and then do min-filtering to enhance the shading details in the bright background.

**Why for M = 1 the subtraction requires adding 255?**

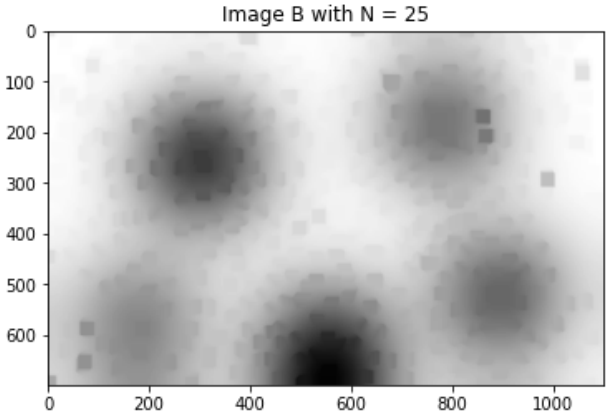
When M is 1, after the max-filtering is completed first, most of the pixels in image A are more likely to have pixel values close to 255. Then, most of the pixels in image B obtained after doing min-filtering on image A are still more likely to have pixel values close to 255. Therefore, when calculating I - B, it is likely that many pixels in image B have pixel values that are greater than those in the same position in Image I. Thus, 255 is added to make the white background still white and avoid the errors caused by the rollback of negative numbers.

**Discussion of the best value for parameter N**

25 is the best value of N. Following the save steps we choose the best value of N in task 1, we can find that when N is less than 25, there are still some cells in the image A, when N is equal to or larger than 25, all cells are approximately disappeared altogether visually in image A. Therefore, we think 25 is the best value of N.




The background image B which is the min-filtered image of the image A is obtained and shown below.




Therefore, we can compute the output image O from Image I and Image B, which is shown below.