“Computer Vision”

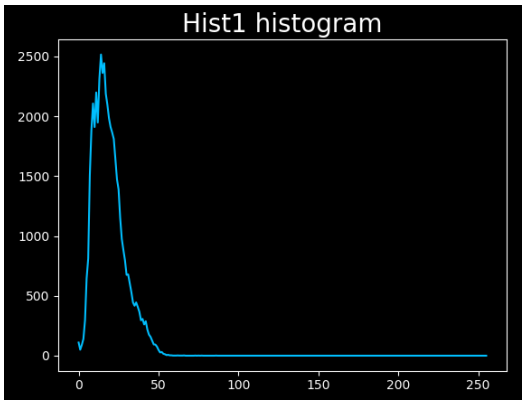
Shervin Halat

98131018

Homework 1

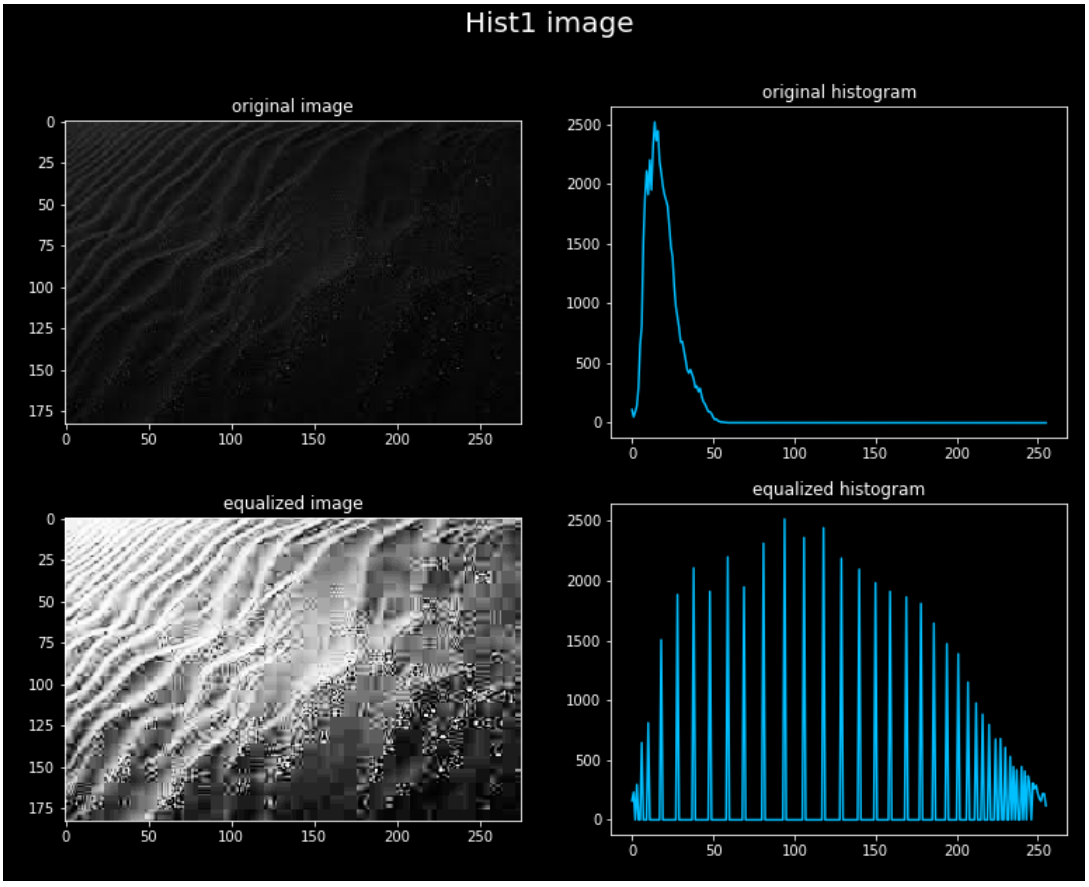
1.

First image:

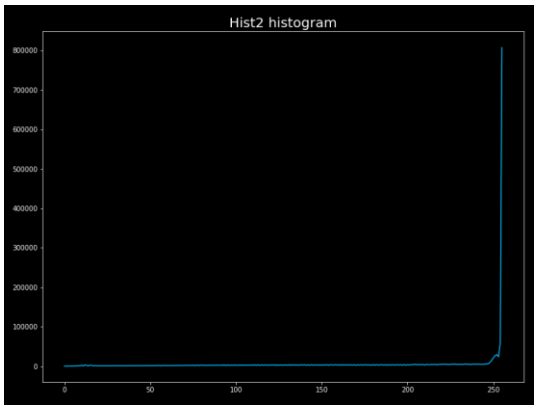


Regarding the first plot (Hist1 histogram), the first image is not appropriate for processing owing to the fact that almost all of its pixels are concentrated in low side of the gray scale between 0 and 50, and it has significantly darkened the image and vanished its details. Therefore, a histogram equalization approach is needed to distribute the histogram to a wider range in order to adjust the current image low contrast.

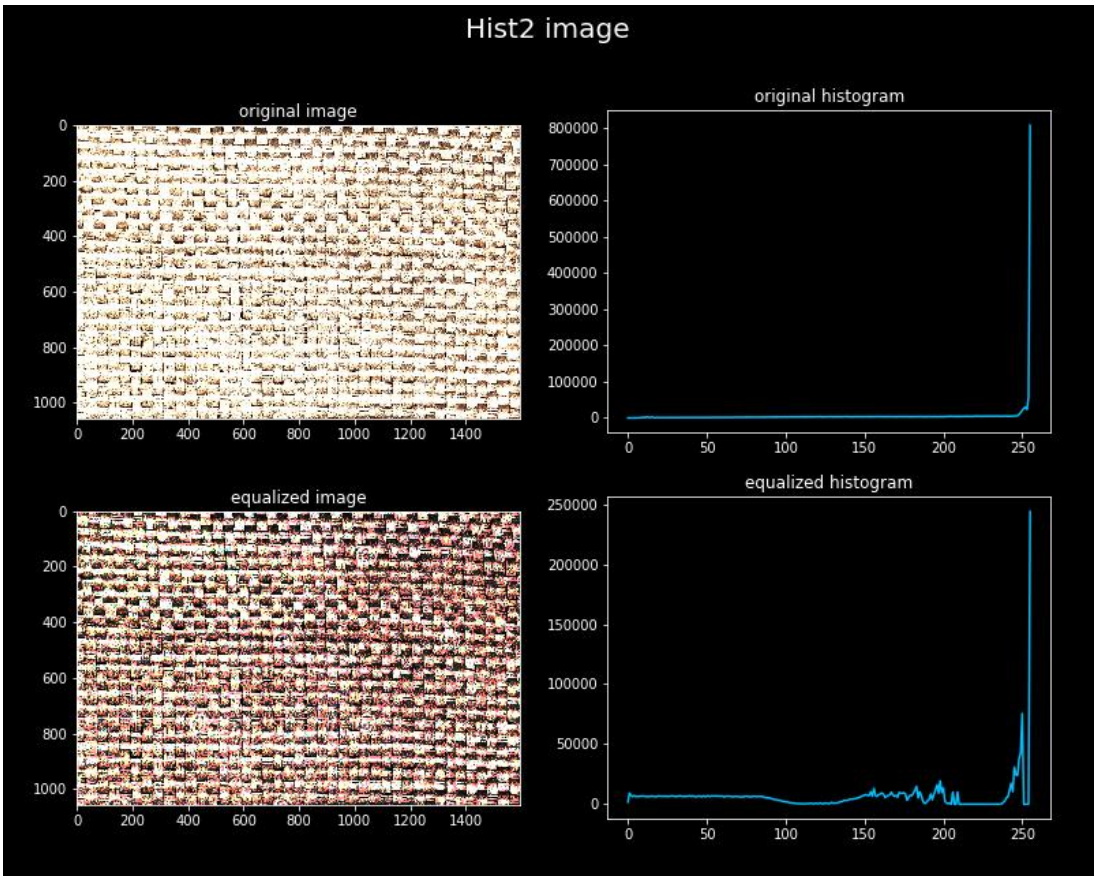
By applying histogram equalization, following results will be obtained:



Second image:

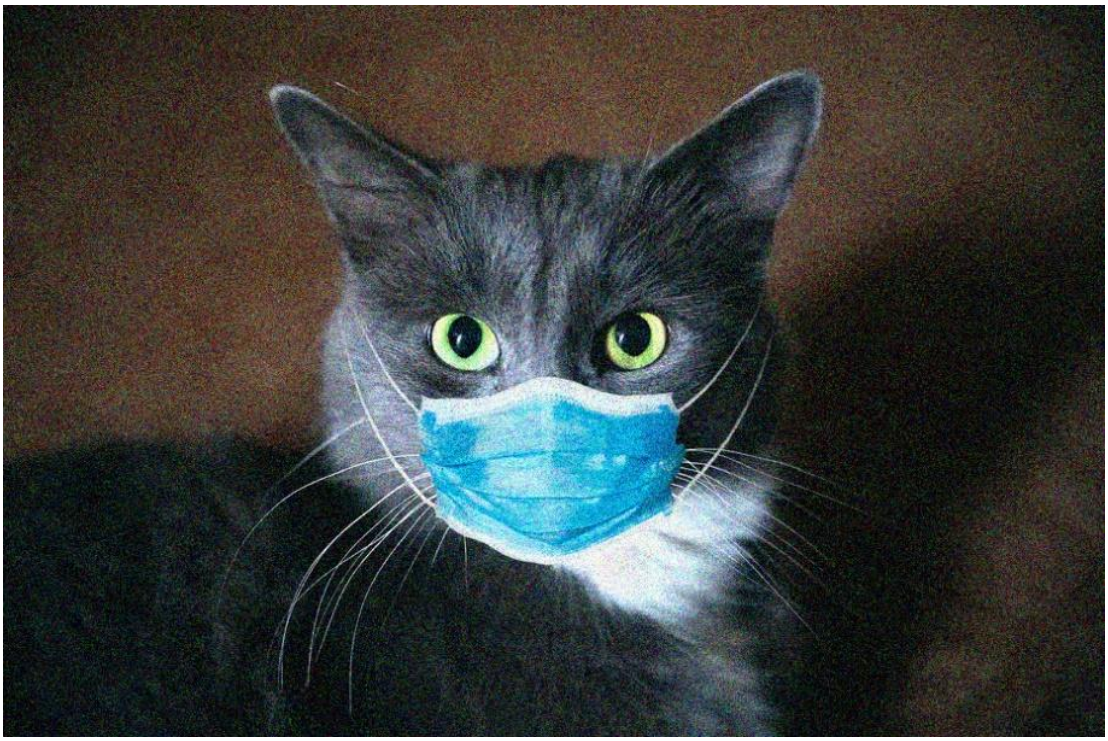


Regarding the plot above (Hist2 histogram), the second image is also not appropriate for processing owing to the fact that almost all of its pixels are concentrated in high side of the gray scale above 220, and it has significantly brightened the image. Therefore, a histogram equalization approach is needed to distribute the histogram to a wider range in order to adjust the current image low contrast. To do the mentioned approach, since the second image is a colored image, as against the first image, histogram equalization can be done in various ways. The more usual one is to apply equalization on all channels of RGB color space of the image then merge them again. This approach is applied here and the results are as follows:



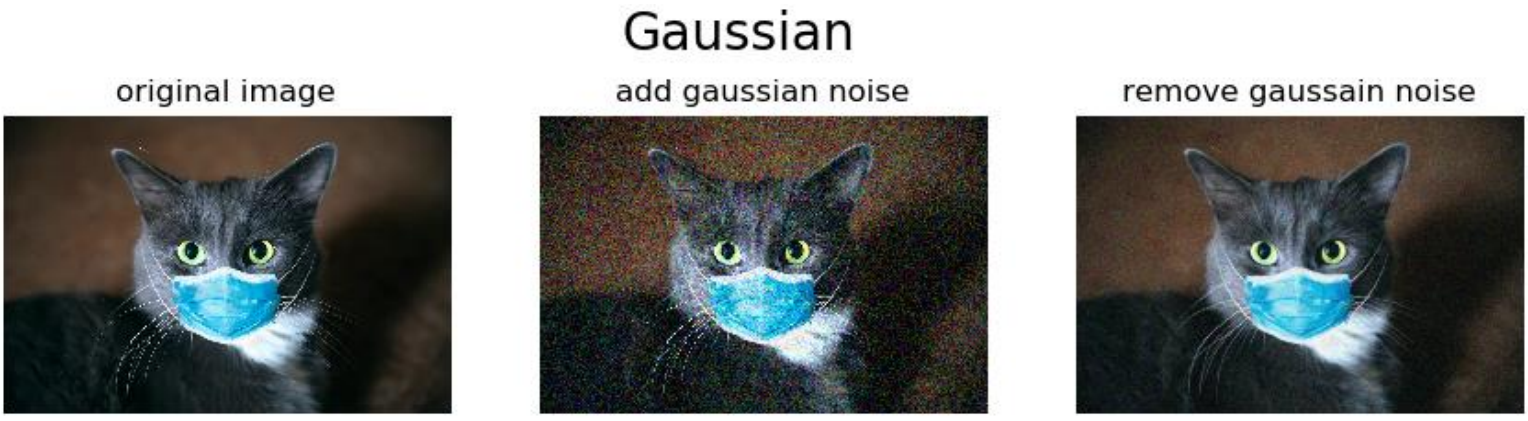
2.

Gaussian:



To remove gaussian noise, gaussian smoothing kernel of size 5x5 has been used. Since gaussian noise with zero mean was added to the image, a symmetric kernel (such as gaussian or averaging kernel) had to be used. Also, gaussian kernel was preferred since it preserves edges better.

Obtained results are as follows:



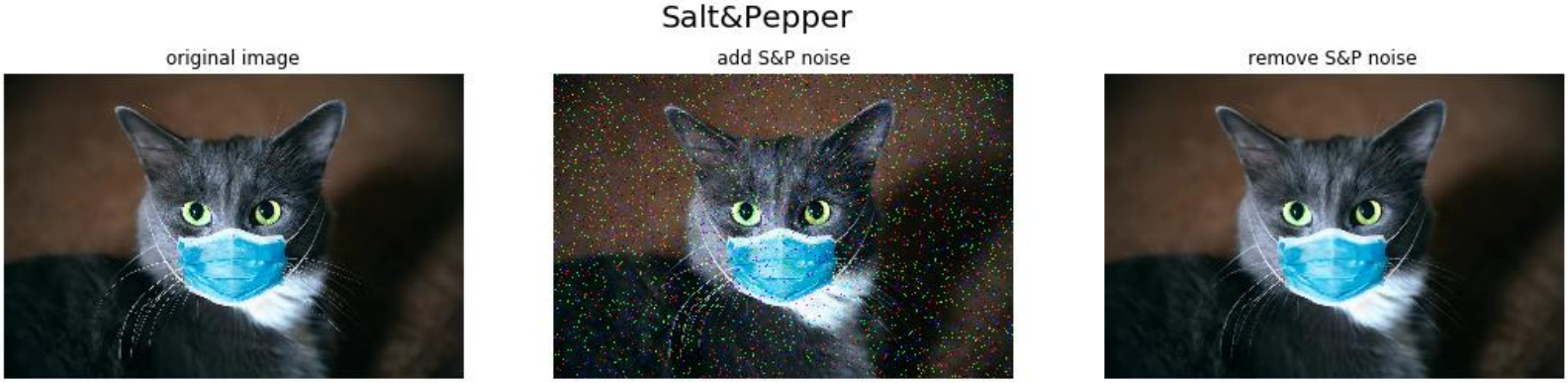
Uniform:

Just similar to the gaussian noise, gaussian smoothing kernel was used for uniform noise as well, and also for the same reason.

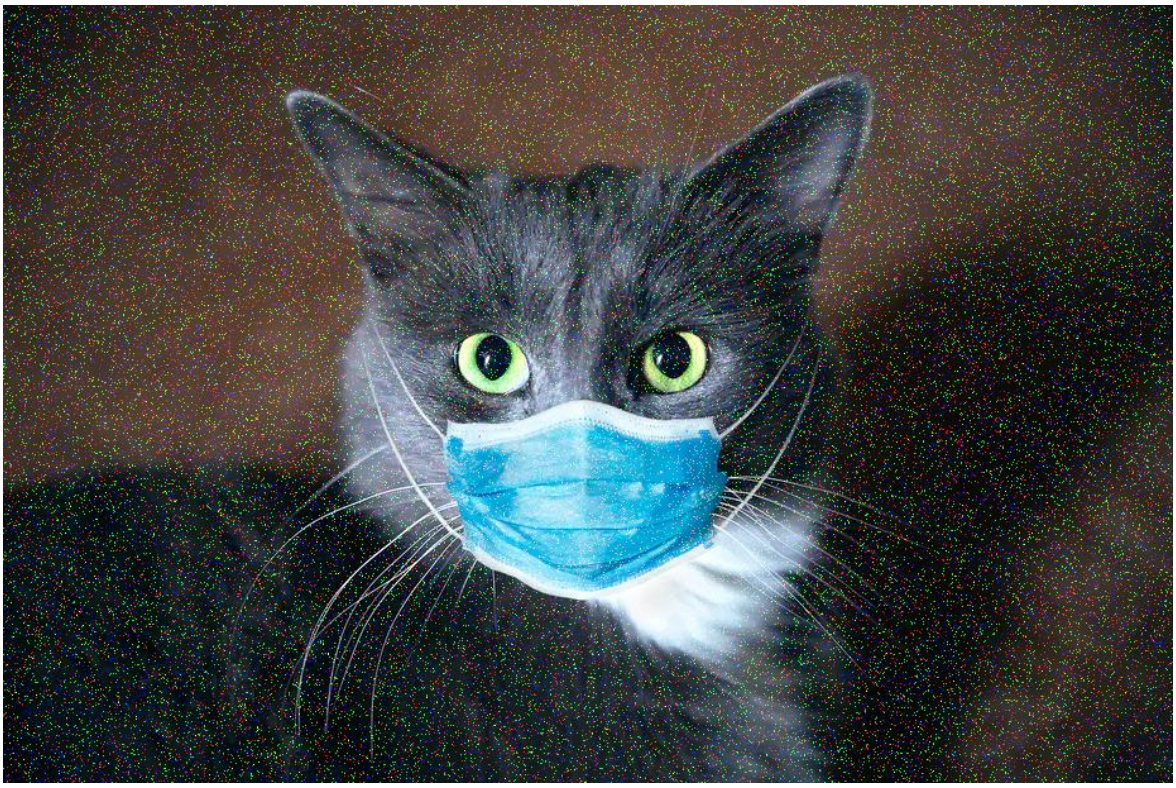
Results obtained are as follows:

Salt & Pepper:

To remove salt & pepper noise, median filter works well since this kind of noise (as well as impulse noise) has values of 0 or 255; therefore, by applying median filter these noises will be placed at the beginning or end of the sorted list of kernel values and will not be chosen; hence, these noises will be eliminated from the image.

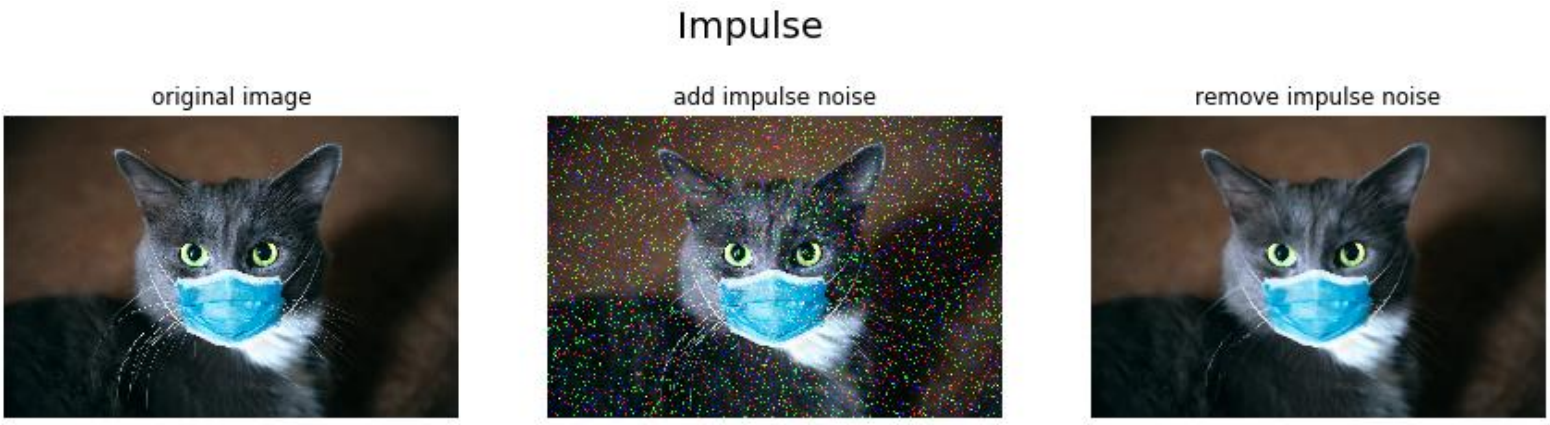


Impulse:



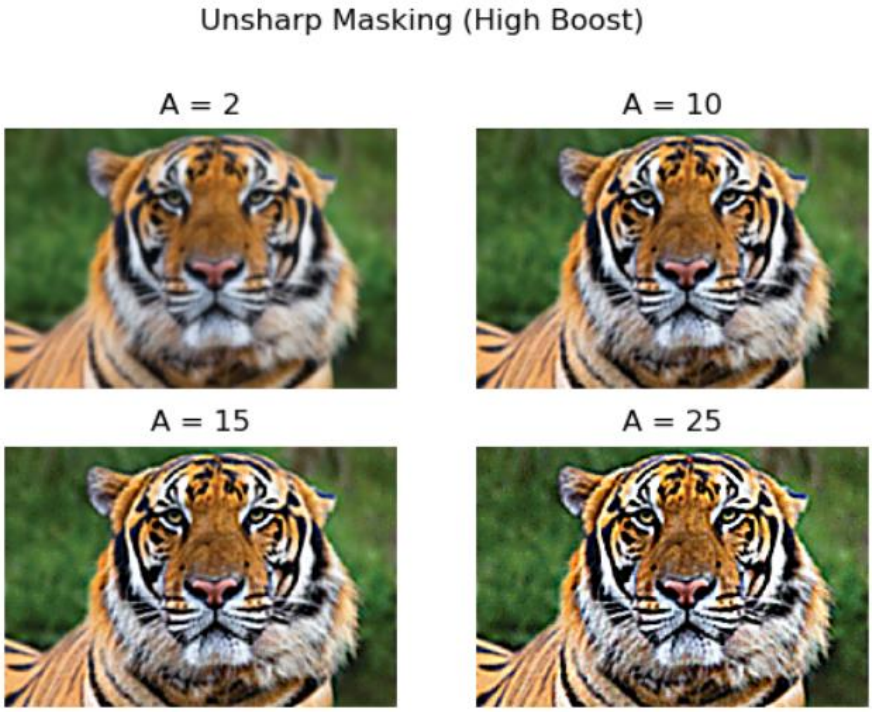
Just similar to the S&P noise, median kernel was used for impulse noise as well, and also for the same reason.

Results obtained are as follows:



3.

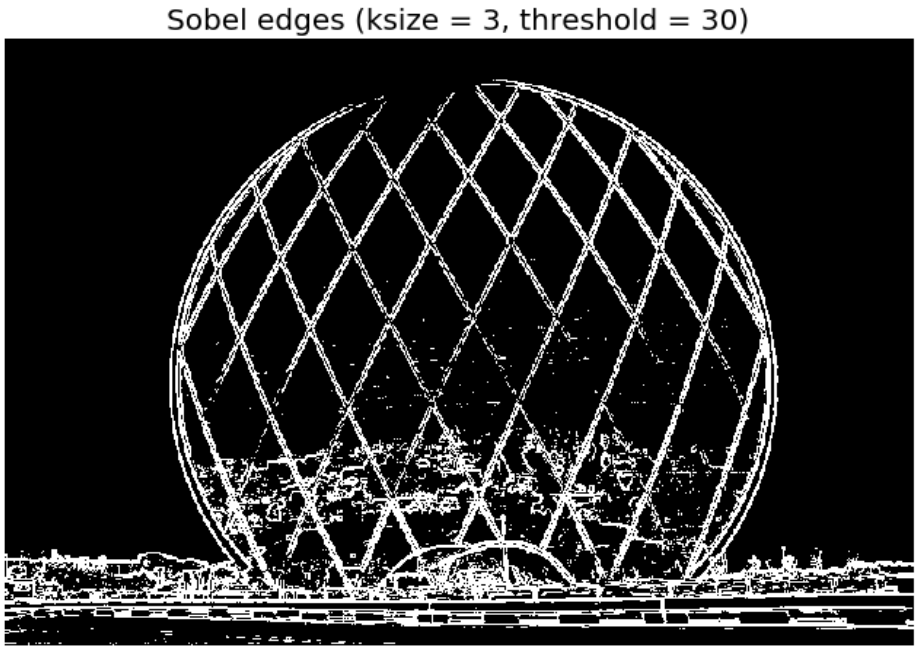
“A” value of 2, 10, 15, and 25 were considered:



4.

Sobel:

Detected edges by Sobel operation (with kernel size of 3 and threshold of 30) are as follows:



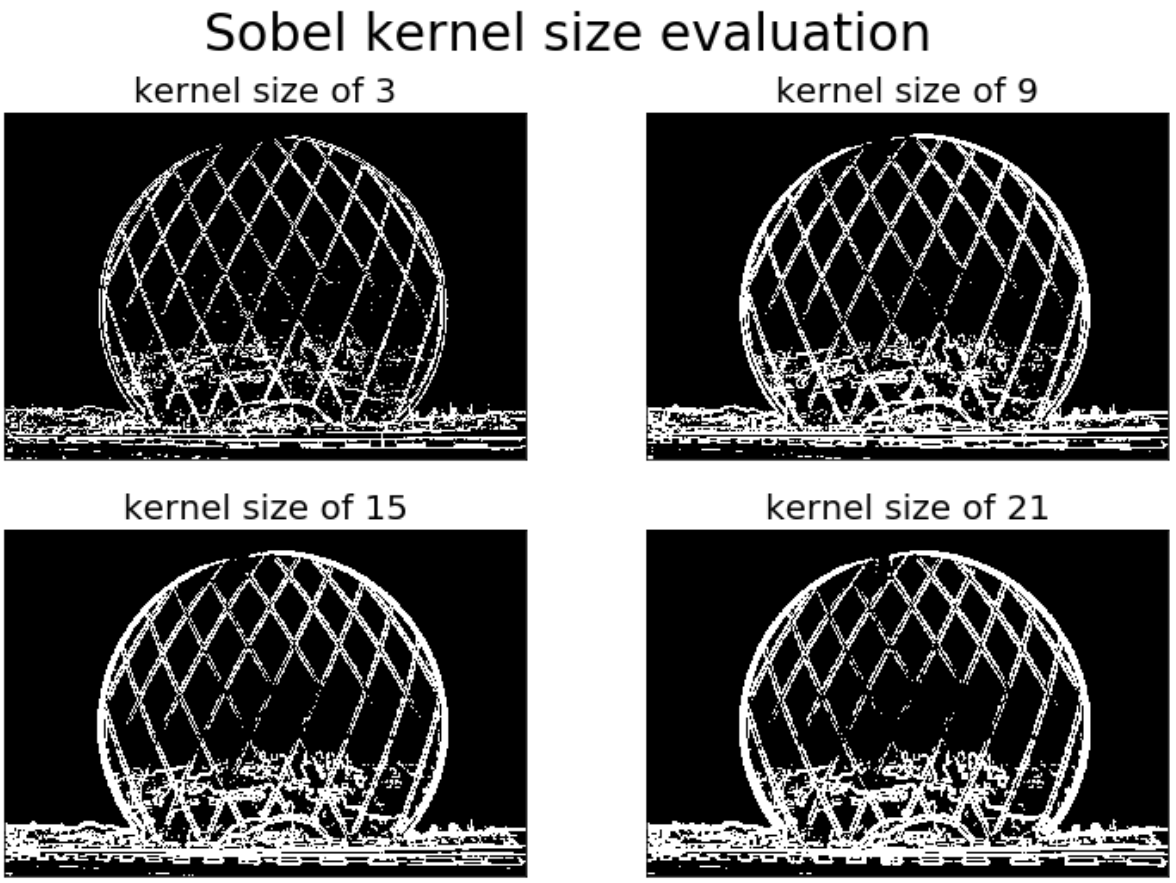
Sobel kernel size evaluation:

Based on the following figure, it can be figured out that as the kernel size of Sobel operation increases, two major changes occur:

1. Probably, increase in kernel size leads to more blurriness of the image; therefore, less noises are considered as edges. Hence, increase in kernel size improves the edge detection from this point of view.
2. It seems that one downside of increase in kernel size may be increase in thickness of detected edges. Since we prefer to illustrate edges by lines with one pixel

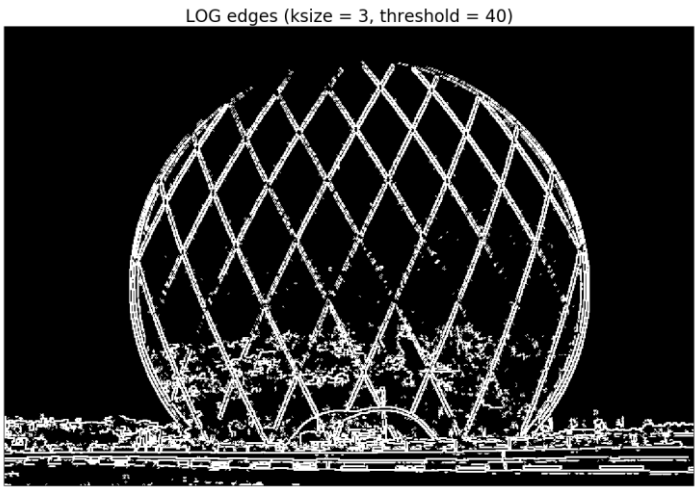
width, this can be considered as a pitfall of increase in kernel size.

Related figure is as follows:



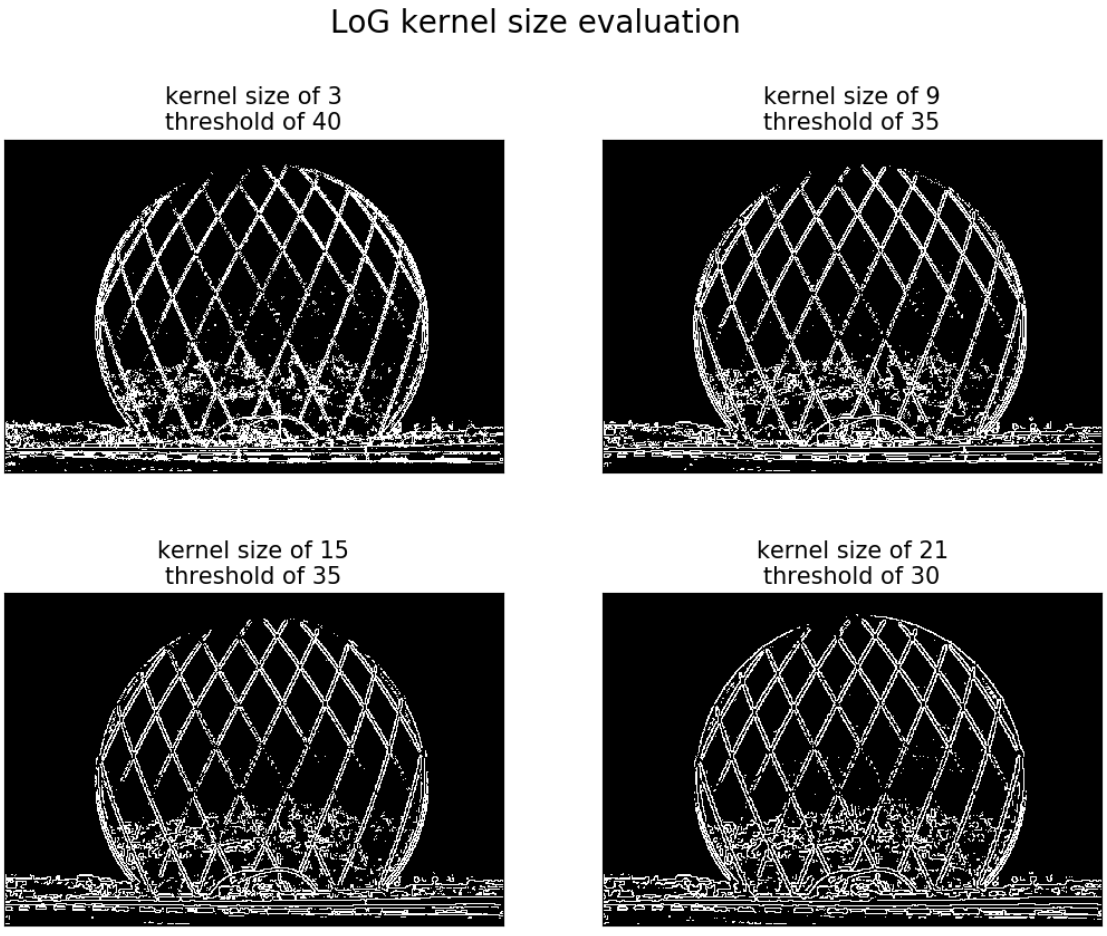
LOG:

Detected edges by LOG operation (with kernel size of 3 and threshold of 40) are as follows:



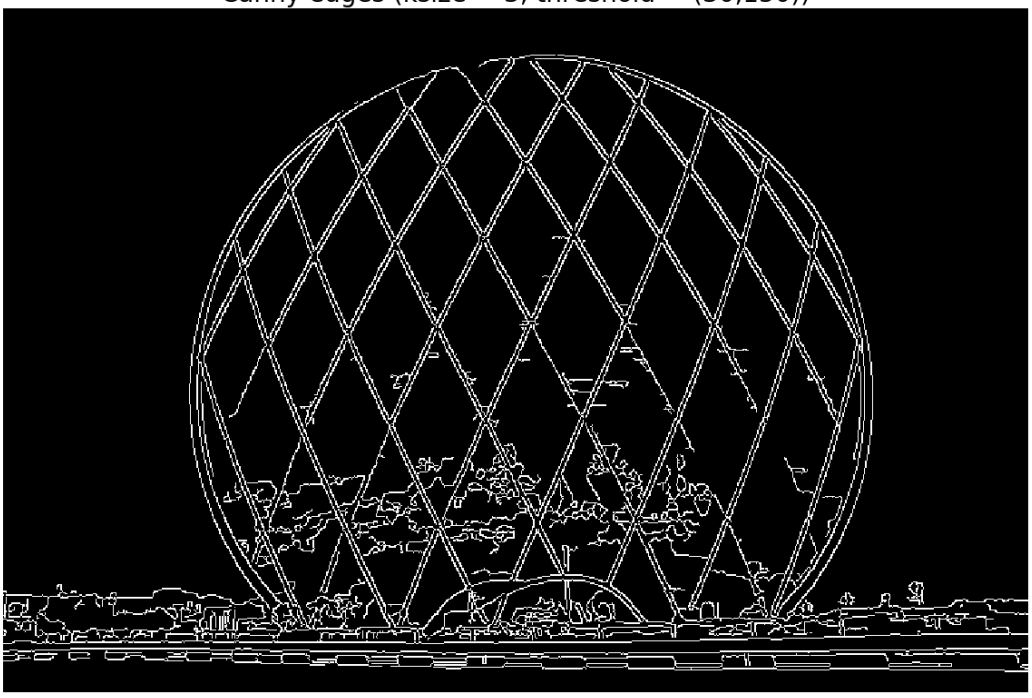
LOG kernel size evaluation:

Based on the following figure, it can be figured out that as the kernel size of LOG operation increases, robustness of operation against noises improves and edges are better illustrated.



Canny:

Canny edges (ksize = 3, threshold = (30,150))



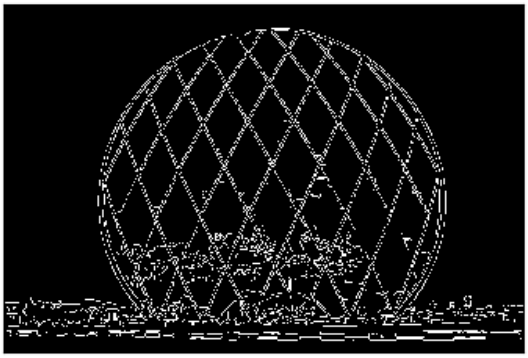
Canny kernel size evaluation:

Based on the following figure, it can be figured out that as the kernel size of Canny operation increases, operation seems to become more sensitive to noises; hence, it requires more gaussian smoothing kernel size in order to obtain acceptable output.

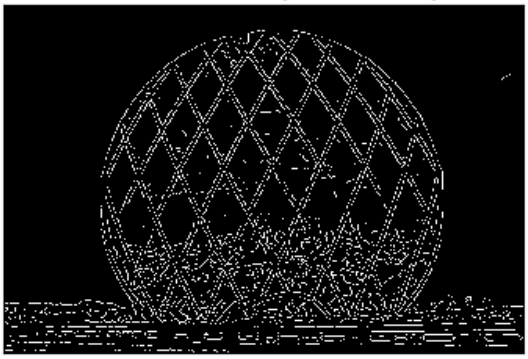
The results of canny operation for three kernel sizes of 3, 5, and 7 are as follows:

Canny kernel size evaluation

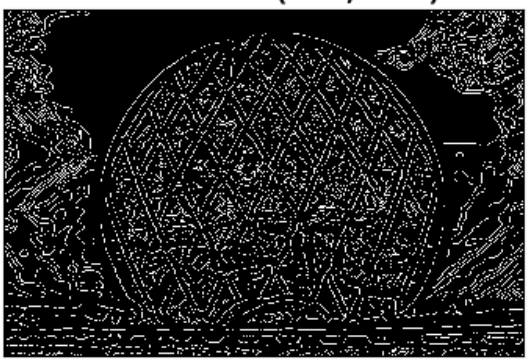
kernel size of 3 & gaussian size of (5, 5) & threshold of (30, 150)



kernel size of 5 & gaussian size of (15, 15) & threshold of (210, 240)



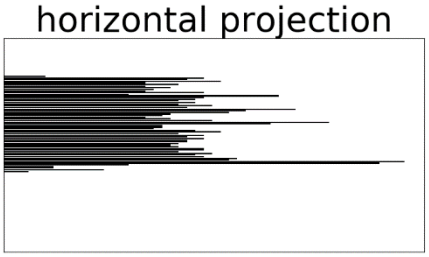
kernel size of 7 & gaussian size of (27, 27) & threshold of (200, 230)

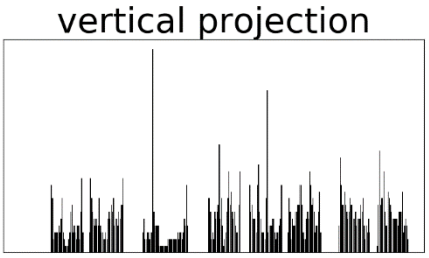


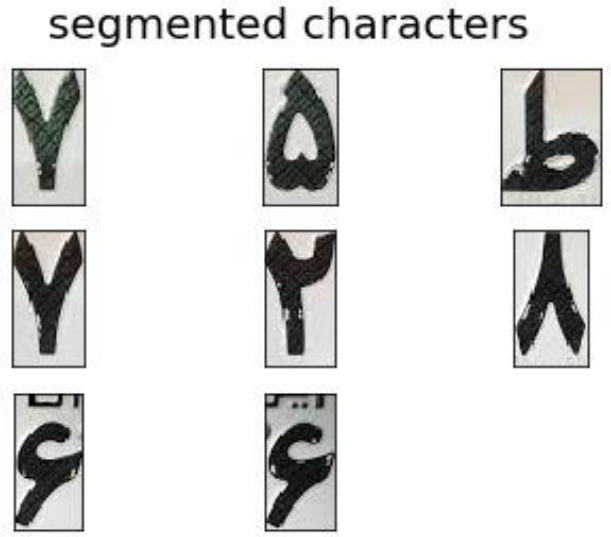
5.

To solve this question, Canny edge detector was applied. This selection is mainly because of the fact that this operation is more robust to noise, and more importantly, detected edges are quite continuous and joined which works wonders for character segmentation which is based on vertical and horizontal projection histograms.

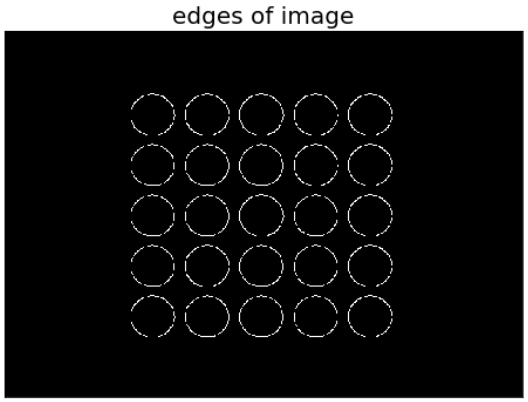
Objects1:

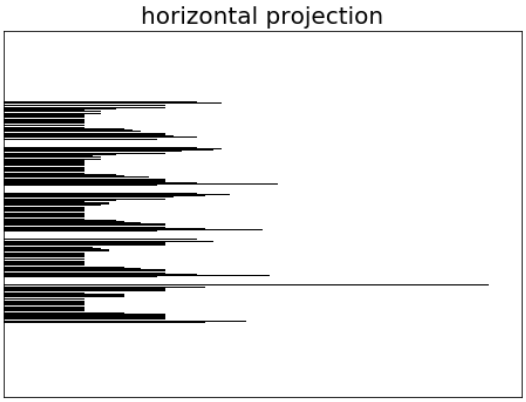


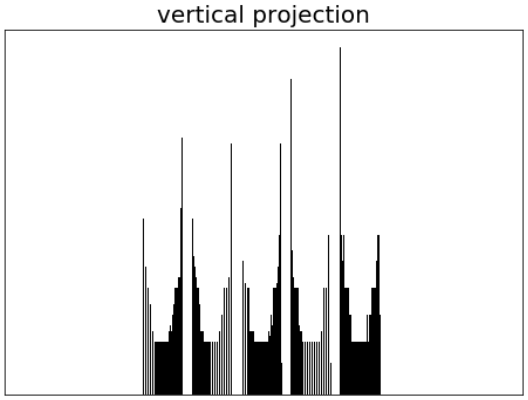


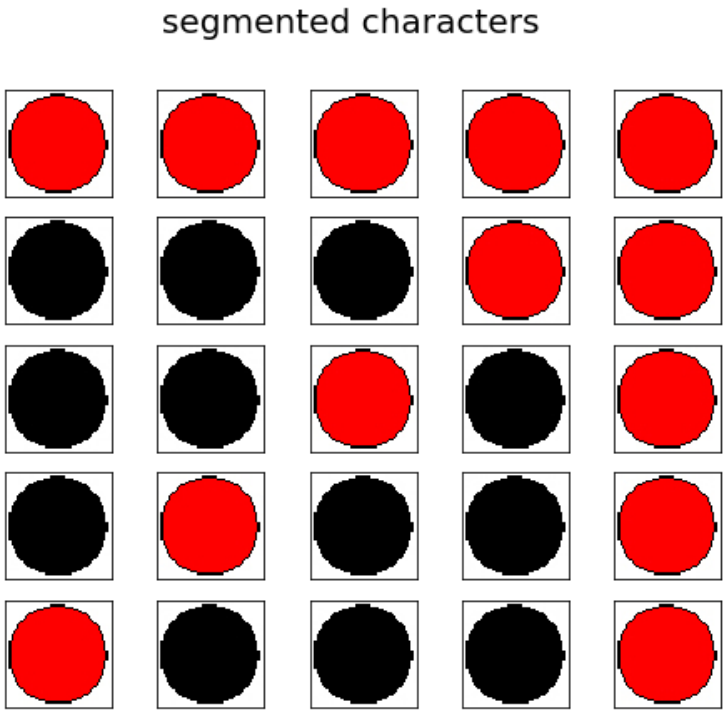


Object2:


edges of image


horizontal projection


vertical projection


segmented characters