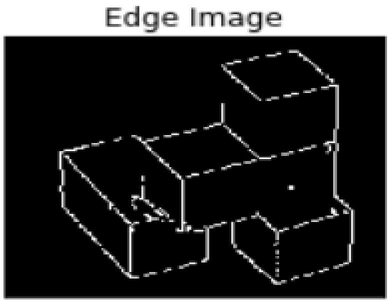
Q1. Edge Detection

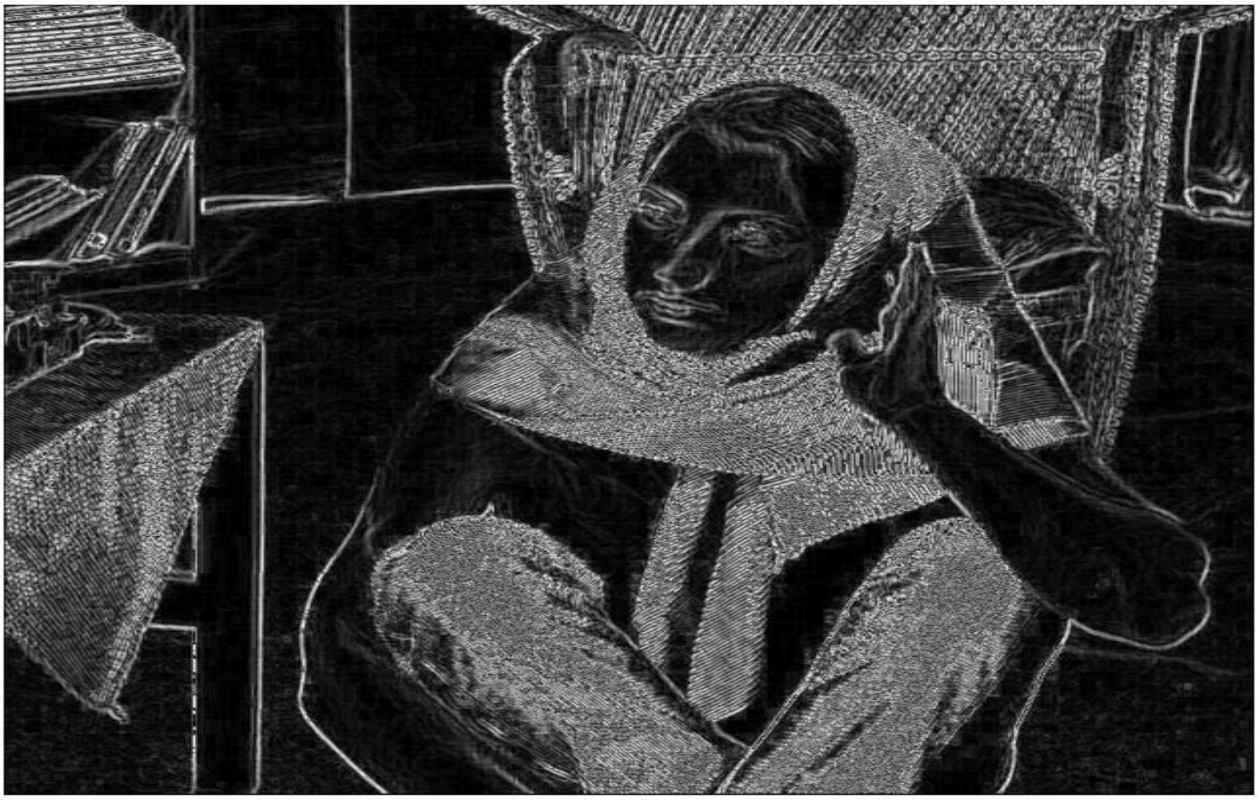
1) Apply the Canny edge detector(use cv2.Canny or MATLAB’s edge function) to bell.jpg and cubes.png. Tweak the values of the arguments(minVal and maxVal), and report the values that give the best results for each image. Hint: For the bell image, try to detect as many edges in the bell as possible, while avoiding edges in the background ?



Bell = (50,250) and Box = (50,150)

2) Consider Roberts, Prewitt, and Sobel filters and Laplacian filters. Apply these filters on barbara.jpg and make observations upon comparing their outputs. Compare these with the output of Canny edge detector on the same image ?

Prewitt



Sobel

Roberts

Laplacian 1

Laplacian 2

Canny

Prewitt - Prewitt operator is used for detecting edges horizontally and vertically and is the simplest of all with a small, separable, and integer valued filter

Sobel Operator - The sobel operator is very similar to Prewitt operator, here more weight means more edge detection.

Roberts - Roberts operator, approximate the gradient of an image through discrete differentiation. Robert mask is 2 × 2 matrix. Robert mask convolve with the entire image using horizontal and vertical Robert masks to give edge detected image in x direction and y direction respectively.

Laplacian - Laplacian is a second order derivative, the Laplacian edge detector uses only one kernel. It calculates second order derivatives in a single pass

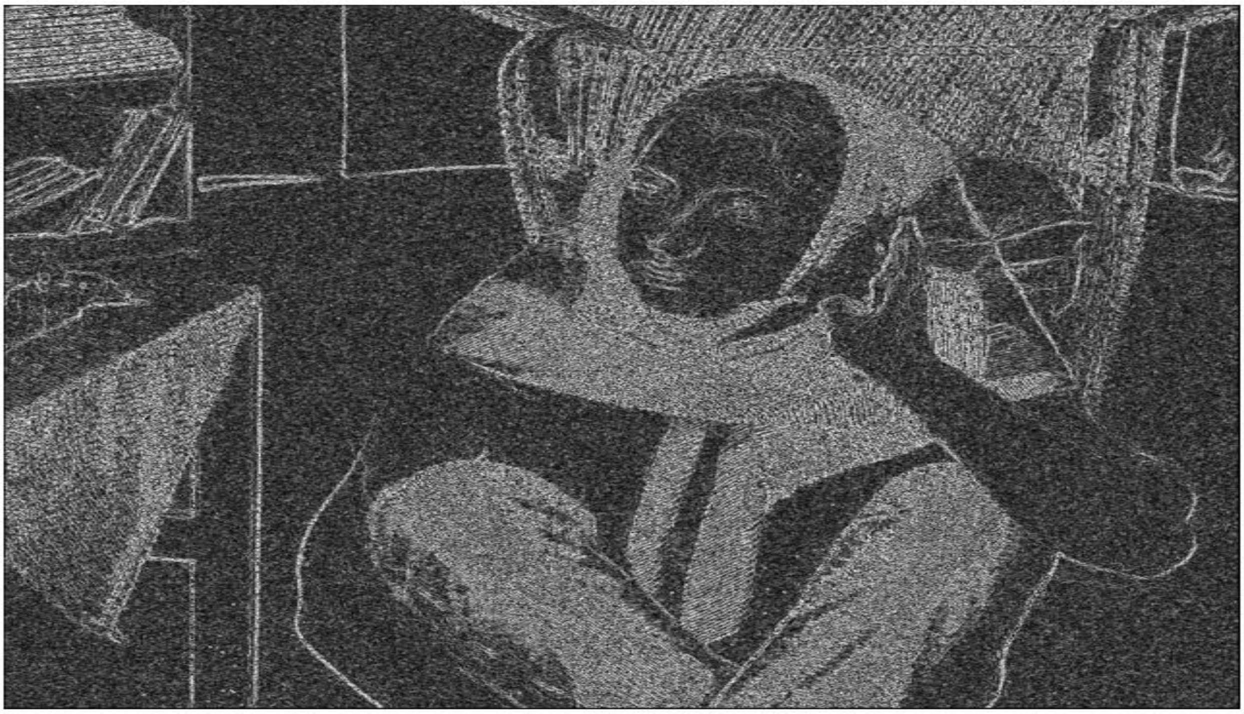
Canny – Canny is a multistage to detect various type of edges. It follows the below steps:

- Apply gaussian filter to smooth the image and remove noise
- Finding intensity gradients
- Doing non maximum suppression
- Doubling the threshold
- Edge tracking by hysterisis

3) Add noise to the input image above using Gaussian sampling. Study the effect of applying the filters on noise-affected inputs. ?

Noisy Image

Prewitt Noisy



Sobel Noisy

Roberts Noisy

Laplacian 1 Noisy

Laplacian 2 Noisy

- all gradient based (first order derivative) edge detections such as prewitt, sobel, roberts and canny are sensitive to noise.
- when the noise occurs these operators have a tendency to assume it as a part of the edge and also sometimes it misses the true edges due to corruption of noise
- canny overcomes this problem by applying gaussian filter on noisy image which reduces noise to great extent.
- Robert’s yield the worst on noise and canny gives very good output.

- Laplacian (is second order derivative), the Laplacian edge detector is extremely sensitive to noise. Laplacians are computationally faster to calculate (only one kernel vs two kernels as that in gradient based)

Q2. Prove that subtracting the Laplacian from an image is proportional to unsharp masking.

Laplace Operator

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$$= f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y) \quad \text{——— (1)}$$

To Prove :- $fs(x,y) = f(x,y) - \nabla^2 f(x,y)$

where, $fs(x,y)$ = unsharp masking

$f(x,y)$ = original image.

$\nabla^2 f(x,y)$ = Laplacian.

Solution :- Unsharp Masking

$$\boxed{fs(x,y) = f(x,y) - \bar{f}(x,y)}$$

where, $\bar{f}(x,y)$ is smooth image

$fs(x,y)$

$= f(x,y) - \nabla^2 f(x,y)$

$= f(x,y) - [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)]$

$= 6f(x,y) - [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) + f(x,y)]$

$= 5\left[\frac{6}{5} f(x,y) - \underbrace{\frac{1}{5}\left(f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) + f(x,y)\right)}\right]$

$= 5\left[\frac{6}{5} f(x,y) - \bar{f}(x,y)\right]$

Page No.
Date: / /

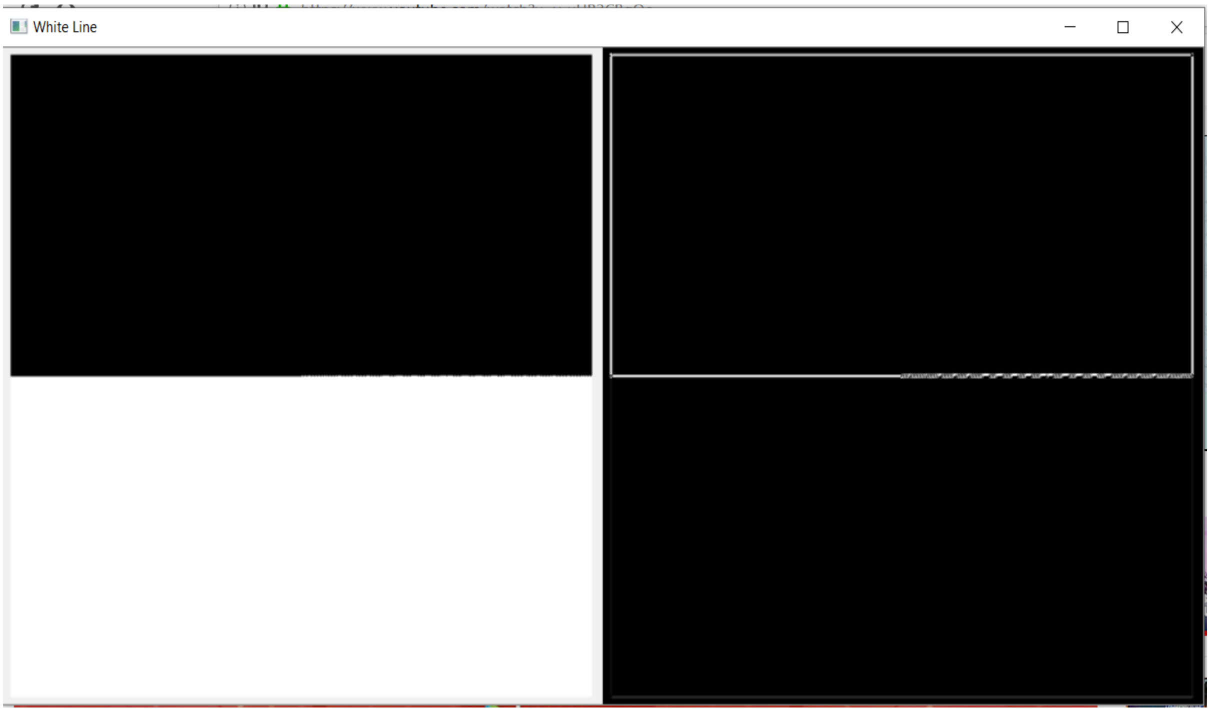
where $\bar{f}(x,y)$ denotes the average of $f(x,y)$ in a predefined neighbourhood that is centered at (x,y) and includes the center pixel & its four immediate neighbours.

$$\therefore \underbrace{f(x,y) - \nabla^2 f(x,y)}_{\text{subtracting Laplacian from image}} \propto \underbrace{f(x,y) - \bar{f}(x,y)}_{\text{unsharp masking}}$$

Hence, Proved,

Q3.

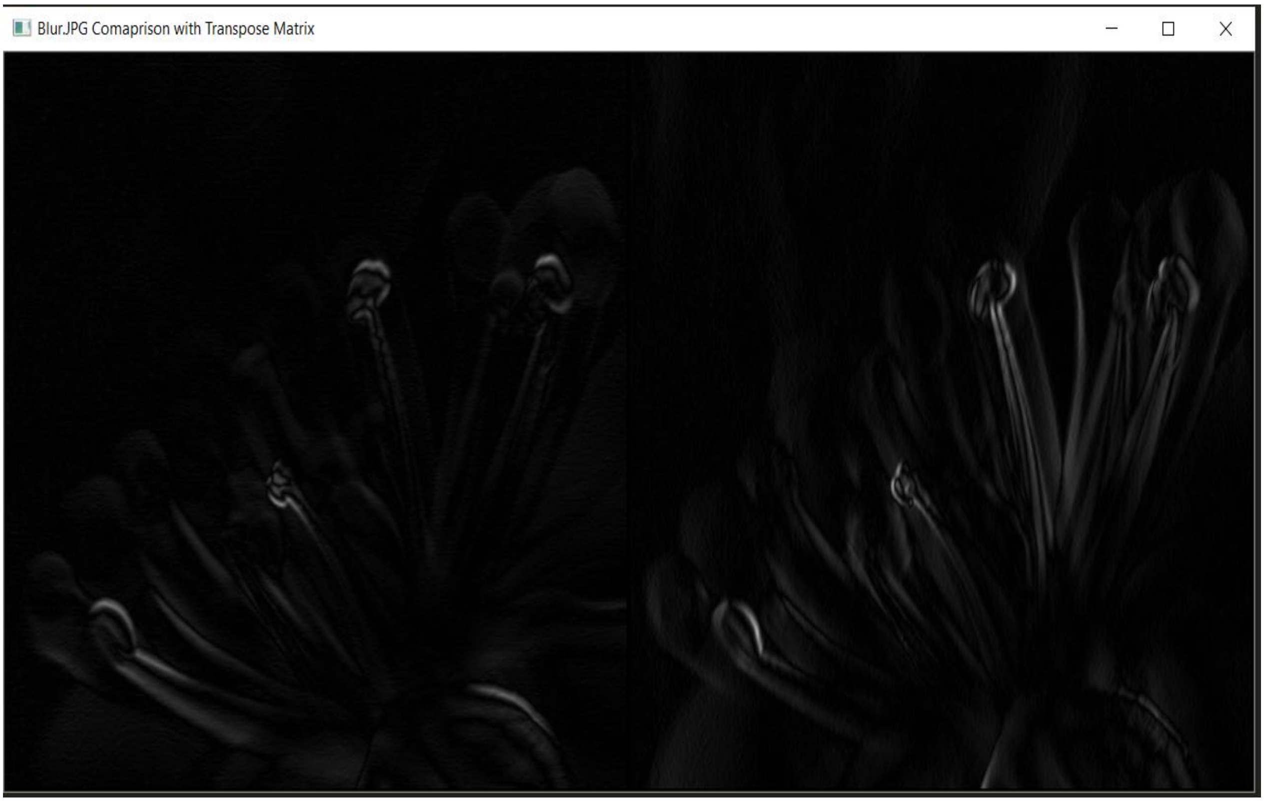
1) Create a matrix of size 3x3 which when convolved with box.png results in a white line where the white meets the black



Matrix $A_{(3x3)}$ =

1	1	1
1	0	-1
-1	-1	-1

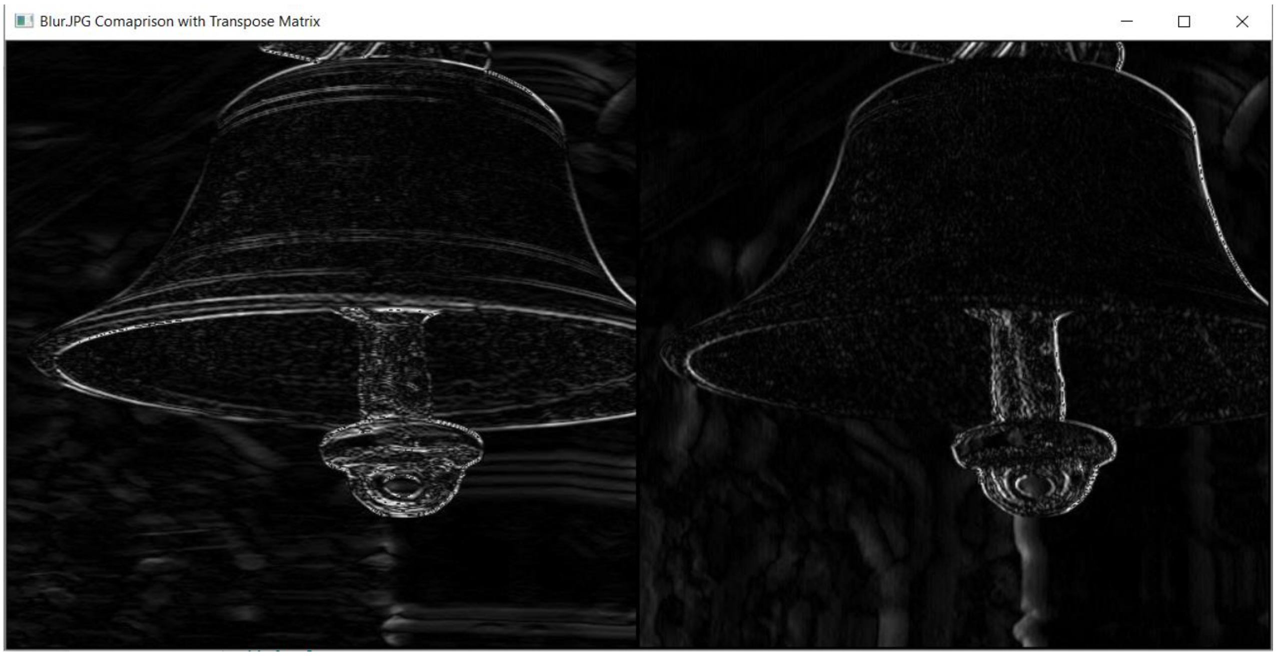
2) Convolve blur.jpg with the above matrix and the transpose of the above matrix. Report your observations ?



Convolved with A Convloved with A^T

As we see that in the first picture when convolved with matrix A, all the vertical edges are more visible. Similarly, in the second picture when convolved with A^T, all the horizontal edges are visible. So, in this way we can see that we can detect both horizontal and vertical edges from an image.

3) Follow the above steps for suitable images of your choice?



Convolved with A 	Convloved with A^T

Q4. High-Boost Filtering

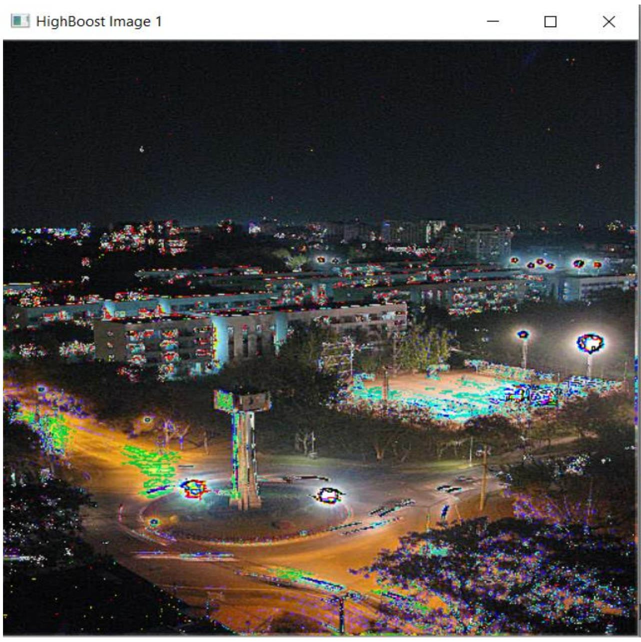
1) Implement high-boost filtering on the image bell.jpg varying the window size and the weight factor and report your observations?

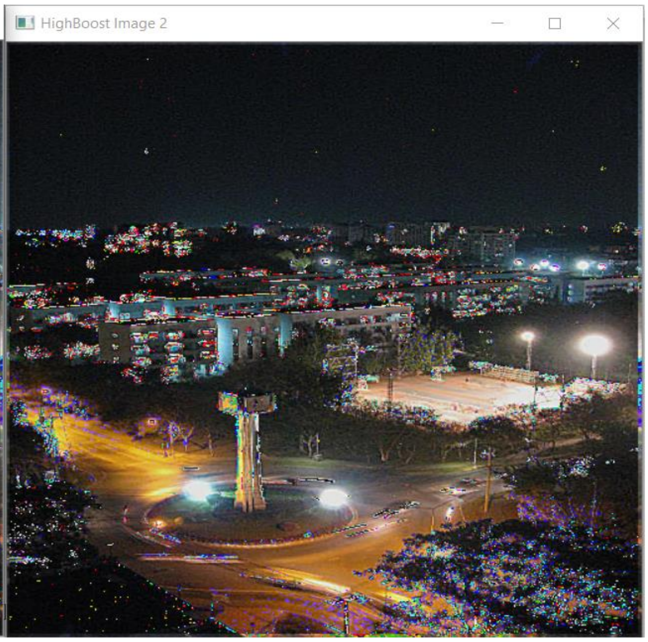
K = 3

K = 5

The low spatial frequency components are suppressed while the high spatial frequency components are obtained. As the K size is increased the image has got sharpen slightier.

2) Repeat the above exercise on suitable images of your choice ?





K = 3

K = 5

3) How is bilateral filtering different from high-boost filtering ?

➔ Bilateral Filtering

- It is a non-linear, edge preserving and gives smooth images.
- The output pixel value depends on a weighted combination of neighbouring pixel values

➔ High Boost Filtering

- It sharpens the image or performs edge enhancement using smoothing filter.
- Here we first smooth the image - we suppress most of the high-frequency components.
- We subtract this smoothed image from the original image and the output image will be the mask which will have most of the high frequency components because the smoothing filter will block this high frequency.
- Finally, we will add the mask back with original image that enhances the high frequency components

➔ Comparing both bilateral and high boost filter

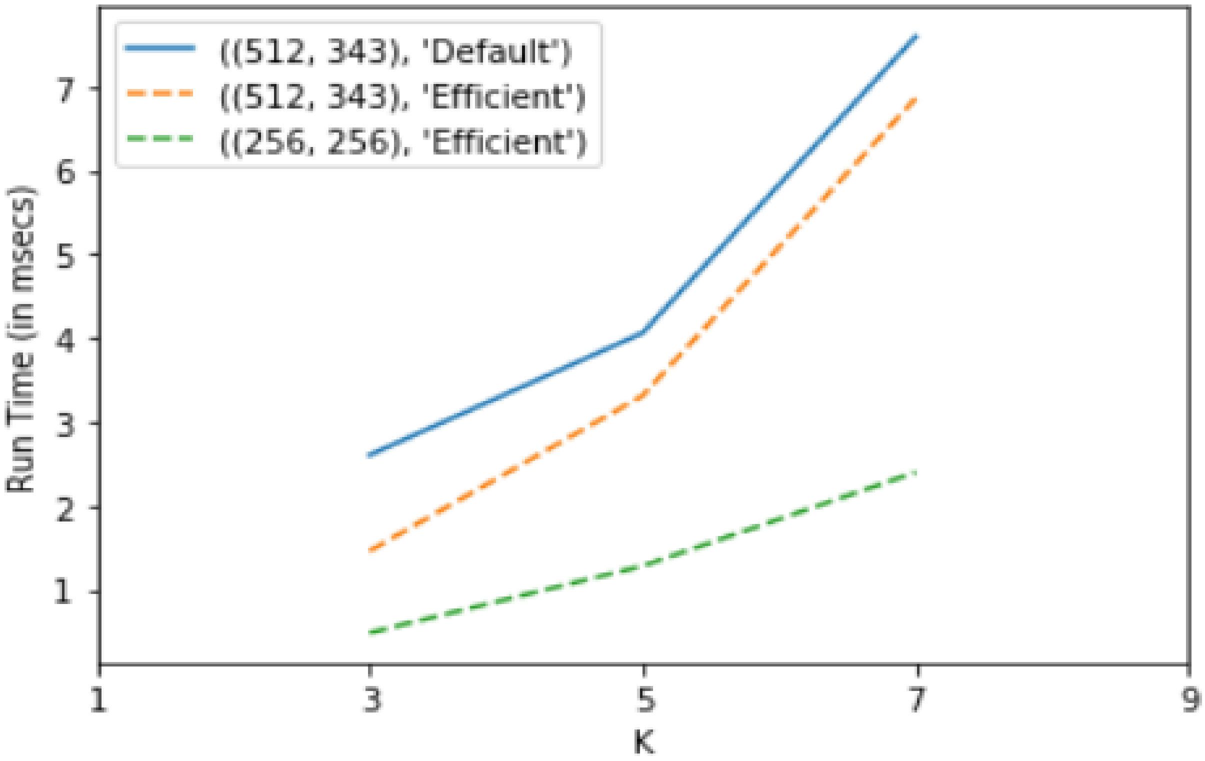
- Bilateral provides smooth images and high boost filter subtracts smooth images from original image.
- In bilateral output, pixel depends on weighted combination of neighbour pixels. which will smooth the image. While in high boost filter because of addition with mask the output pixel gets enhanced further i.e frequency enhances.

Q5.

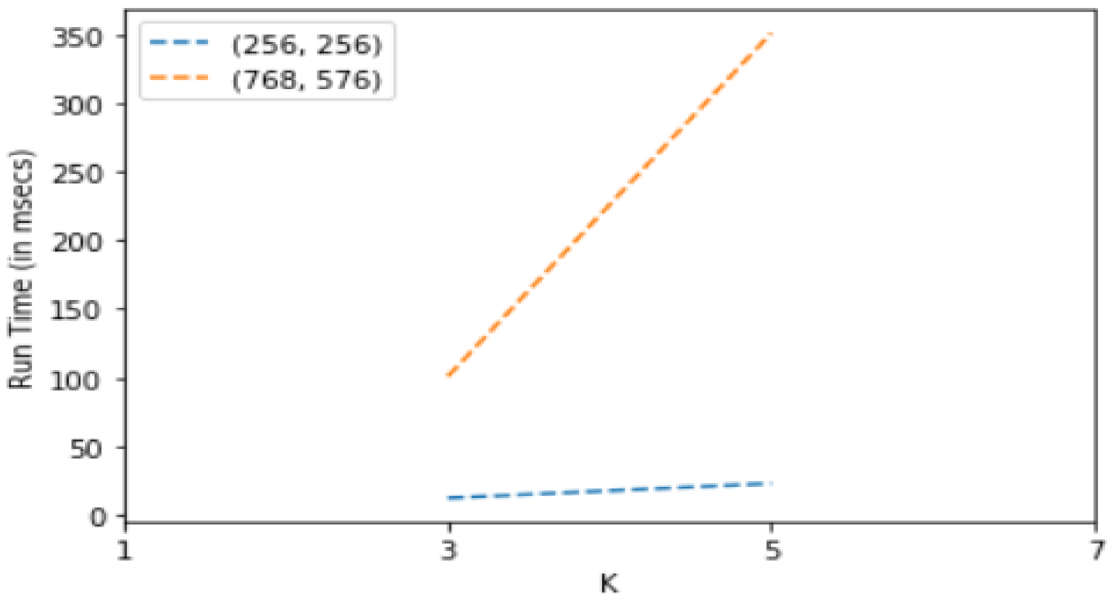
1) Implement an algorithm for low-pass filtering a grayscale image by moving a k×k averaging filter of the form ones(k)/(k^2).

Filtered Image (Linear Filter)

2)

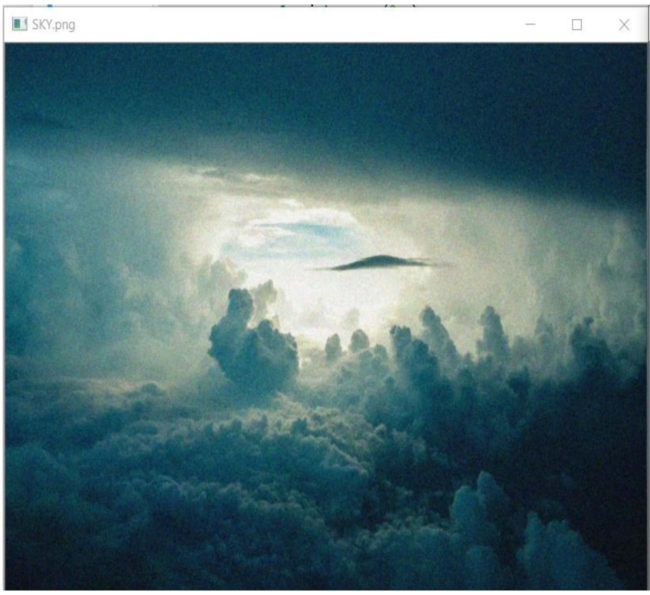

((512, 343), 'Default')
((512, 343), 'Efficient')
((256, 256), 'Efficient')
Run Time (in msecs)
K

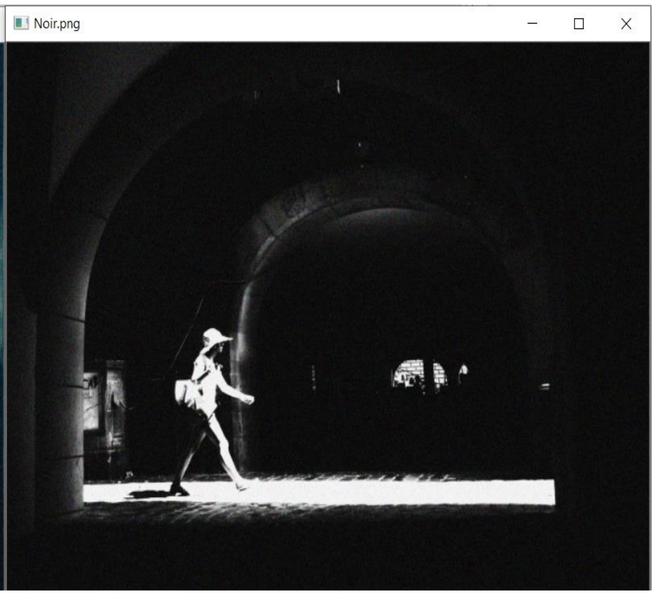
3)


(256, 256)
(768, 576)
Run Time (in msecs)
K

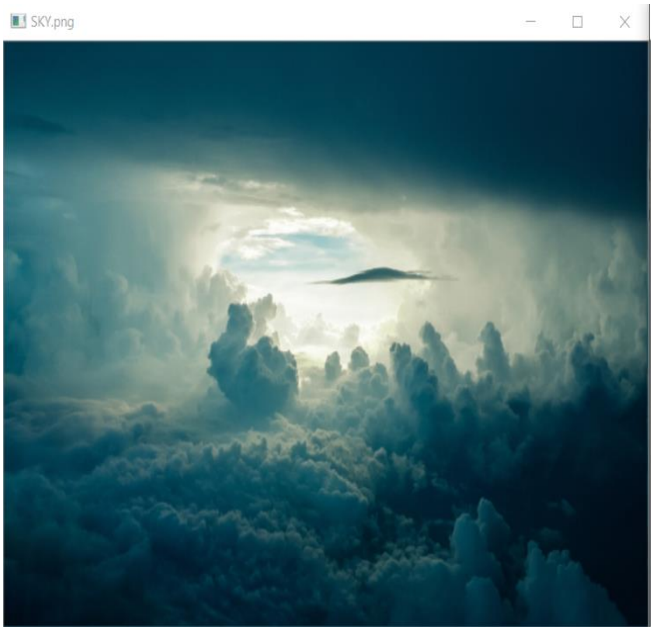
Q6. Bilateral Filter

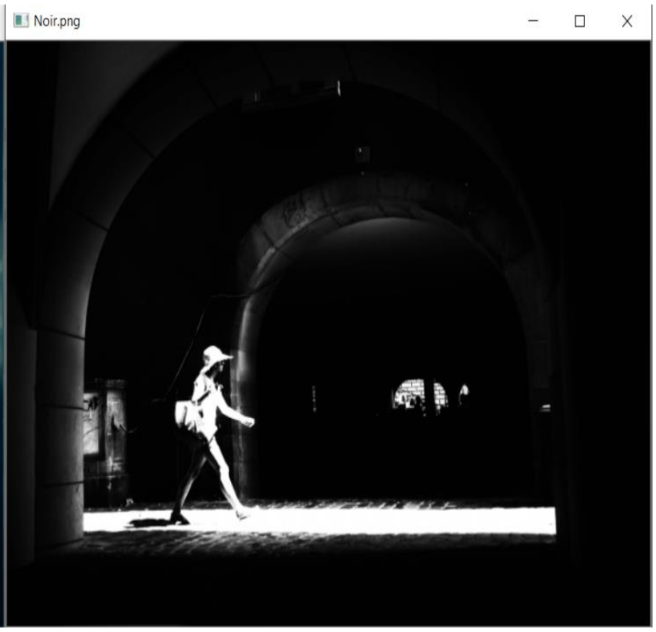
1) Implement bilateral filter and apply it to sky.png and noir.png



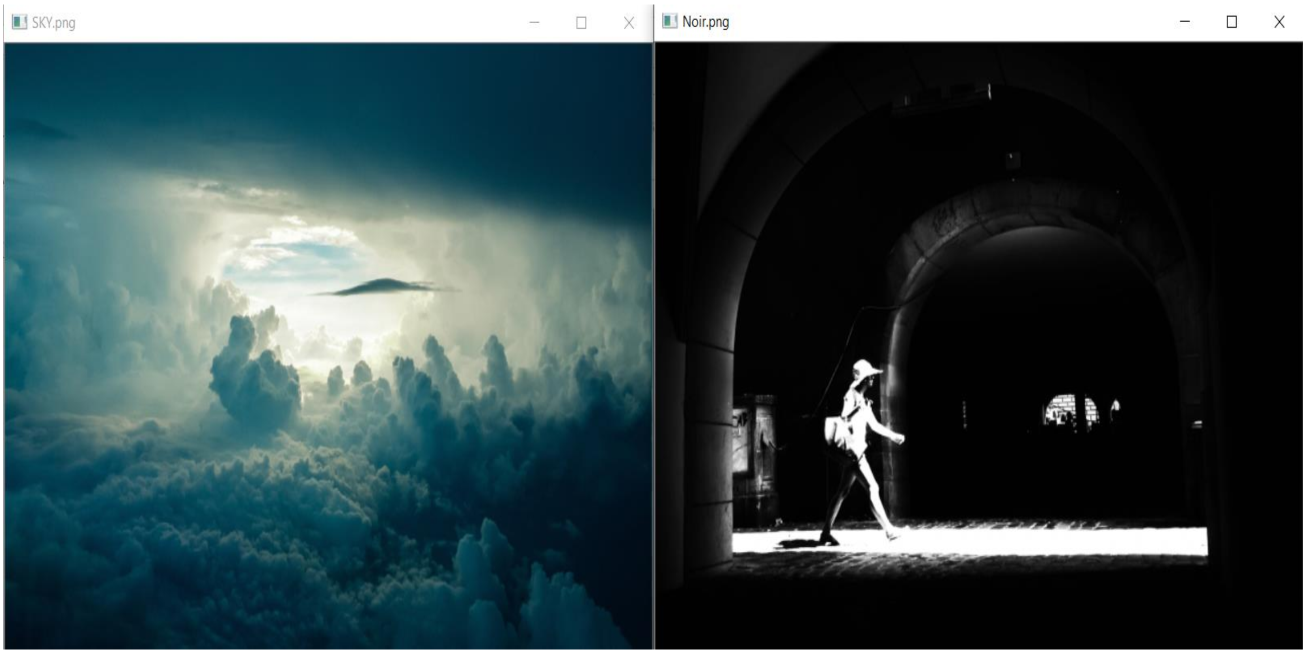


2) Vary the effect of domain and range components of the bilateral filter and try to minimize the L2 distance from the ground truth images. Display the images and the approximate values for the sigma

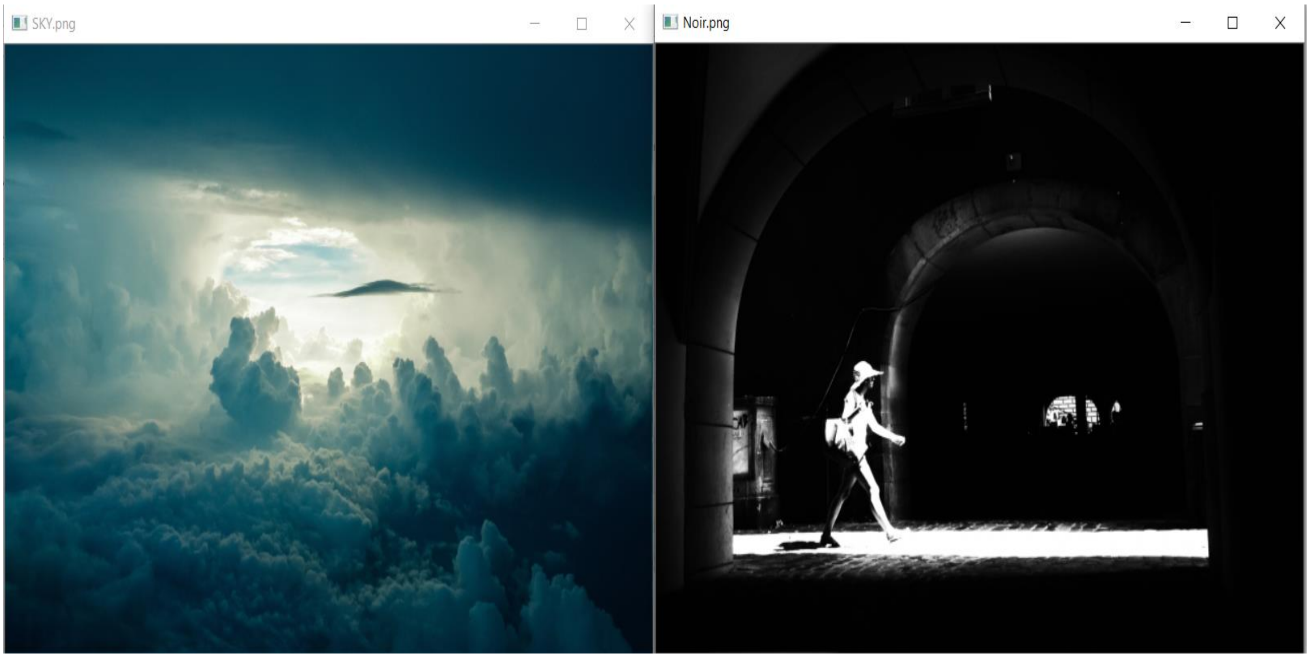




L2 Distance = 10



L2 Distance = 5



L2 Distance = 2

Here as I am decreasing the Distance value, I can see that the darkness of the image is also decreasing.

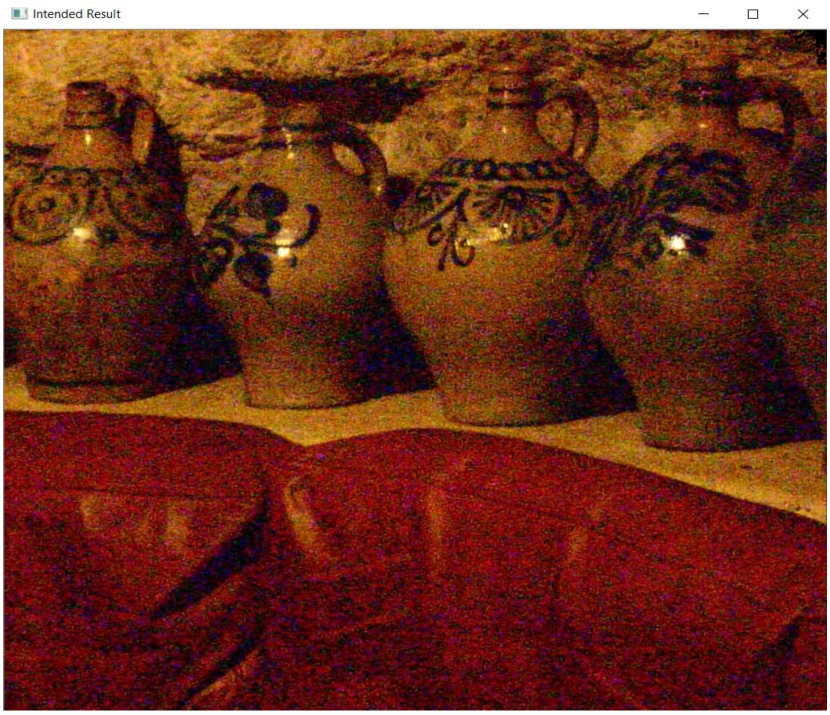
On L2 distance = 10, image is darker

On L2 distance = 5, image becomes less darker than previous

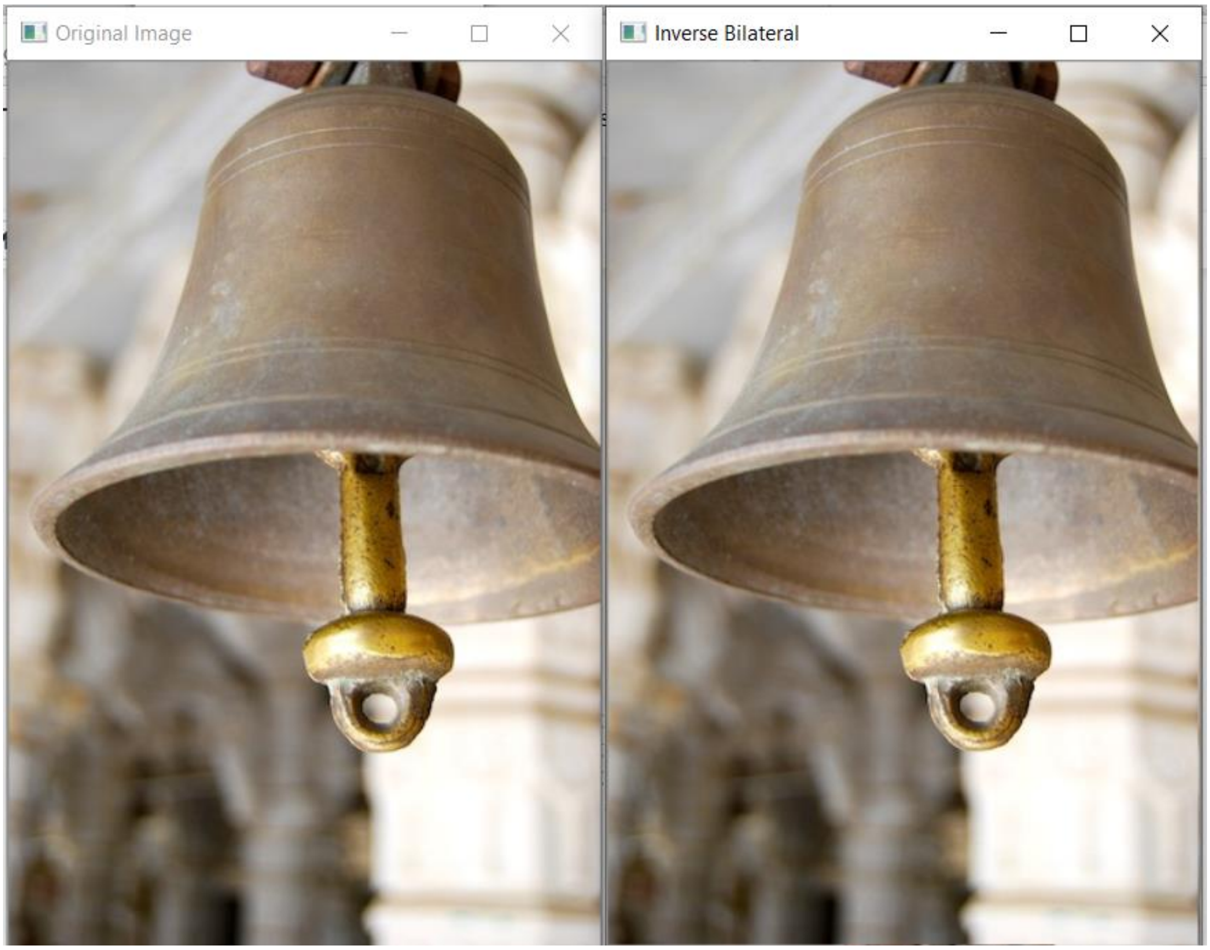
On L2 distance = 2, image becomes brighter than previous two images.

Q7. Applications of Bilateral Filter

1) Cross Bilateral Filter



2) **Inverse Bilateral Filter** : Does it makes sense to develop an inverse bilateral filter, which blurs an image at edges and preserves the homogeneous regions. If it makes sense, design it and suggest its applications



Q8. Image Restoration

- Degraded.jpg has noise in it (i.e salt and pepper noise) and to remove the noise from the image I have used median filter.
- As median filter is non-linear filtering technique and used for removal of noise from images. The main idea of the median filter is to run through the every pixel entry by entry, replacing each entry with the median of neighbouring entries
- Steps-
    1. Taking degraded image.
    2. Restoring the image by applying median filter on degraded image with window size of 3.
    3. Displaying the restored image.