Canny edge Detector:

Canny Edge Detection

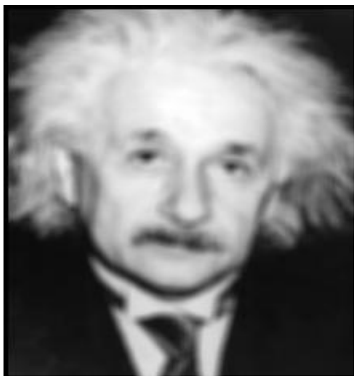
Introduction:

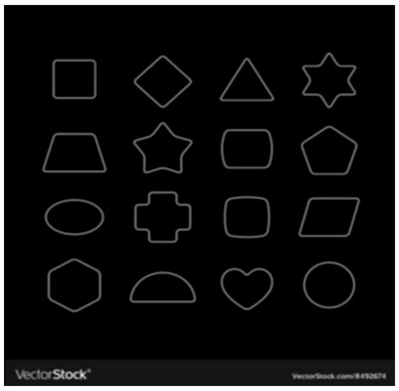
In this assignment we are going to implement canny edge detection algorithm and harris corner detection algorithm with out using any inbuilt library functions like cv2.canny() etc.

The Canny edge detection algorithm is composed of 5 steps:

1)Noise reduction:

Convert the input image from colour to gray-scale. To get rid of the noise on the image, I applied Gaussian blur to smooth it. To do so, image convolution technique is applied with a Gaussian Kernel having sigma =1. If we do not remove the noise there might possibility of getting false edges produced by these noise. The output of the images after smoothing with gaussian filter are:





2) Gradient calculation:

This step detects the edge intensity and direction by calculating the gradient of the image using edge detection operators. Edges correspond to a change of pixels’ intensity. To detect it, the easiest way is to apply filters that highlight this intensity change in both directions: horizontal (x) and vertical (y).

When the image is smoothed, the derivatives Ix and Iy with respect to x and y are calculated. It can be implemented by convolving I with Sobel kernels Kx and Ky, respectively.

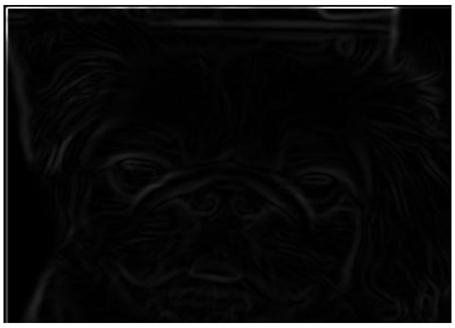
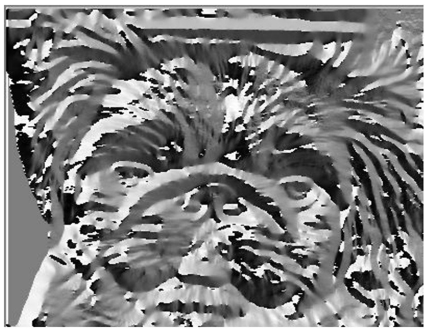
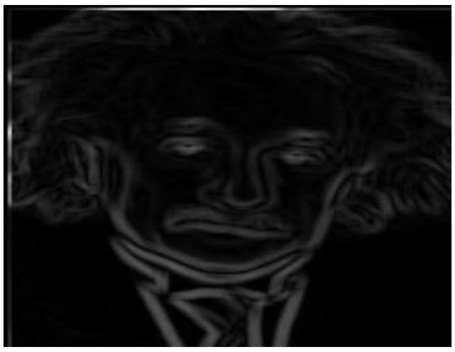
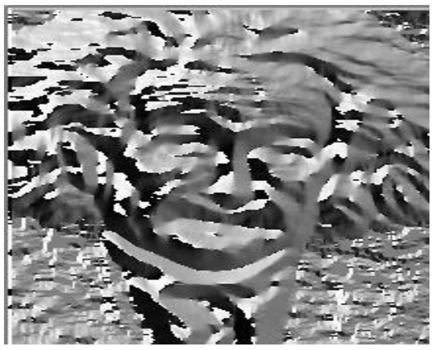
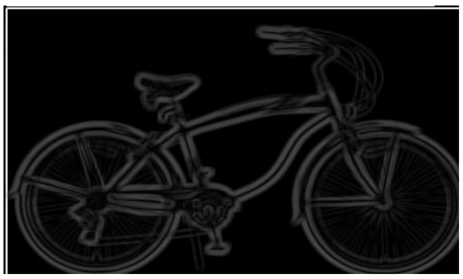
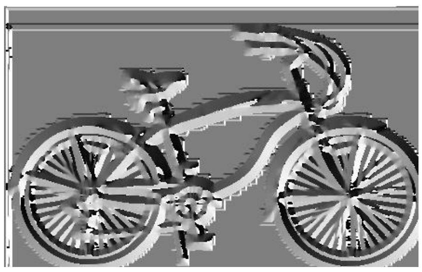
Sobel filters : Sx = [[1, 0, -1], [2, 0, -2], [1, 0, -1]]

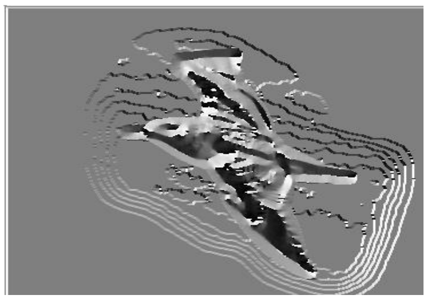
Sy = [[-1, -2, -1], [0, 0, 0], [1, 2, 1]]

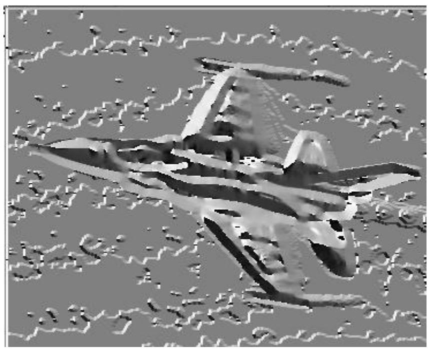
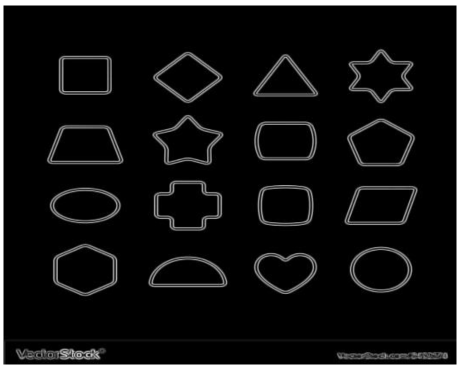
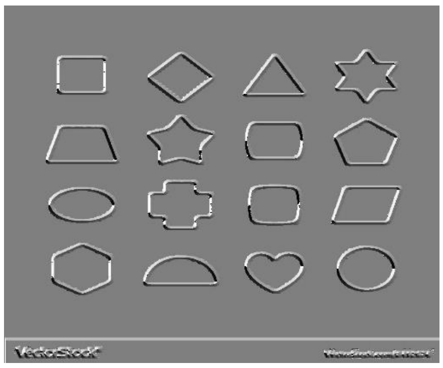
Then, the magnitude G and the slope θ of the gradient are calculated as follow:

$|G| = \text{sqrt}(Ix^{2} + Iy^{2})$ and $\theta = \arctan(Iy/Ix)$

magnitude image **direction image**



3) <u>Non-maximum suppression:</u>

From the result generated in the above step we can see that some of the edges are thick and others are thin. Non-Max Suppression step will help us mitigate the thick ones. Moreover, the gradient intensity level is between 0 and 255 which is not uniform. The edges on the final result should have the same intensity.

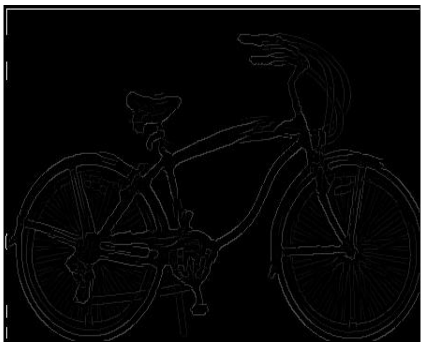
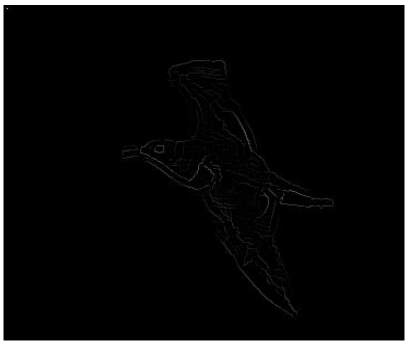
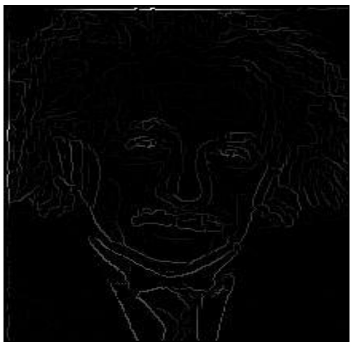
The algorithm goes through all the points on the gradient intensity matrix and finds the pixels with the maximum value in the edge directions. Each pixel has 2 main criteria (pixel intensity and edge direction in radians). Based on these inputs the non-max-suppression steps are:

- Create a matrix initialized to 0 of the same size of the original gradient intensity matrix
- For each pixel , Identify the edge direction in ($0, \pi/4, \pi/2, 3\pi/4$) that is closest to the orientation at that pixel based on the angle value from the angle matrix.
- Check if the pixel in the same direction has a higher intensity than the pixel that is currently processed

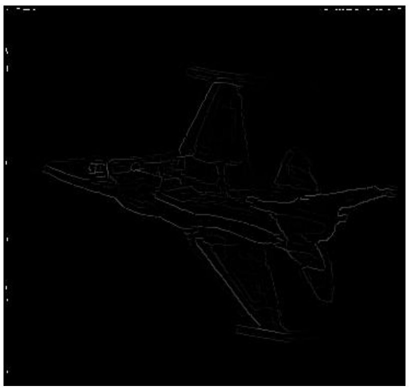
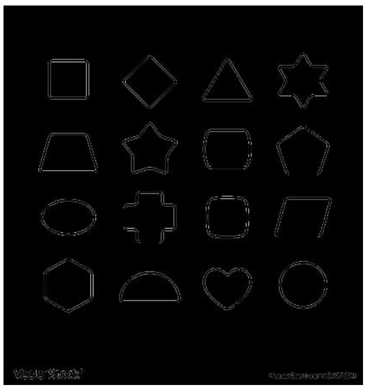
- Return the image processed with the non-max suppression algorithm

NMS(non maximal suppression) Algorithm :

Images obtained after applying Non-maximal suppression alogorithm:

Bicycle **bird** **einstein**

Dog **plane** **toy image**

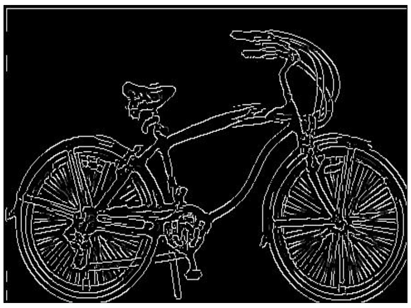
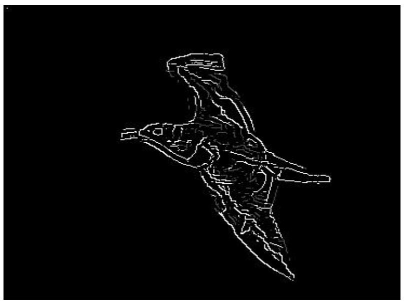
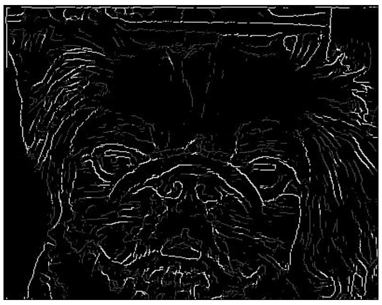
4)<u>**Double threshold:**</u>

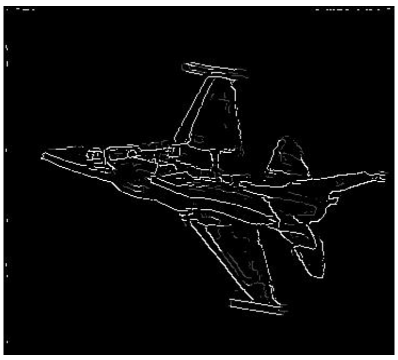
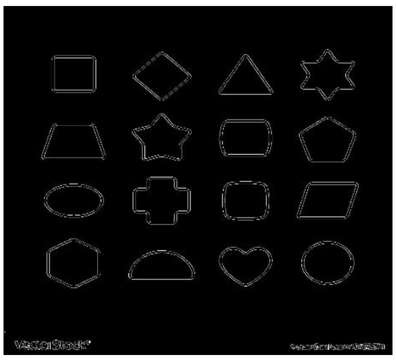
The double threshold step aims at identifying 3 kinds of pixels: strong edge, weak edge, and definitely not edge:

- Strong pixels are pixels that have an intensity so high that we are sure they contribute to the final edge.
- Weak pixels are pixels that have an intensity value that is not enough to be considered as strong ones, but yet not small enough to be considered as non-relevant(not edge) for the edge detection.
- Other pixels are considered as non-relevant for the edge.
- High threshold is used to identify the strong pixels (intensity higher than the high threshold)
- Low threshold is used to identify the non-relevant pixels (intensity lower than the low threshold)

- All pixels having intensity between both thresholds are flagged as weak and the Hysteresis mechanism (next step) will help us identify the ones that could be considered as strong and the ones that are considered as non-relevant.

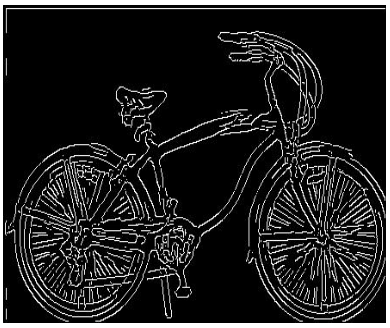
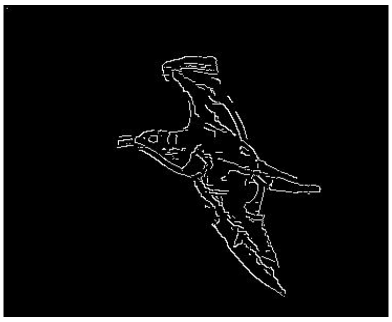
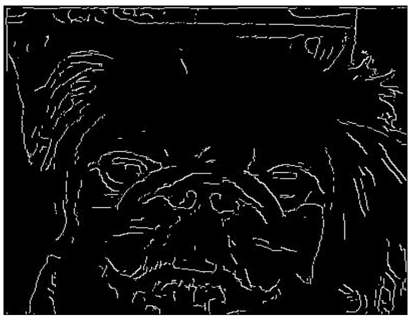
The resultant images are shown below:

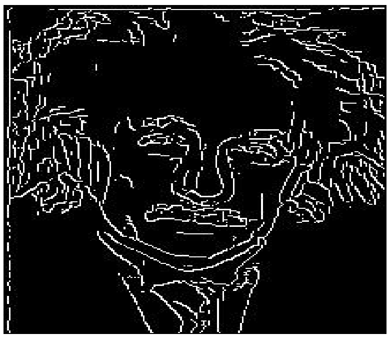
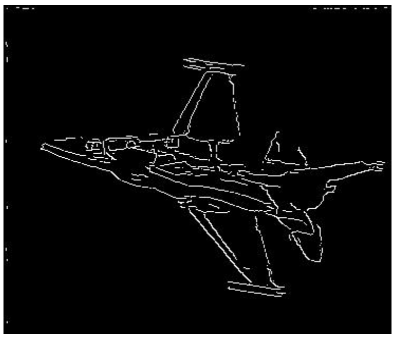
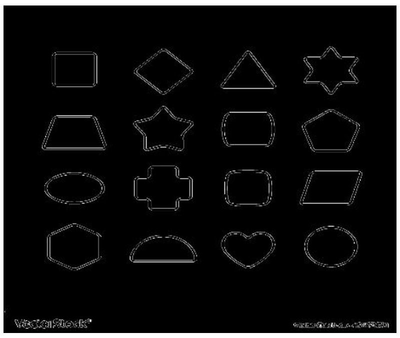




5) <u>**Edge Tracking by Hysteresis:(Hysteresis linking)**</u>

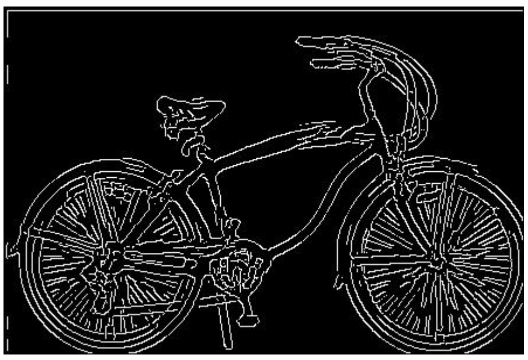
Based on the threshold results, the hysteresis consists of transforming weak pixels into strong ones, if and only if at least one of the pixels around the one being processed is a strong one. For each strong pixel, recursively visit the weak pixels that are in the 8 connected neighbourhood around the strong pixel, and label those also as strong (and as edge). Label as "not edge" any weak pixels that are not visited by this process. The resultant images are :

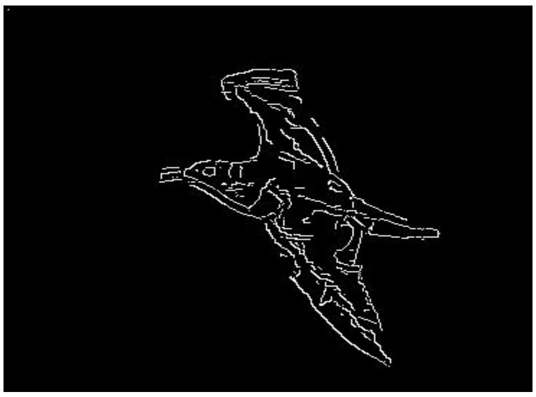
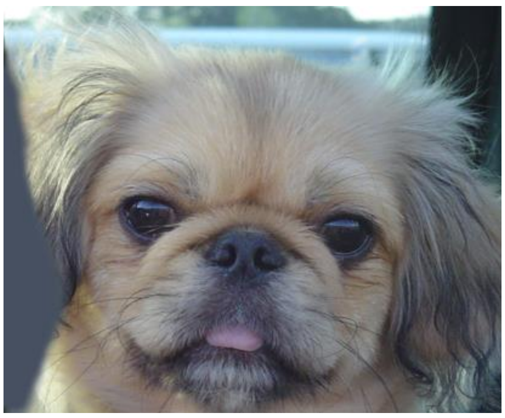
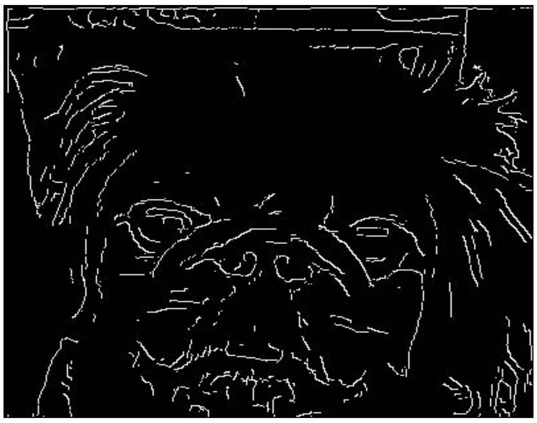




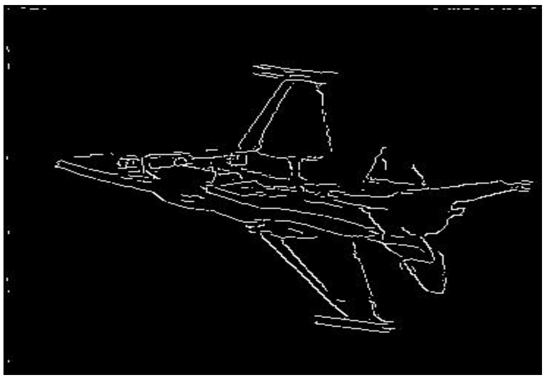
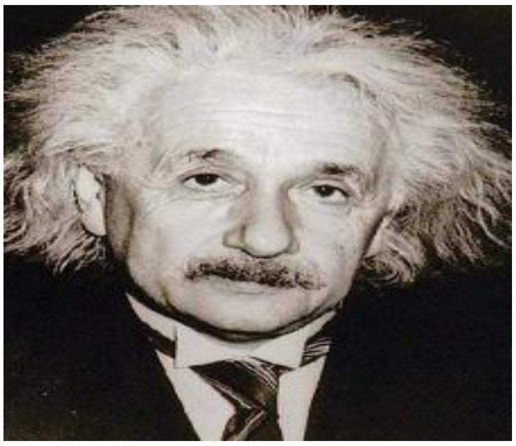
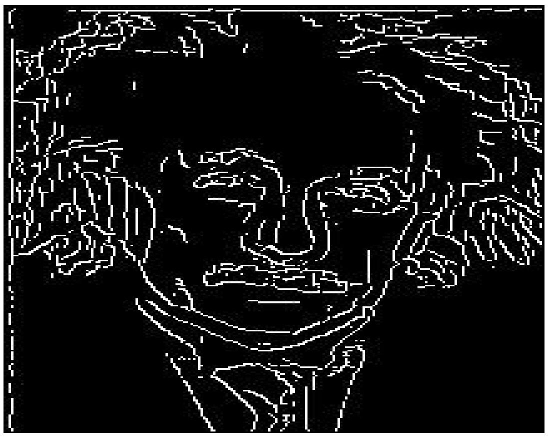
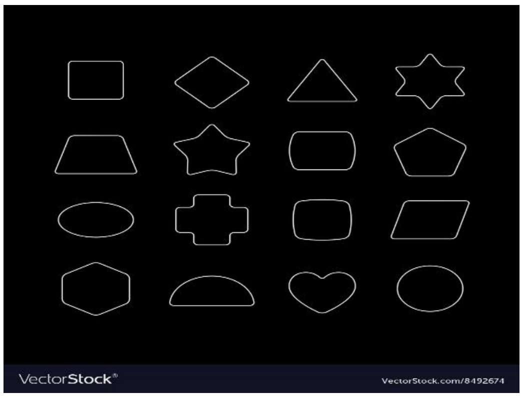
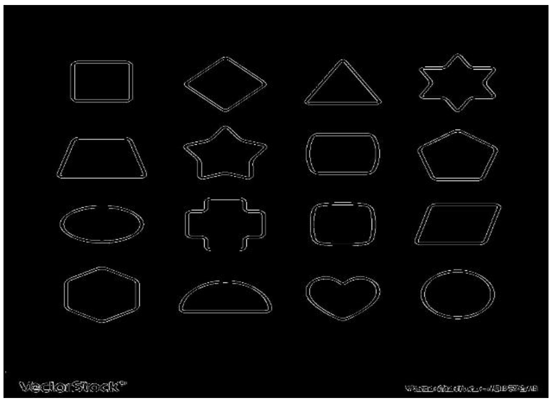




Final output by canny edge detection :

The final images after applying canny edge detection are :



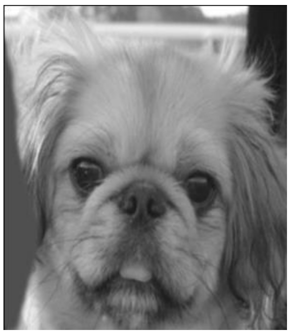




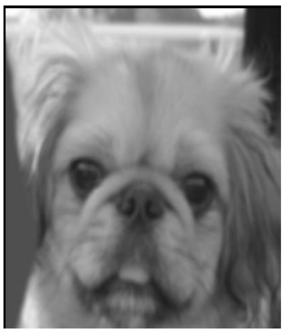




VectorStock®
VectorStock.com/8492674

VectorStock®

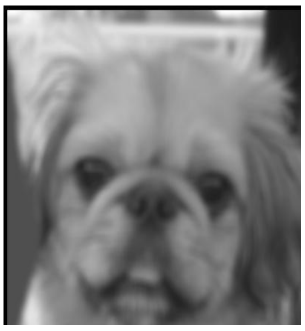
Observations:

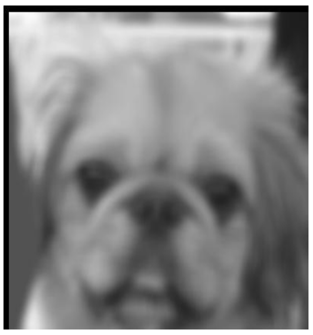
➔ Gaussian images for different size of filters with sigma =1 :

Here filter size = (2*K+1,2*K+1)

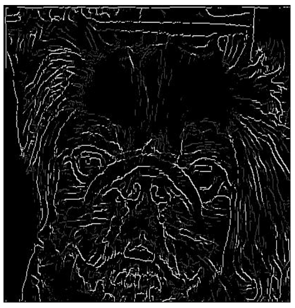

K=1

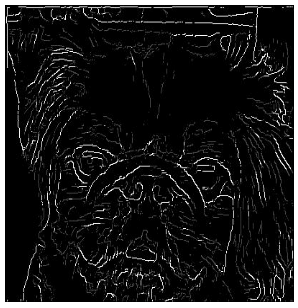

K=3

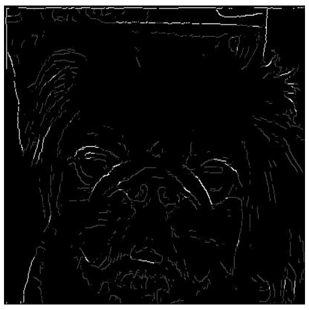

K=5


K =7

➔ Thresolded images for different low and high threshold values:


(low,high) –(15,30)

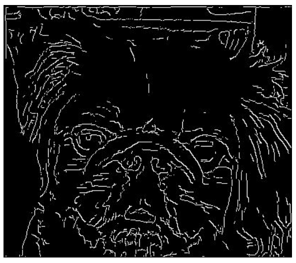

(low,high) –(20,40)

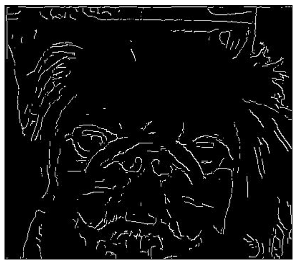

(low,high) –(35,80)

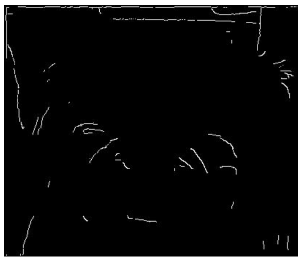
(low,high) –(70,150)

We can observe that as the threshold values increases the loss of edges increases. It detects only very high intensity edges which corresponds to the threshold values and creates discontinuities in the final edge images.

➔ Final images for different low and high threshold values:


(low,high) –(15,30)


(low,high) –(20,40)


(low,high) –(35,80)

(low,high) –(70,150)