Computer Vision - Homework 1

Ruchi Manikrao Dhore - W1652116

Monday 23rd January, 2023

1 Introduction

In this homework, we studied and built programs for filtering using Gaussian filtering, moving average filter and median filter, edge Detection using canny edge detector and Sobel filter, corner detection using Harris Corner detector with different parameter values, 2D convolution and 2D correlation.

2 Filtering

As part of this problem statement, filtering was applied on 10 images. The input was taken from each of the 10 classes from CIFAR-10 dataset. To accomplish this, Open-CV filtering library was used. Three types of filters were applied on all 10 images and filtered image outputs were plotted using matplotlib.pyplot library.

To apply the three filters, following methods were used:

Filter	Method
Gaussian	cv2.GaussianBlur(image, kernel size(ksize.width, ksize.height), sigma value)
Median	cv2.blur(image, kernel size(ksize.width, ksize.height))
Average	cv2.medianBlur(image, kernel size)

Table 1: Open-CV Filtering Methods

For all three filters, we have provided the kernal size of 5x5 with both width and height as 5. For Gaussian filter, we also specified the standard deviation for the X (sigmaX) and Y (sigmaY) directions. The value for sigmaY is automatically picked up as same as sigmaX. Hence, in the table we see only one parameter for sigma. Figure 1. Filtered Image Output represents the output of applying all three filters. For Gaussian filter, different sigma values were applied:

$$\sigma = 1, 5, 10, 50, 100 \tag{1}$$

Let's go through the observations for all three filters. Firstly, as we increase the value of sigma of the Gaussian filter, we observed that the image starts becoming more blur, appearing more smoother and removed the details of the original image. The higher sigma

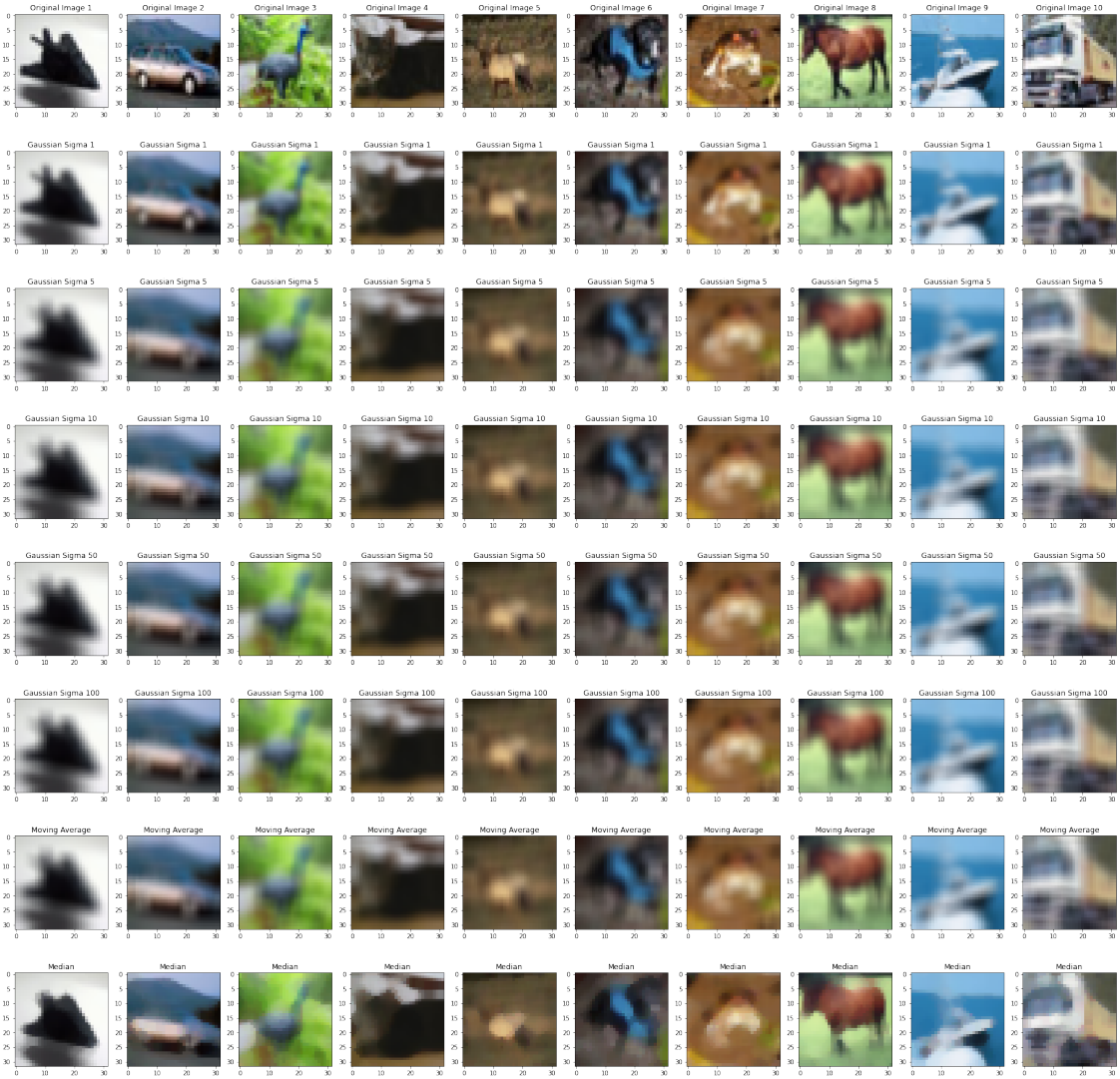

Figure 1: Filtered Image Output

value makes the pixels look more like its neighbouring pixels. Also, there was reduction in noise.

Secondly, the first major difference across all three filters is for the Gaussian filter (for higher sigma value) and moving average produce more blurred images in comparison to the median filter. In Median filter, we can still prominent edges. Gaussian filter with higher sigma value and moving average produce almost same set of blurred images.

Lastly, selection of the filter depends on the application. The filtered value for the central element in the Gaussian filter and moving average could be a value that does not exist in the original image itself. However with median filtering, the central component is always updated to some other pixel value in the image. If we don't want to lose the pixel values, median filtering would be preferred. If given a choice to choose among all filters, I would go with Gaussian filter since it gives us the freedom of how much blurriness we want in our images using the sigma parameter.

3 Edge Detection

As part of this problem statement, canny edge detector was applied on 10 images. Again, the input was taken from each of the 10 classes from CIFAR-10 dataset. To achieve this, Open-CV Edge detection library was used. Different set of threshold values were used to plot new edge maps. Additionally, Sobel filter was also applied on the same set of input images and new edge maps were plotted.

Table 3 lists the methods used for canny edge detector and sobel filter.

Detector/Filter	Method
Canny edge detector	cv2.Canny(gray, lower threshold value, high threshold value)
Sobel filter	np.sqrt(np.power(sobelx, 2) + np.power(sobely, 2))

Table 2: Open-CV Edge Detector Methods

Figure 2. Edge Detection Output represents the output for different threshold value for canny edge detector and sobel filter.

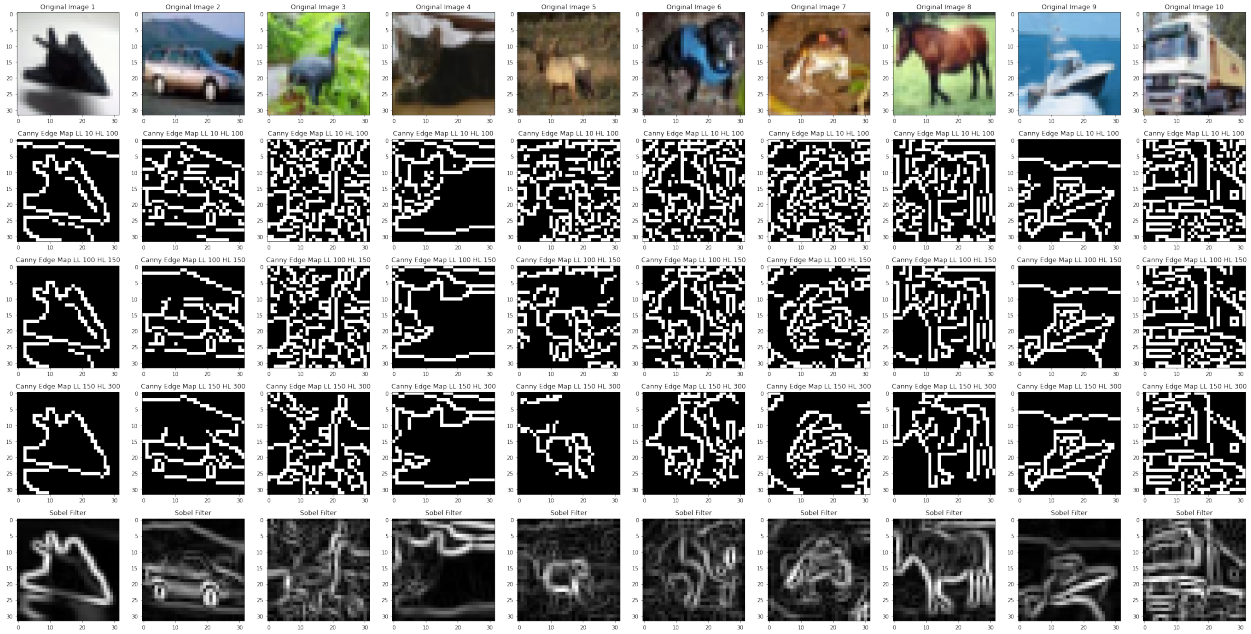

Figure 2: Edge Detection Output

Table 3 lists the threshold values that were used for this assignment.

Lower threshold	High threshold
10, 100, 150	100, 150, 300

Table 3: Threshold Values

Let's go through the observations. Firstly, the values for lower and higher threshold were gradually increased and the differences among them were kept different. For the first set of values (10, 100), all the edges of the object of the image are detected. For the second set of values (100, 150), we started losing certain edges of objects surrounded the main object. For the third set of values (150, 300), we start losing the edges of the main objects as well, for example in sample image 2 and sample image 5. However, the images where we don't have more objects or divisions in the image, the threshold values did not effect much, for example in sample image 1 and sample image 9.

Secondly, if we compare the edge maps obtained using Canny edge detector and sobel filter, we observe that the edges produced in sobel filter are not thick and smooth. The edge maps produced using Canny edge have evident edges. Though, sobel filter also produces image with edges it is not prominent.

Lastly, I would prefer canny edge over sobel filter to detect edges. The main reason is with the right set of threshold values, we can detect the needed object from the image. With canny edge, there is good reduction in noise and the image appears more smooth in comparison to sobel filter.

4 Corner Detection

As part of this problem statement, Harris corner detector was applied on 10 images. The input was taken from each of the 10 classes from CIFAR-10 dataset. To achieve this, Open-CV library was used. Different values were provided for the harris corner parameter.

For the harris corner detector, different parameter were provided like 0.04 and 0.07. Table 4 lists the method used for the same.

Detector	Method
Harris corner detector	cv2.cornerHarris(gray, 2, 3, harris$_{p}arameter$)

Table 4: Open-CV Edge Detector Methods

Figure 3. represents the output for different parameter values.

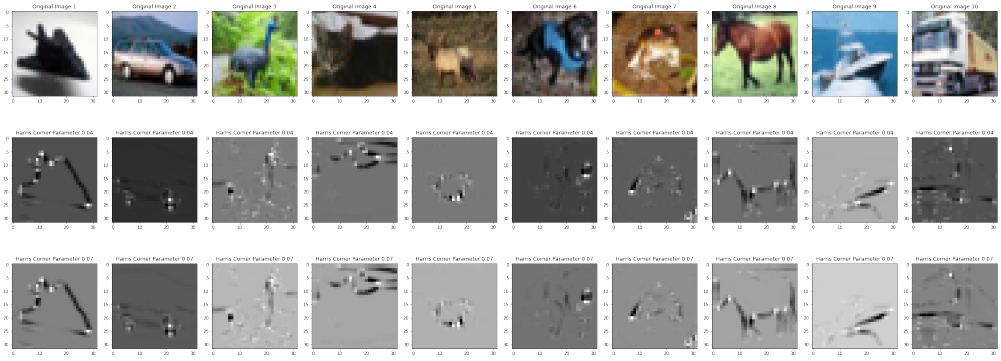

Figure 3: Harris Corner Detector Output

Let's go through the main observation. The main difference for different harris corner parameter values is the gray component reduces for higher values of the parameter. As we can see in the output, for 0.04 parameter value, the image is darker in comparison to the parameter value 0.07.

5 Convolution and Correlation

As part of this problem statement, 2D convolution and 2D correlation was applied to an image from CIFAR-10 dataset. To achieve this, Numpy library was used. The kernel dimension used for the implementation was.

Figure 5. represents 2D convolution and 2D correlations.

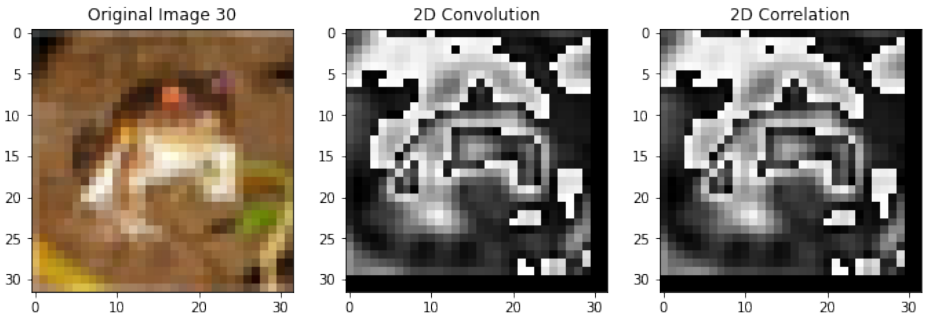

Figure 4: 2D Convolution and 2D Correlation

Let's go through the observation. The main difference between them is there is reduction in noise and more smoothness in 2D convolution as compared to 2D correlation.

6 Conclusions

In conclusion, we have implemented different techniques like filtering, edge detection, corner detection, convolution and correlation. Based upon the application and requirements, specific technique can be utilized.