ELEN644 Homework 1 Report

Justin Goh

January 23, 2023

Introduction

The images used for this assignment were taken from the CIFAR-10 dataset via Tensorflow (see Figure 1). Code was written in Python.

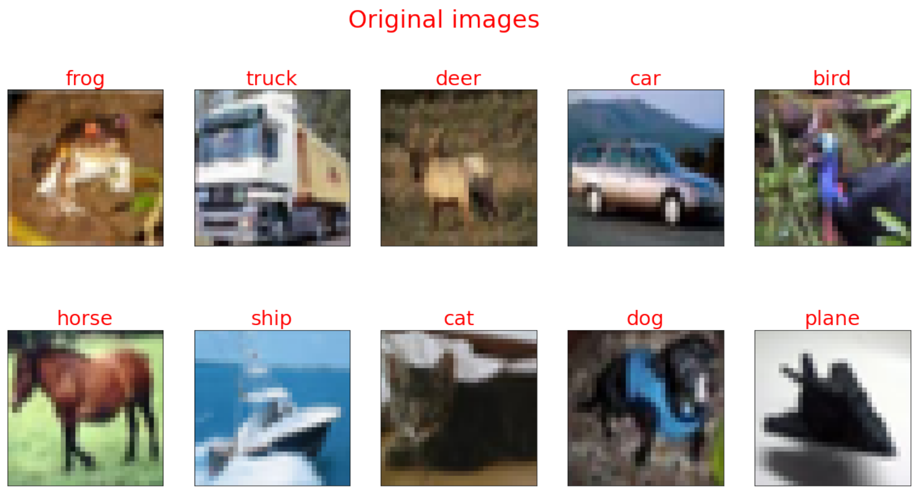

Figure 1: Loaded images from CIFAR-10

1 Problem 1: Filtering

1.1 Questions

1. Apply Gaussian filtering for sigma = 1, 5, 10, and 100 to each of the 10 images from the CIFAR-10 dataset. Plot the filtered image. What are the observations as the sigma values are increased?

2. Apply a moving average and median filter on the images and plot them. What are the differences between the filtered image of the Gaussian, Median and Mean filter? Which do you prefer?

1.2 Analysis

For this comparison, I will compare a Gaussian filter with sigma value of 5. Average and Median filter will have a kernel size of 5. Observations are as follows:

Gaussian: All images are significantly blurred. It is hard to tell the content for some of the images (i.e deer, car, cat, dog) after applying the Gaussian filter. See Figure 2.
Average: All images appears to also have been blurred to a similar degree as the Gaussian filter. It was also observed that when the kernel size of the average filter was increased to 10, the images were significantly more blurred than images filtered with a Gaussian filter of sigma value 10.
Median: The median filter blurred the images however the edges of the image content was better preserved than that of the Gaussian or average filter. As the filter size increased, the details of the image content diminished.

If my goal was to smooth an image, I would choose a Gaussian filter. If I cared more about preserving the edges in an image, I would choose a median filter.

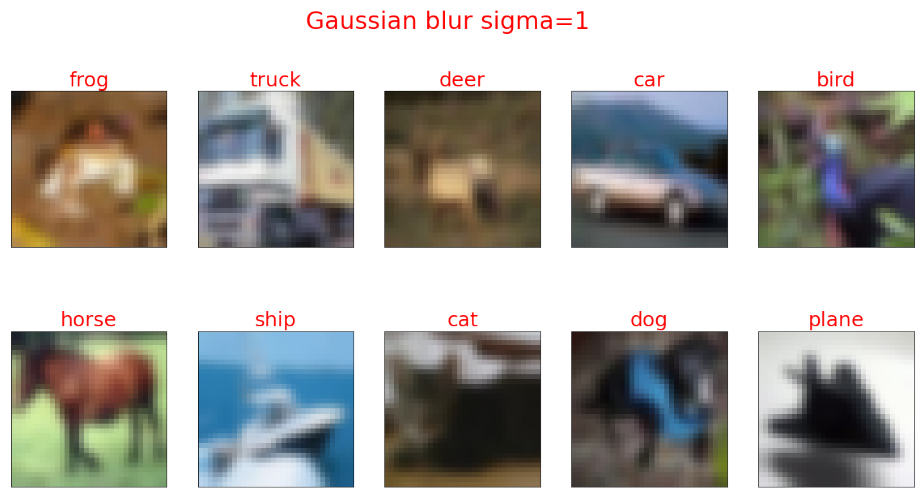

Figure 2: Gaussian filtered, σ=1

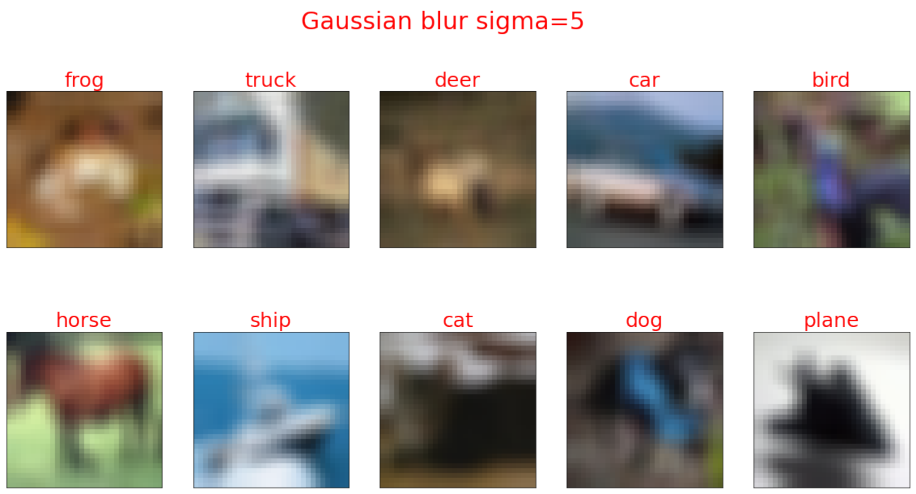

Figure 3: Gaussian filtered, σ=5

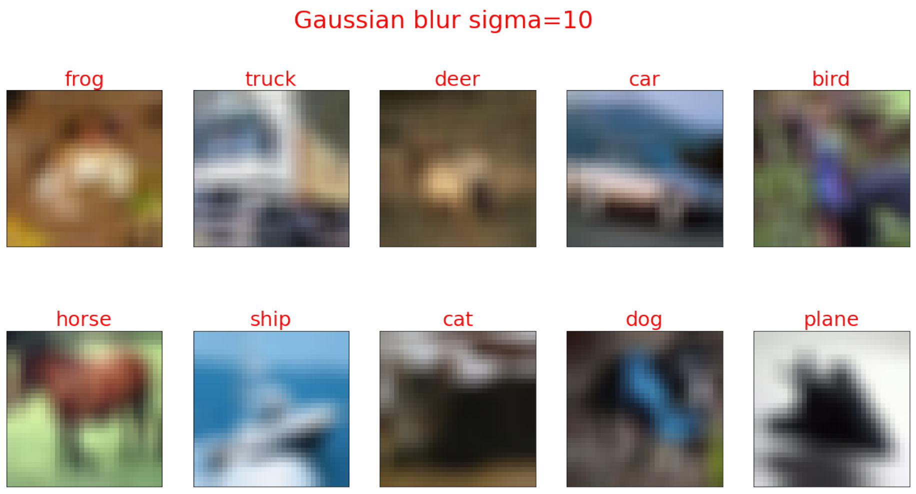

Figure 4: Gaussian filtered, σ=10

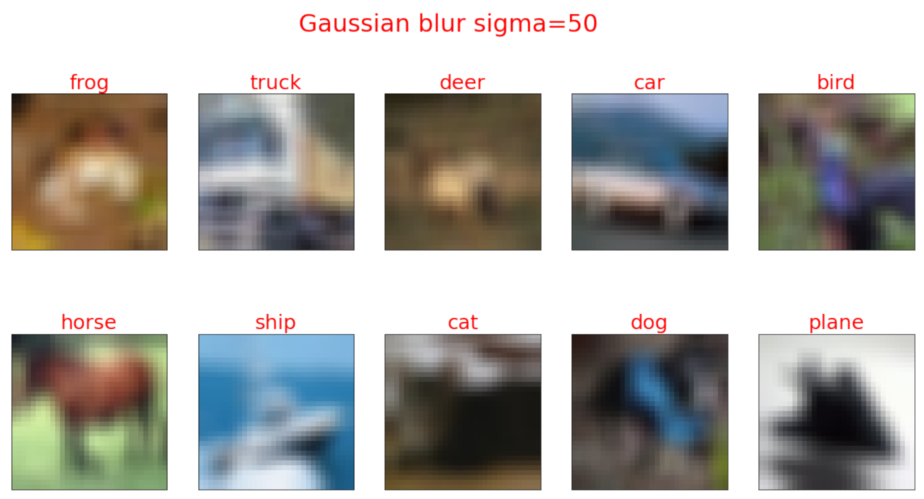

Figure 5: Gaussian filtered, σ=50

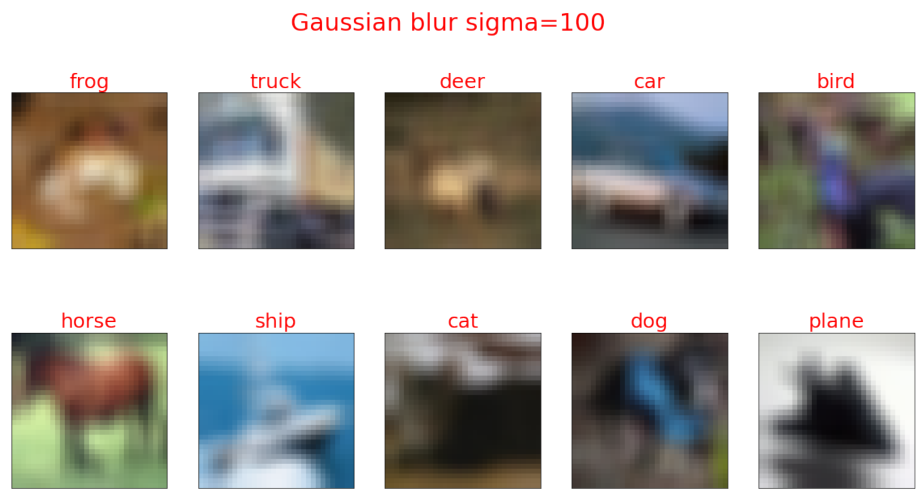

Figure 6: Gaussian filtered, σ=100

2 Problem 2: Edge Detection

2.1 Questions

1. Apply Canny edge detector on each of the 10 images and plot the images. Try increasing and decreasing the default threshold values and plot the new

image for each case. Explain observations from changing the thresholds.

2. Derive the edge maps using a Sobel filter. Compare the edge maps using a canny edge detector and Sobel filter. What are the differences? Would you prefer a canny edge or Sobel filter?

2.2 Analysis

2.2.1 Part 1: Canny Edge

For Canny edge detection algorithms, the final step in determining if an edge exist is done through Hysteresis thresholding. It uses 2 threshold values a maximum and a minimum. Pixels above the maximum threshold are considered strong edges. Pixels below the minimum threshold are considered non-edges. Any pixels between the minimum and maximum thresholds that are connected to a pixel that is considered a strong edge are considered weak edges. They will still appear in the final output image.

When implementing the Canny Edge detection algorithm, it was observed that as we decreased the minimum threshold, there are more edges in the output image. This is because weaker edges are preserved the lower the minimum threshold. I start with a minimum and maximum threshold of 100 and 200 respectively (Figure 7). I then decrease the minimum threshold to 50 and then 5 (Figure 8 and 9 respectively).

If we increase the maximum threshold, we will notice that less edges appear in the output image (Figure 10). This can be attributed to the fact that less strong edges exist. Increasing this value too high will result in little to no edges being preserved (Figure 11).

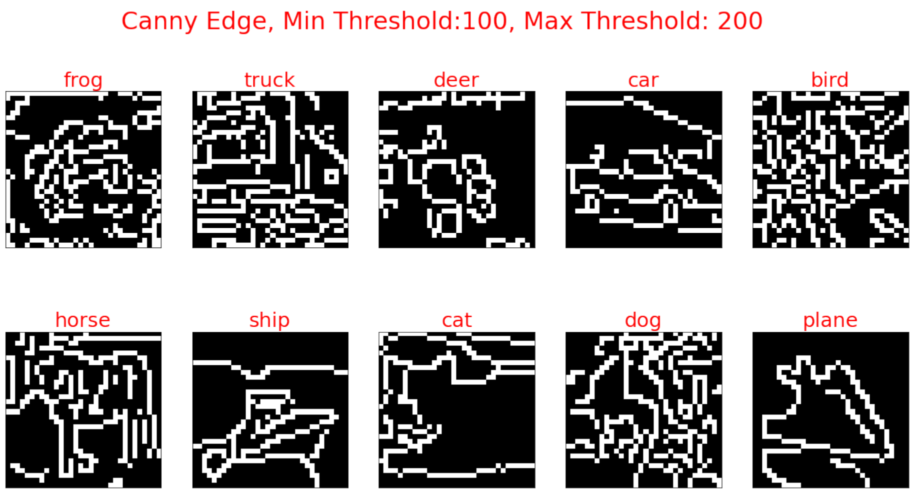

Figure 7: Baseline Canny Edge Detection algorithm

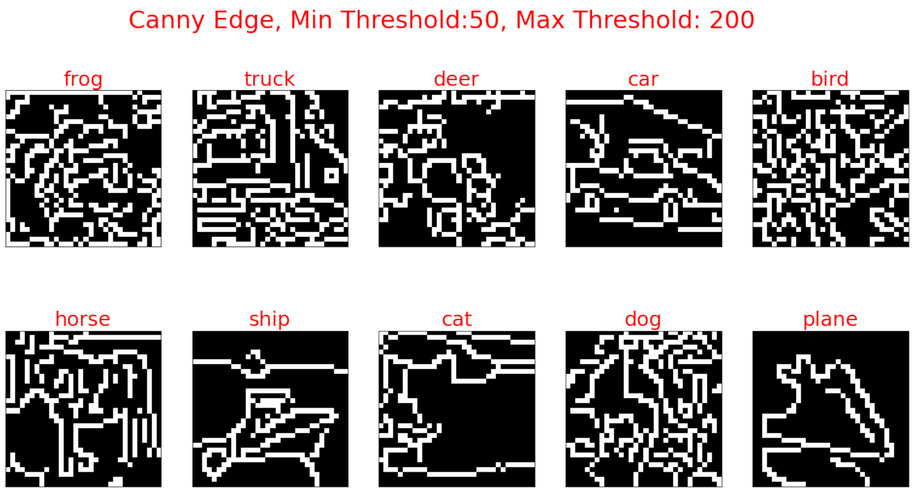

Figure 8: Canny Edge, Min Threshold = 50

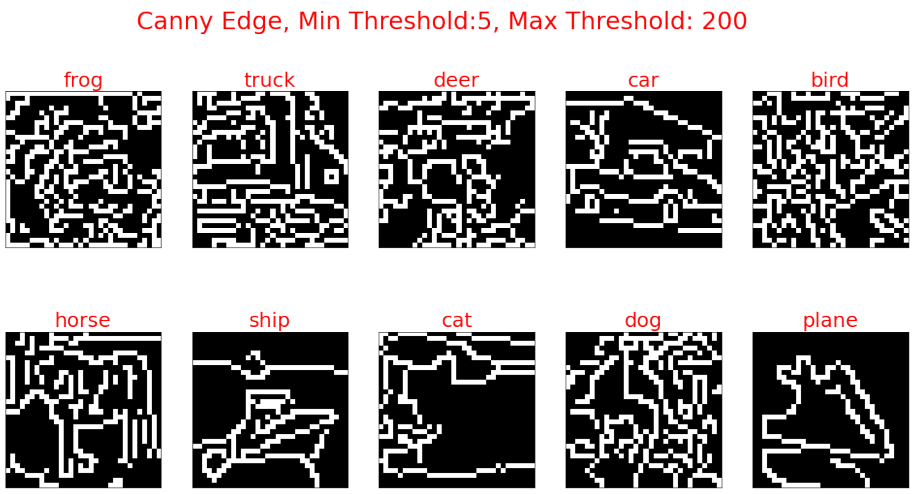

Figure 9: Canny Edge, Min Threshold = 5

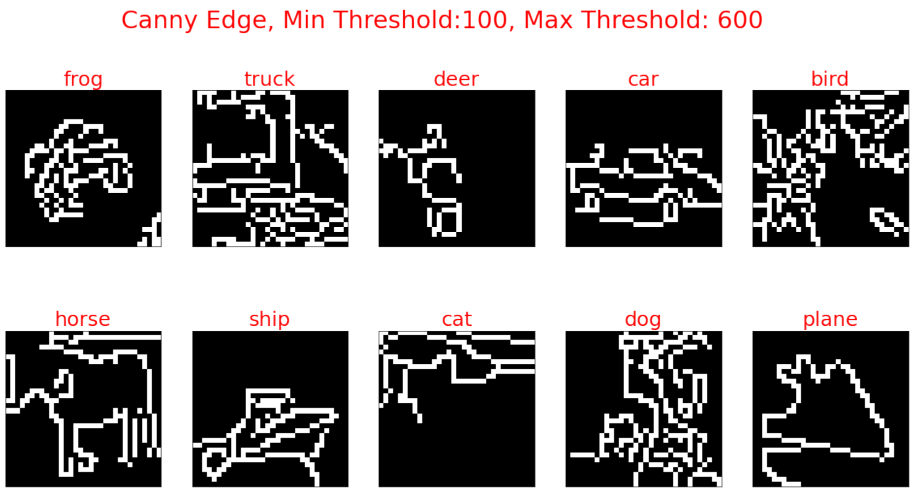

Figure 10: Canny Edge, Max Threshold = 600

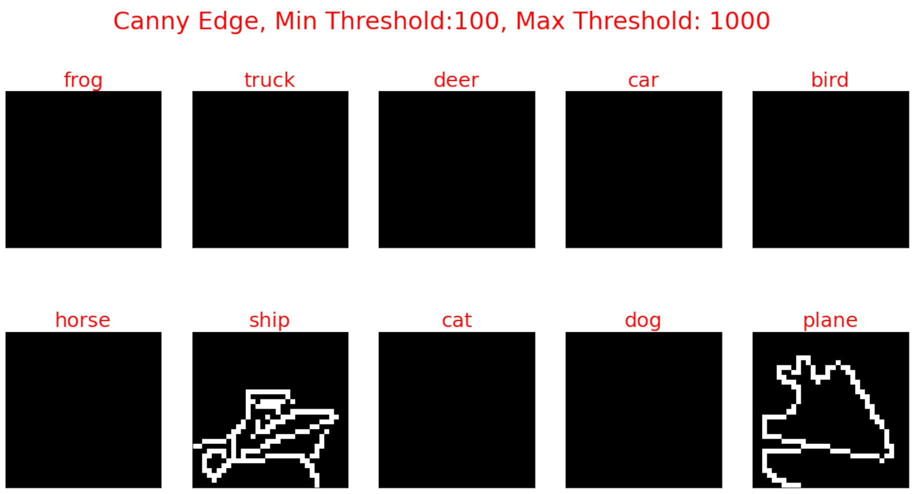

Figure 11: Canny Edge, Max Threshold = 1000.

2.2.2 Part 2: Sobel Filters and Canny Edge comparison

When using the Sobel filter, I observed that more edges from background elements of the picture was preserved. When comparing the performance between the Sobel and Canny Edge, the Sobel appeared to perform better if there was **not** a lot of background elements in the image. If there were a lot of background elements, the Sobel Filter produced an output that looked somewhat noisy (see image for frog, truck, deer, bird, dog in Figure 12). In this instance, the Canny Edge performed better than the Sobel filter. This could be due to the fact that the Sobel filter can detect edges in different orientations. This also makes it more susceptible to noise.

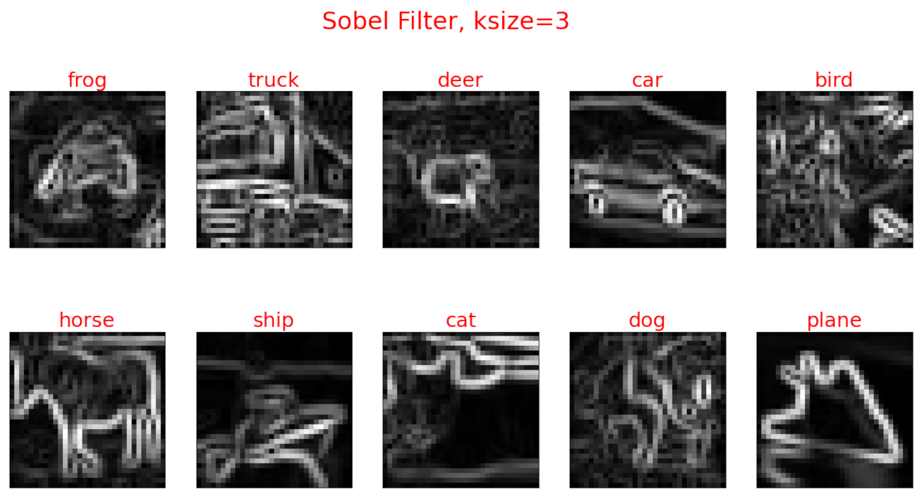

Figure 12: Sobel Filtered Image

3 Problem 3: Corner Detection

3.1 Questions

Apply Harris Corner detector on each of the 10 images. Use parameter as 0.04. Repeat using parameter 0.07. What are your observations when parameter value increased? What are the effects?

3.2 Analysis

When adjusting the parameter from 0.04 to 0.07 when using the Harris Corner detector function, it was observed that using parameter 0.04 resulted in **more** corners being detected than using a parameter value of 0.07.

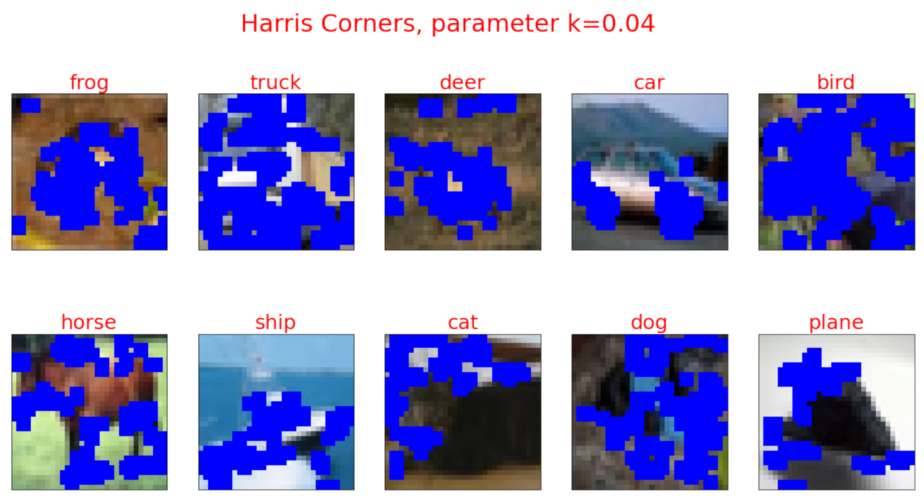

Figure 13: Parameter=0.04

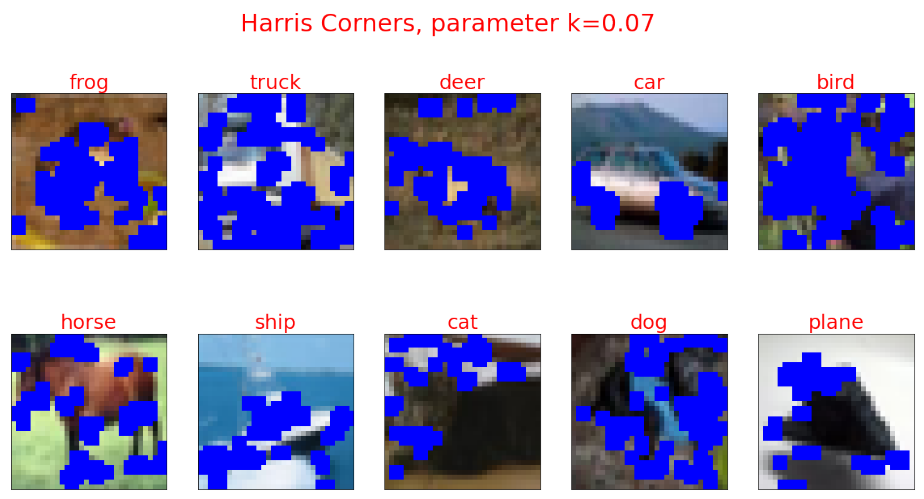

Figure 14: Parameter=0.07

To make this point clear, the parameter was adjusted to 0 and 0.2. Having the parameter equal to 0 resulted in many false corners being detected while having the value of 0.2 resulted in significantly less corners detected.

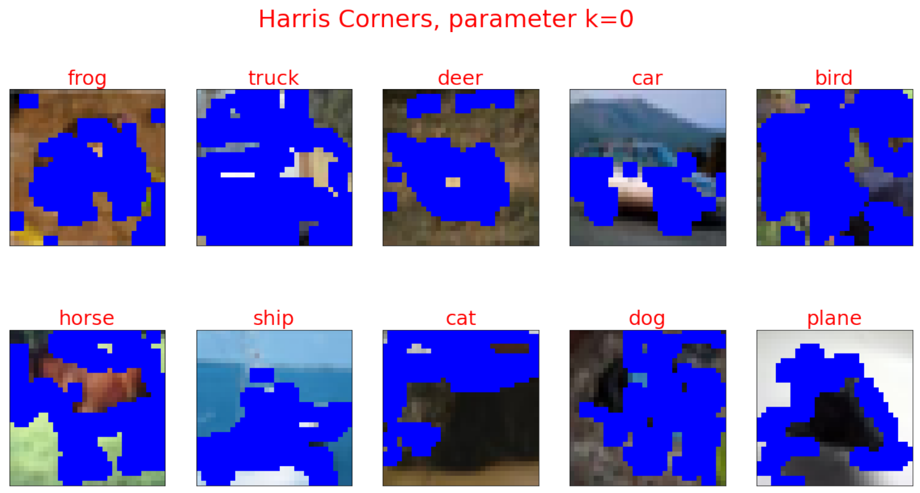

Figure 15: Parameter=0

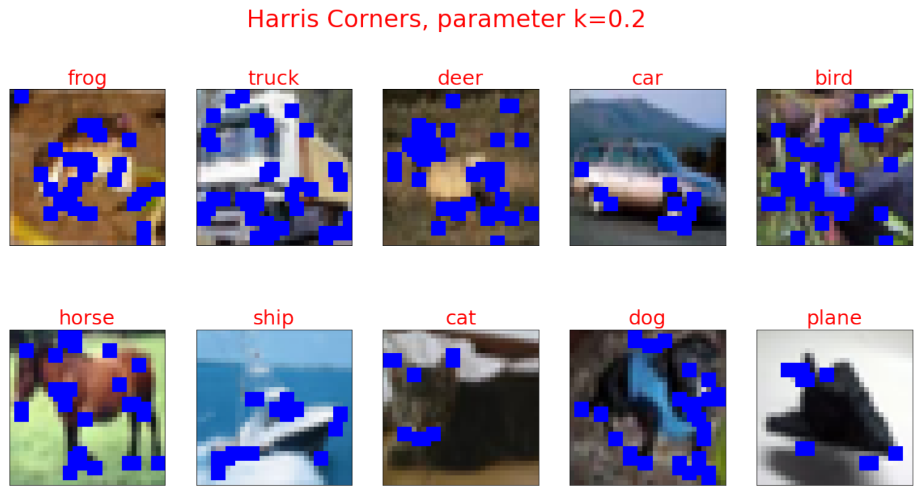

Figure 16: Parameter=0.2

4 Problem 4: Convolution and Correlation

4.1 Questions

Implement 2D-Convolution and 2D-Correlation to an image. Specify kernel size. Plot outputs and explain differences observed.

4.2 Analysis

The kernel size used was a 3x3 box filter. It should be noted that the main difference between a 2D-convolution and 2D-correlation operation is that for a **2D-convolution**, the kernel is flipped horizontally and vertically before performing the element-wise operation. Before applying the 2D-convolution and 2D-correlation operation, the input image was converted to grayscale first.

When applying 2D-convolution and 2D-correlation, it was observed that the output image for both operations was the same. This is because the box filter is a symmetric kernel, which means that the output will be the same regardless if the kernel is flipped or not.

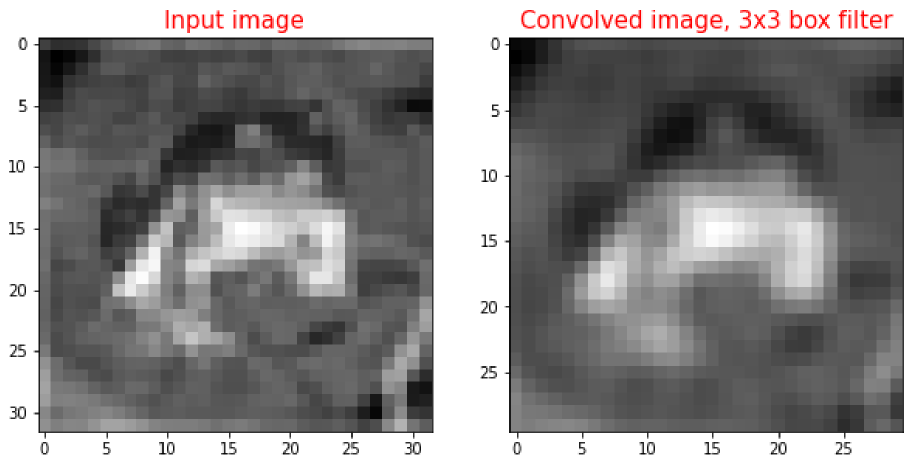

Figure 17: Output of 2D-Convolution

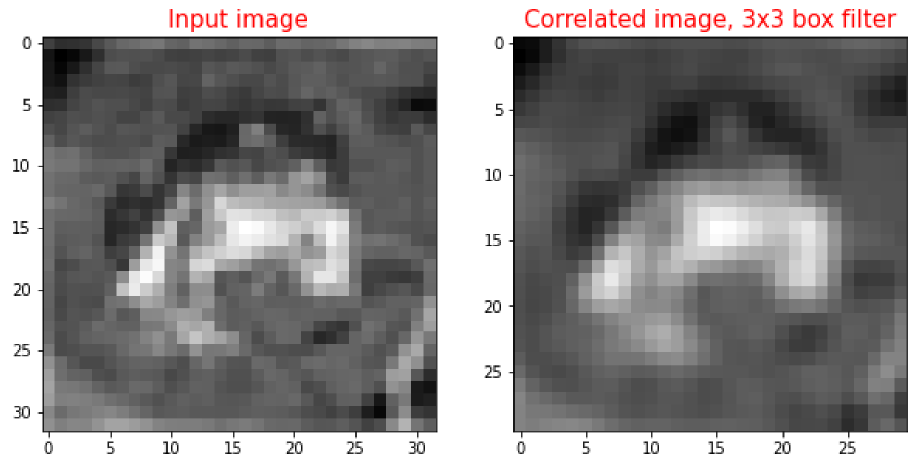

Figure 18: Output of 2D-Correlation