Homework 2 CMSC426

Andrew Lee
March 5, 2021

1 Introduction

The assignment for this project was to perform filtering and edge detection on a selected image. Specifically, to create a Gaussian kernel and a method that performs image convolution with a given filter.

2 Filtering

2.1 Gaussian Kernel

The Gaussian Kernel is a function of n-d number of variables and whose width is determined by a given σ. It is defined in 2-D as

$$f(i,j) = \frac{1}{2\pi\sigma^2} e^{-\frac{i^2 j^2}{2\sigma^2}}$$

We can use this kernel as a filter to achieve Gaussian blurring, which is useful for the upcoming edge detection. However, it is important to remember to normalize the values in the Gaussian kernel so that they add up to 1.

2.2 Convolution

We convolve an image matrix with a filter matrix. For pixels that would look for surrounding elements that are outside the bounds of the matrix, the out of bounds elements are ignored and are not added to the sum for the new pixel value.

2.3 Filters

Applying the h1,h2 and h3 filters to the image results in the images seen below. The first one affects each pixel by increasing the intensity of the center pixel while also decreasing it based on the intensity of the neighboring pixels. The second filter does the same thing except only in the vertical direction, and the third filter does the same in the horizontal direction. As a result, all the images are varying amounts more gray than the original image and have less contrast.

3 Edge Detection

3.1 Noise reduction with Gaussian filtering

We use the gausskernel(sigma) method to create a Gaussian kernel with standard deviation sigma and use it as a filter with the image, using myfilter(I,h) to convolve the 2 matrices. the resulting image can be seen here:

3.2 Finding image gradients

Using the given Sobel filters as filters for the Gaussian filtered image, we get the derivatives, denoted as dx and dy in the code. To get the magnitude of the gradient, we use np.hypot(dx,dy) to get our matrix of gradient magnitudes. Afterwards, we normalize these values by dividing them by the maximum value in the matrix and then multiplying them by 255 to fully span the possible values of pixel intensity. To get the direction of the gradient, we use np.arctan2(dy,dx) to get our angle in radians with a range of [-π, π]. we then round these values to the nearest $\frac{\pi}{4}$ and convert to either $0°, 45°, 90°$, and $135°$ for later use.

3.3 Edge thinning/Non-maximum suppression

For each pixel, we check the direction of the gradient and assign certain pixels in a 3x3 area to check accordingly. The pixels we need to examine can be determined from the project specifications. However, because the specifications list standard x,y coordinates, the appropriate adjustments were made so that the correct pixels were examined. If an examined pixel were to go out of bounds, it is ignored and not compared to the main pixel. If the center pixel has the greatest magnitude compared to the other pixels, and it is greater than the t_low threshold, the pixel is kept, and we set the value of the pixel to its magnitude. Otherwise, the value of the pixel is set to 0.

3.4 Hysteresis Thresholding

Some of hysteresis thresholding was done in edge thinning, by filtering out elements whose magnitudes are less than t_low. After this we use label() with the default structure([[0,1,0],[1,1,1],[0,1,0]]) to group up the remaining connected pixels. Then, we cycle through the label

values, setting the pixels with each label values to 1 if the label contains a value above t_high, or 0 if it does not.

3.5 Testing

Various images with different sigma, t_low, and t_high values can be seen in the attached files as this text editor could not contain the amount of images. I noticed that increasing the sigma would cause the edges to become less sharp, and areas with a high density of edges would have less with high sigmas. Increasing the t_low results in the length of some of the edges being shortened. Increasing the t_high removes many patches of edges that have weaker gradient values. Increasing this value is the most useful for removing noise.

4 Conclusion

The part in the project specifications where it listed standard x,y coordinates instead of matrix coordinates had me stuck for a while. I also had issues where I forgot to make a deep copy of the image when I was working with it, resulting in the original image being repeatedly altered whenever an action was done to it. Normalizing the magnitude matrix was also area of difficulty for me, as I didn't know I had to do this until I was far into the project.

Figure 1: Original Image

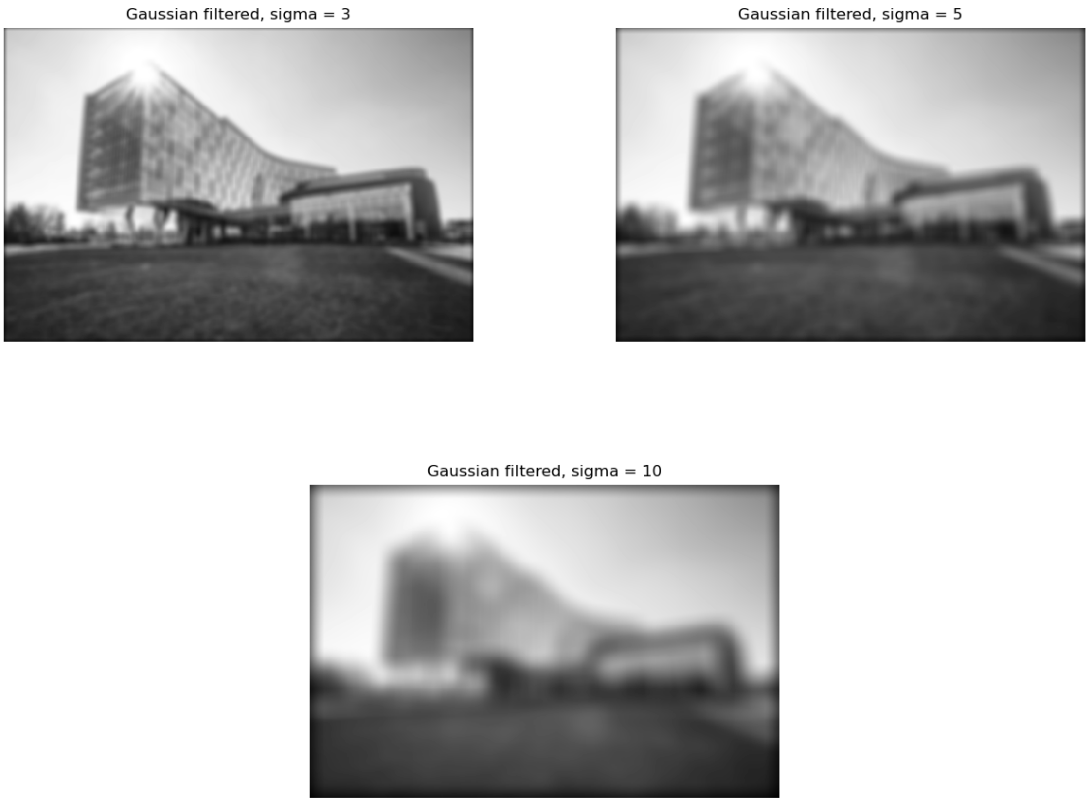

Figure 2: Image with Gaussian Filtering with sigma 3, 5, and 10

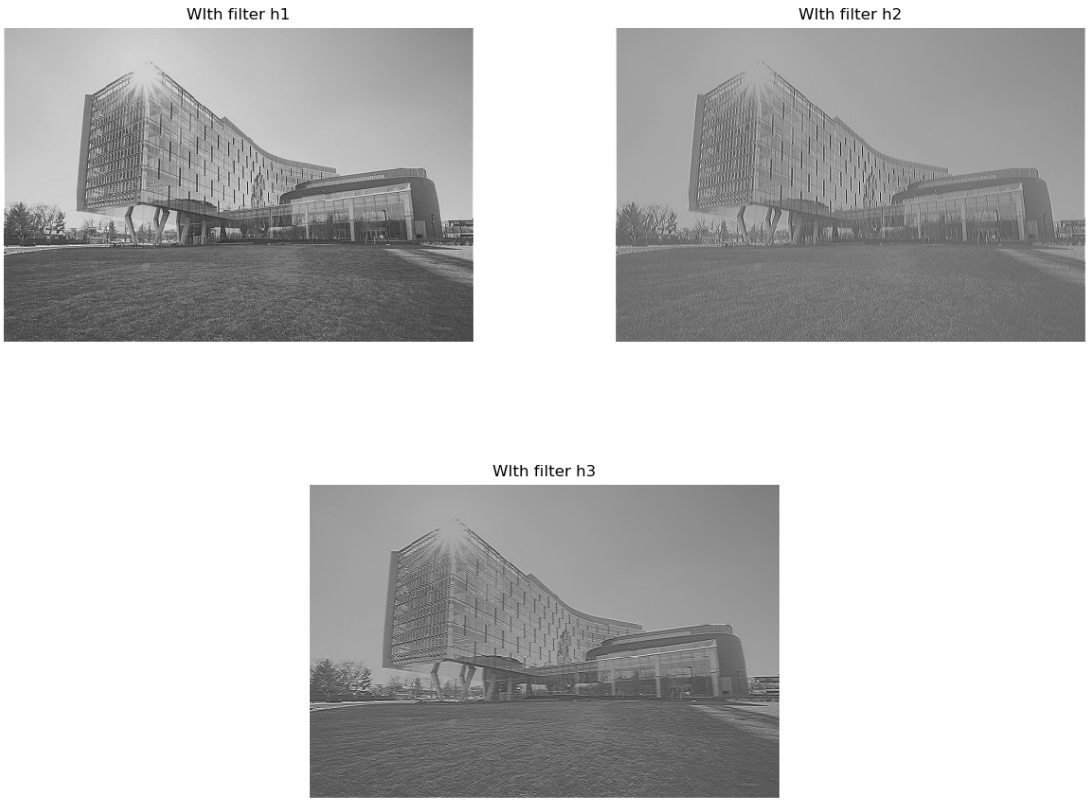

Figure 3: Image filtered with filters h1, h2, and h3