Jonathan Pearce

260672004

Comp 558

October 4, 2018

Assignment 1

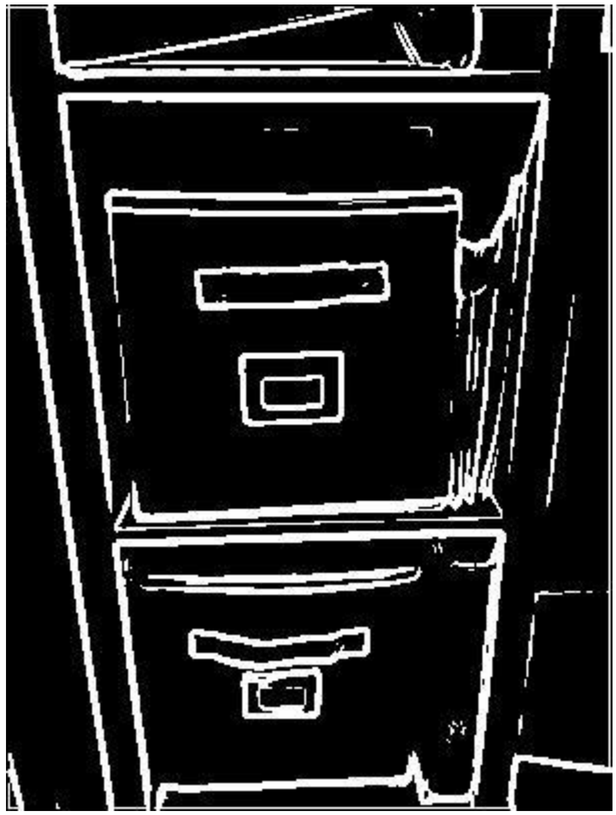
1. Code Submitted
2.

Gray scale, green channel image (303 x 403)

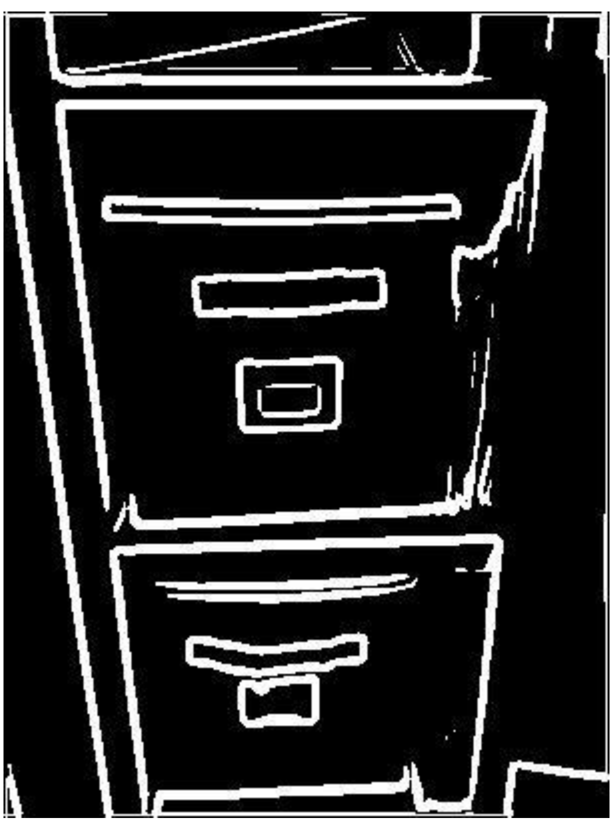
Gradient Images (Gaussian Filter, N=5, σ=1)

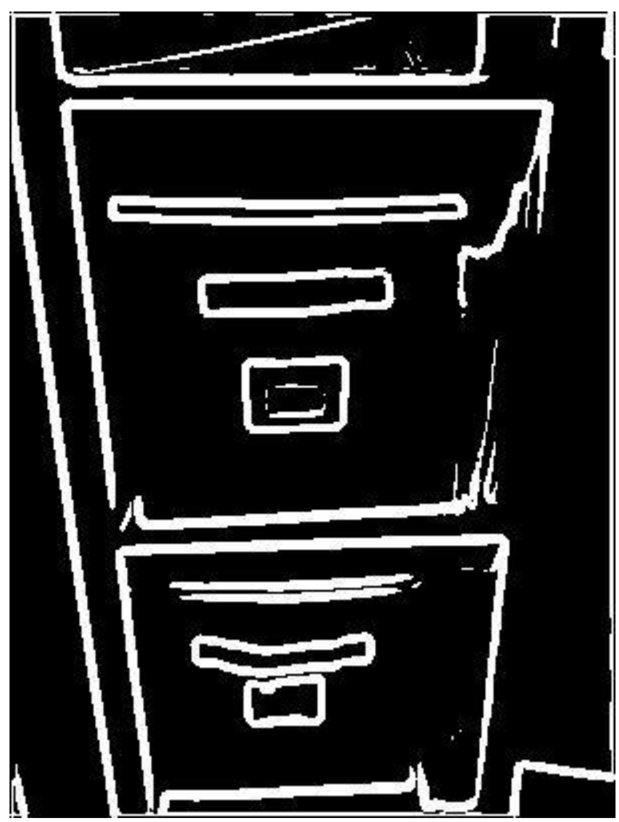
Gradient Images (with intensity x3)

Edge Position Binary Image (τ = 0.525)



Edge Position Binary Image with larger sigma value for Gaussian Filter (left: σ=2 and right: σ=4)





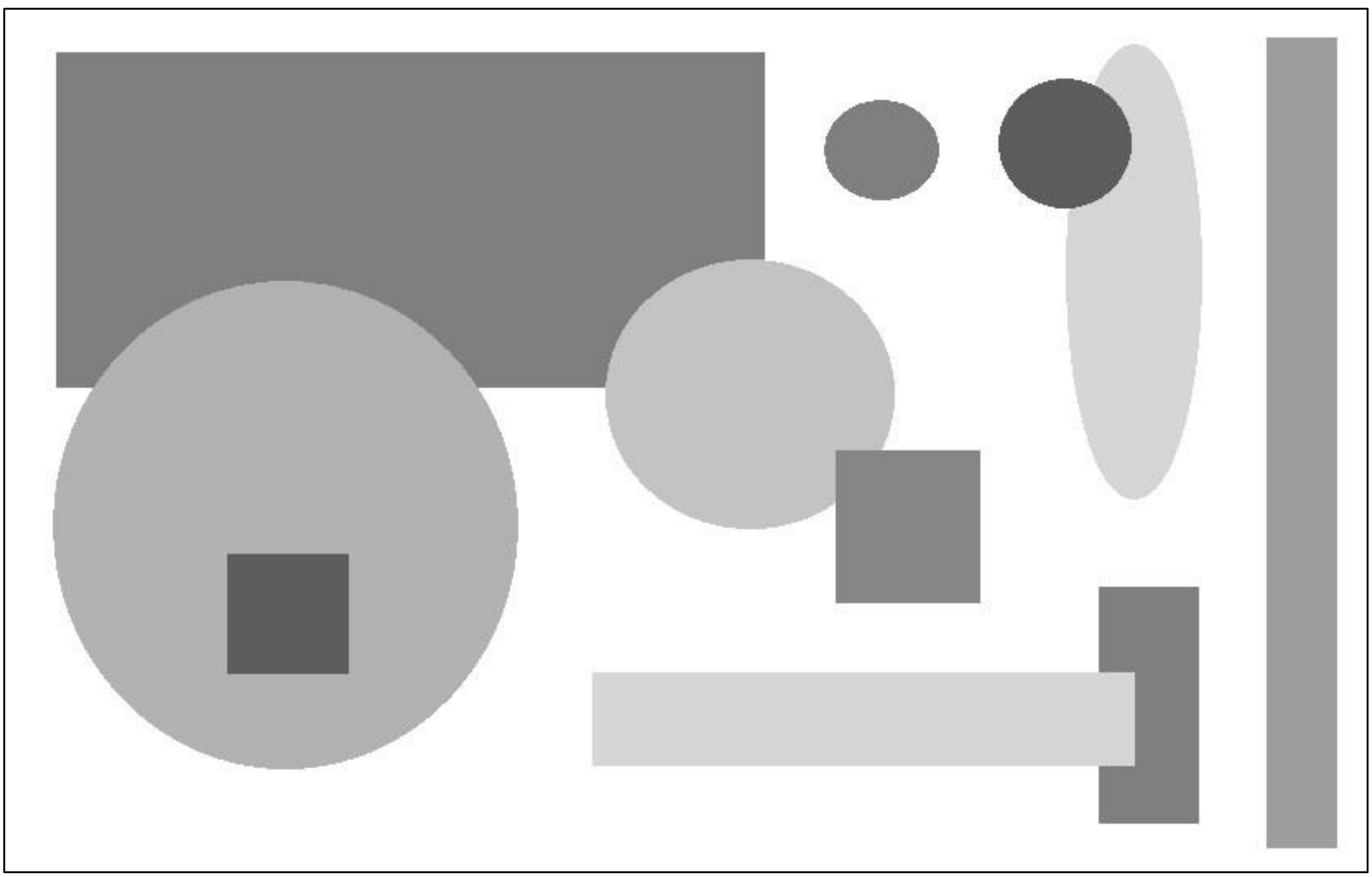
2. The difference between the edge position images is as σ increases the number of edges found and the detail of the edge image decreases. The first image (σ=1) is able to pick up many subtle edges such as the ring of my binder at the top of the image and the edges of my papers and folders found on the right side of the shelves, the last image (σ=4) fails to distinguish many of these edges. As you increase σ the filter becomes more uniform and the convolution creates a more blurry image (greater smoothing/loss of detail). This translates into neighbouring pixels having more similar intensities even at edges. Thus edge detection begins to fail more when σ is raised and the threshold τ remains constant.

3.

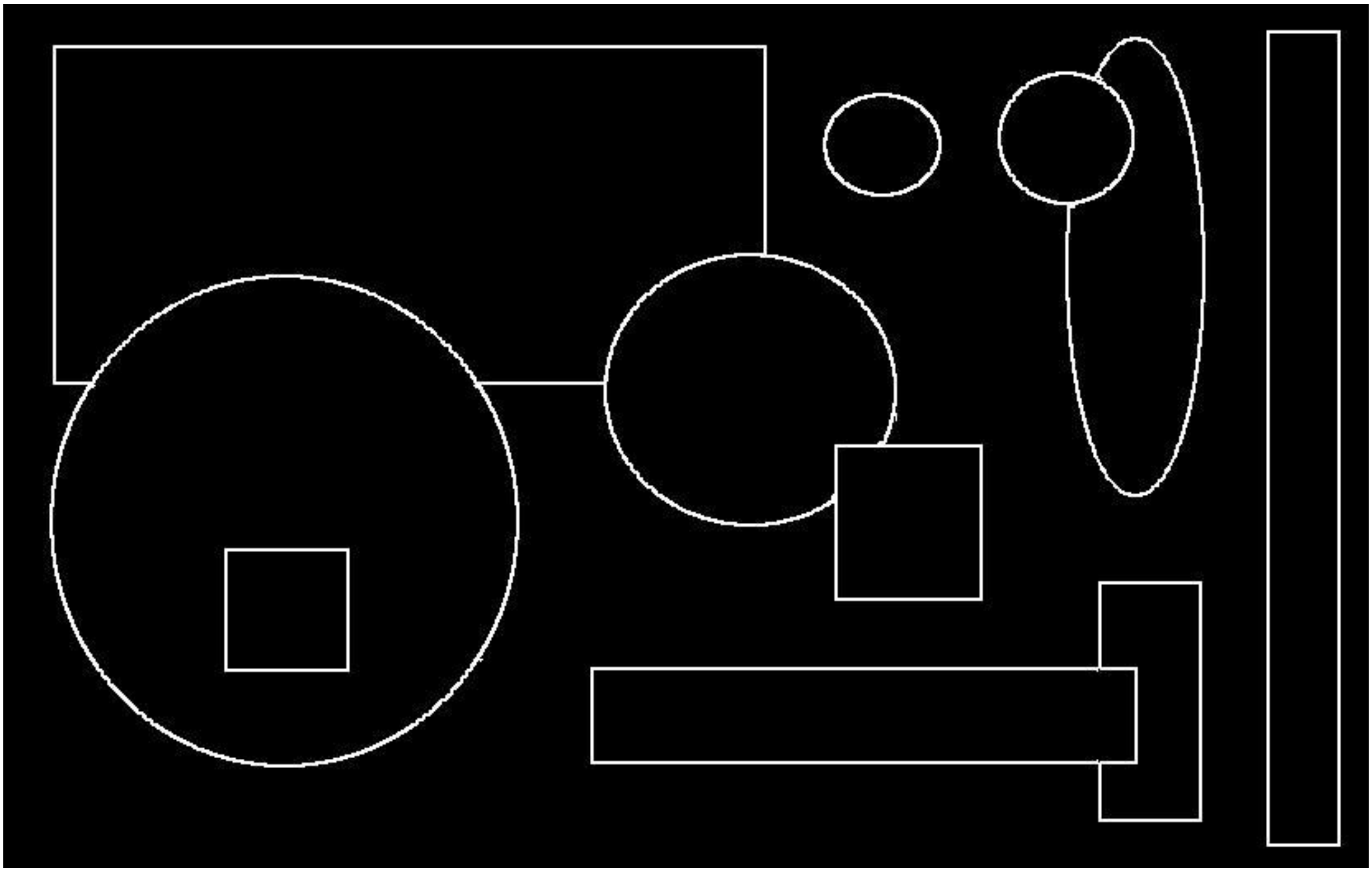
Note 1: My implementation of the Laplacian of a Gaussian had the same assumptions as Question 1A. I assumed N to be odd and the filter to be square (N x N).

Note 2: In doing this question I found that my implementation was detecting far too many pixels as having zero crossings, and the edge position images were very noisy (this shows in the second image where σ=1). I decided to add a threshold step after the convolution and before the neighbour comparison where any pixel with absolute value less that 1e-3 was changed to 0. The idea of this threshold was to account for floating point precision, in order to reduce the number of false positive edge pixels being detected.

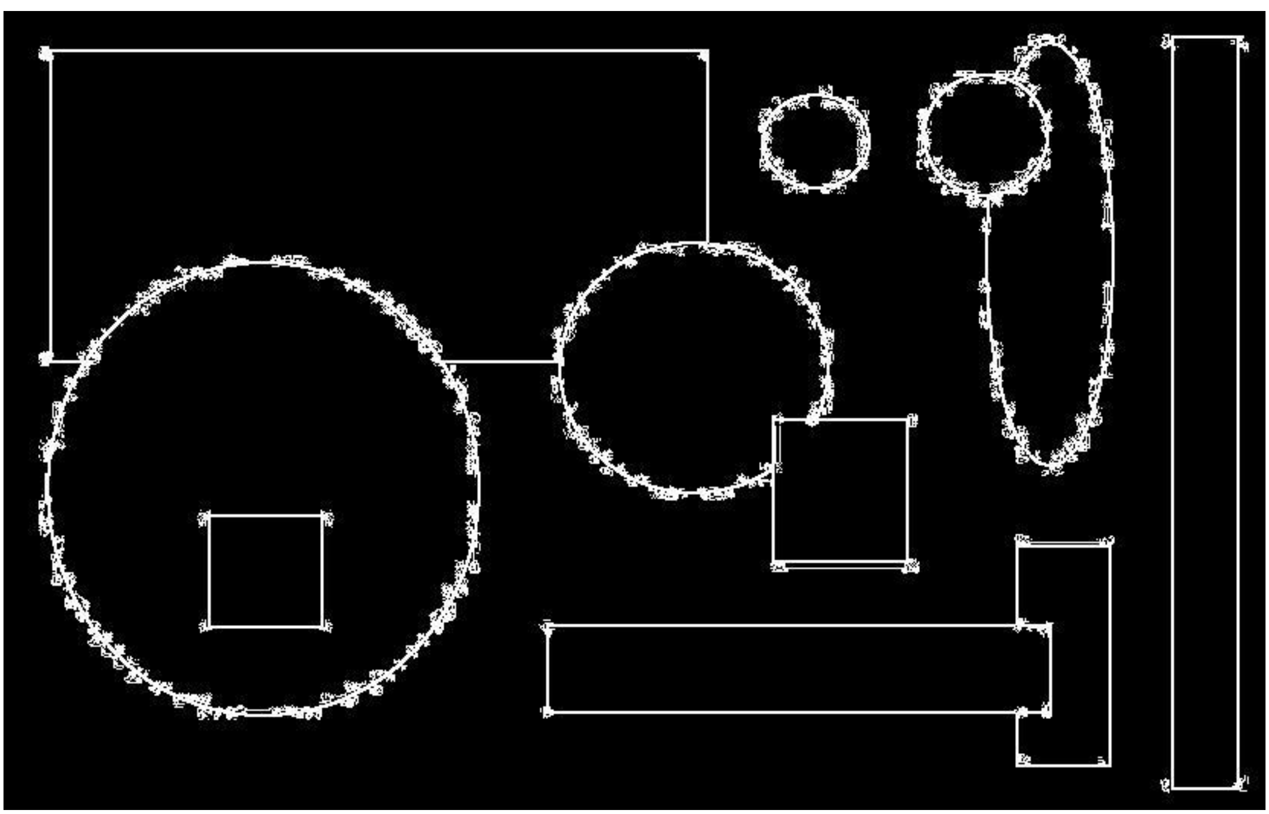
Original Image (border just to show image boundaries)



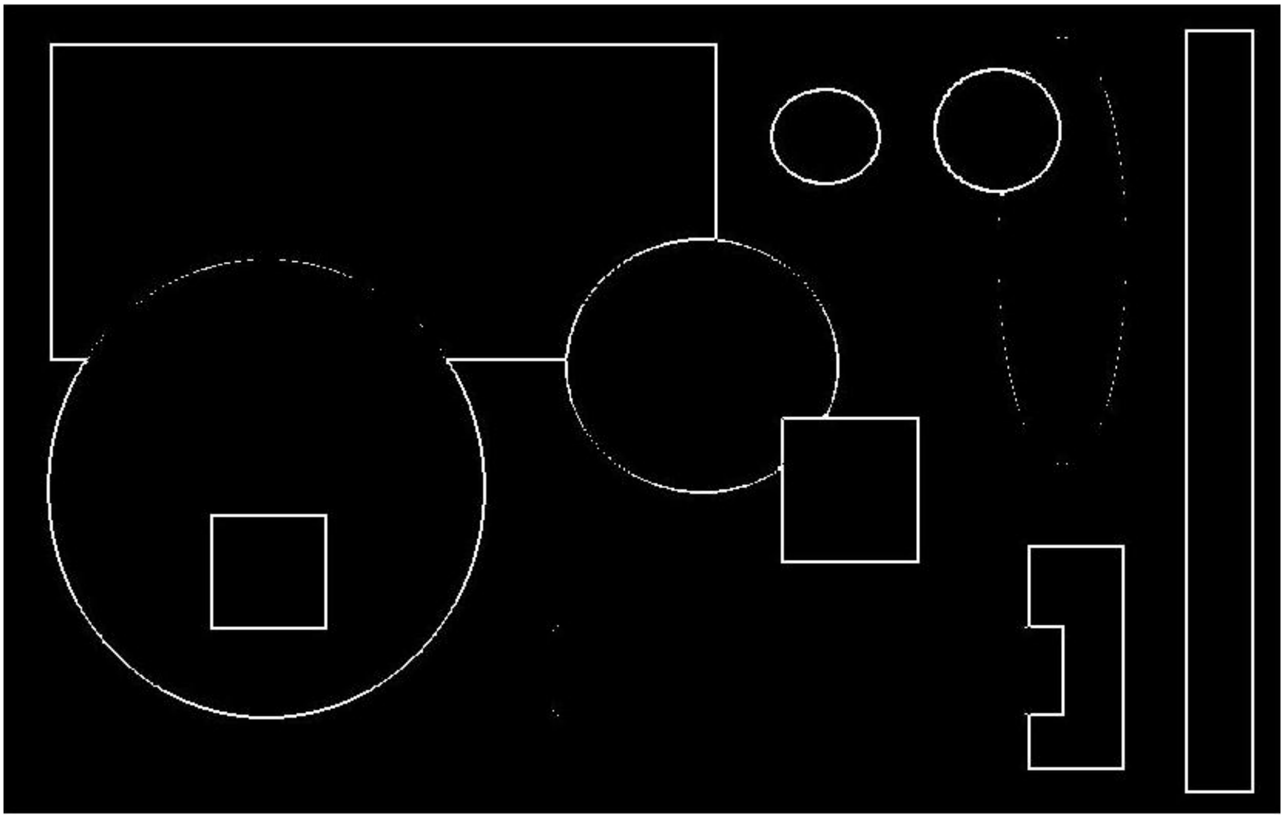
Zero Crossings (Laplacian of Gaussian Filter, N=5, σ=2)



Zero Crossings with alternative value of sigma (σ=1)



Zero Crossings with alternative value of sigma (σ=3)

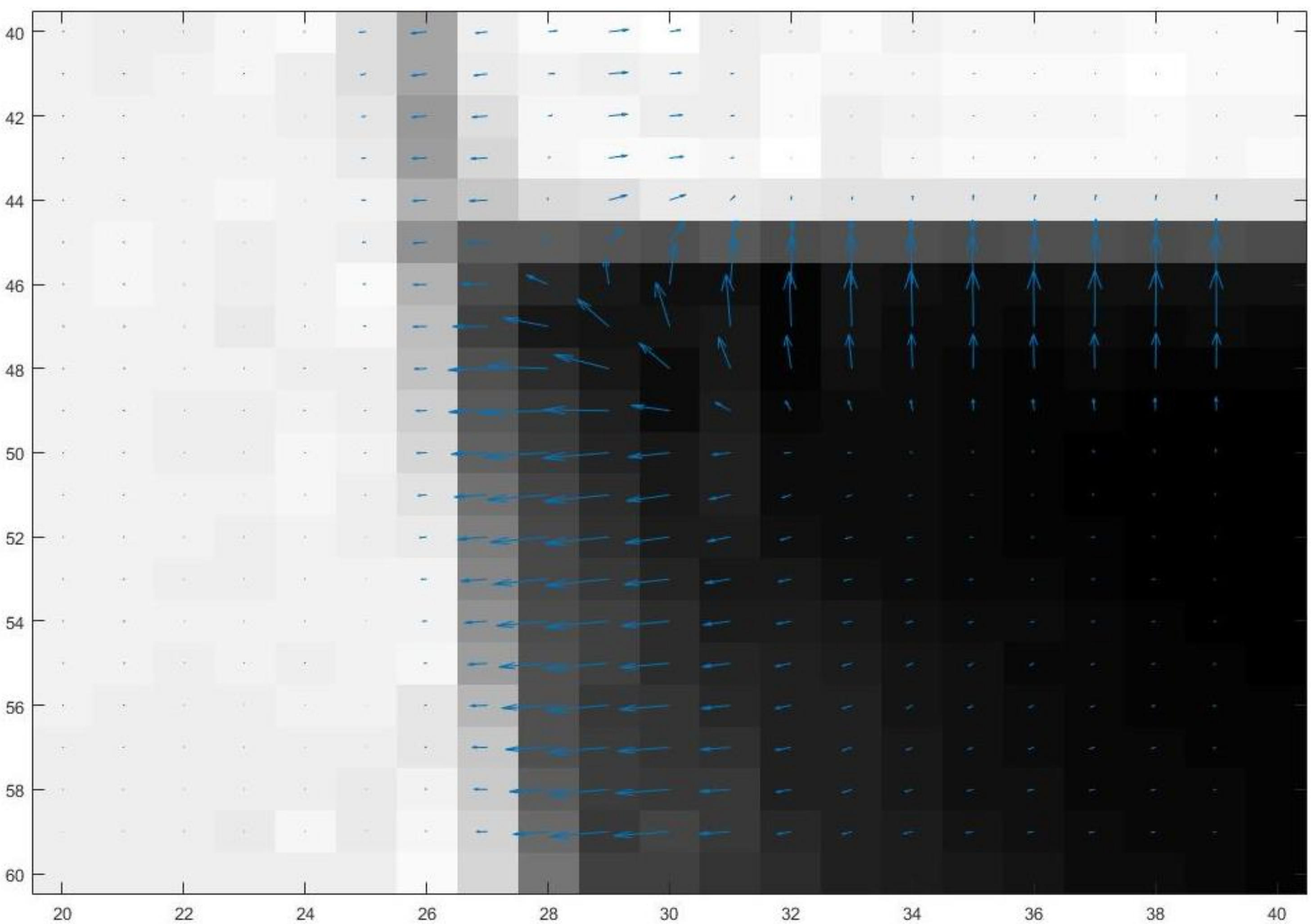


For σ=2 the edges (zero crossings) were very well defined and there were only a few pixels of noise found at locations where two discs intersected (T junction). When σ was lowered to 1, the edges became quite noisy, especially on the discs and a far greater number of (false) zero crossings were found. The decrease in sigma creates a greater oscillation in positive and negative pixels values found after convolution which leads to pixels not on the true edge being marked as edge pixels. Straight edges seemed to avoid this problem successfully however the corners of the rectangles and all of the discs were susceptible to this problem. In contrast when σ is brought up to 3, the oscillation in positive and negative pixels values found after convolution is reduced so greatly that with my threshold in place the shapes with light grey colouring fail to be detected anymore. The values found around the light shapes become so small that it is difficult to detect whether it is an edge or just noise. The most interesting part being the overlapping rectangles in the lower right corner, the horizontal rectangle is lost completely except for the overlapping section. The contrast between the light grey part overlapping the dark rectangle created enough contrast to allow those edges to be found, however the remainder of the rectangle became too similar to the white background of the image after convolution and thus those edges were lost. These edges could have been preserved if I adjusted my floating point to zero value threshold. In general edges near “T junctions” seemed to be more susceptible to noise and false positive zero crossings when compared to isolated edges, especially isolated rectangular edges which were found accurately in all 3 images.

Note: code for question 4 and 5 is in 'q2.m' at the bottom of the file

4.

Question 2 image crop (segment = [20:40, 40:60]) with σ = 1



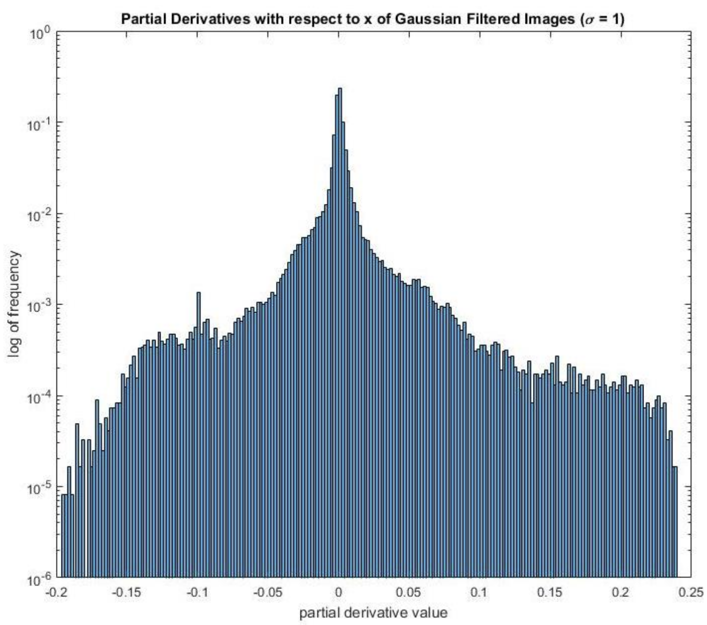
The relationship between the first two images is the white pixels in the right image depict what was interpreted as the edge(s) from the left image, similarly the black pixels translate to what was interpreted as non edges. We note that the edge does not appear to be 1 pixel wide, this is caused by the blurring/smoothing of pixel intensities during convolution.

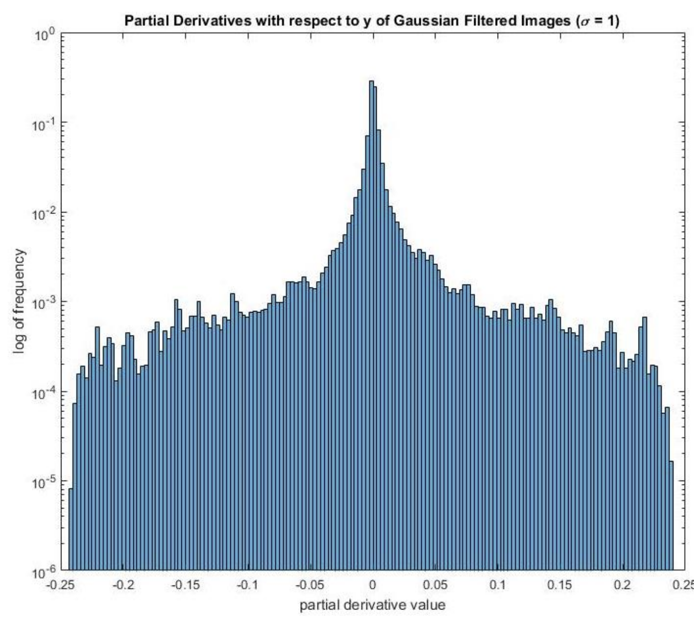
Looking at the third image (quiver overlay) we see that pixels with large arrows were converted into white pixels in the binary image and vice versa for small arrows and black pixels. This makes sense because the size of the arrow is directly related to the gradient magnitude which determined whether a pixel was interpreted as an edge or not when processing the binary edge position image. We see that image gradients are larger at intensity edges and much smaller in flat intensity regions, in fact some arrows are barely visible in our image because the neighbouring pixel intensities are virtually identical and thus there is no local difference.

Looking closer at the large gradients we see that the arrows start from the side of the intensity edge with lower intensity and point toward the side with greater intensity. This agrees with the definition of local differences we defined in class. Gradient arrows that are either vertical or horizontal have all of their intensity difference found in the y direction or x direction respectively, and angled gradient arrows have a local intensity difference in both directions.

5.

Histograms of partials derivative with σ = 1 (left: x, right: y)

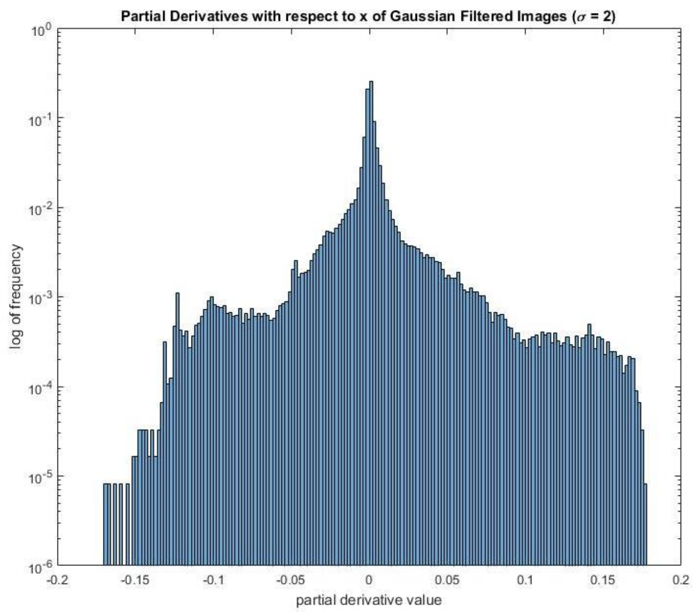


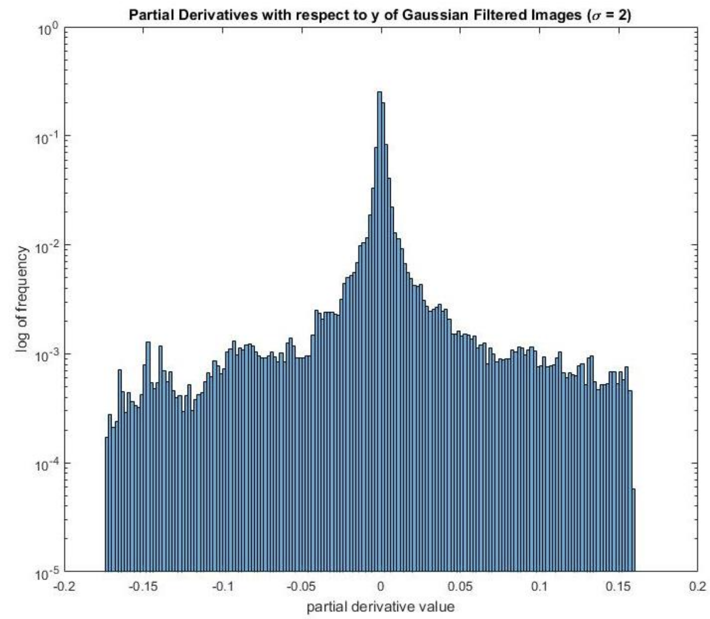


The center of the histogram is very Gaussian like, however as stated in the question the tails of the histogram are not. The main difference is that the histogram frequency values do not tend to zero as the partial derivative values grow (positively and negatively), after the initial drop in frequency from the center, the histogram remains somewhat constant, where as a Gaussian distribution continues to drop and the frequency approaches zero as you travel more and more standard deviations away from the mean. Further the histogram frequencies drop to zero very suddenly as opposed to the gradual decrease in frequency of a Gaussian.

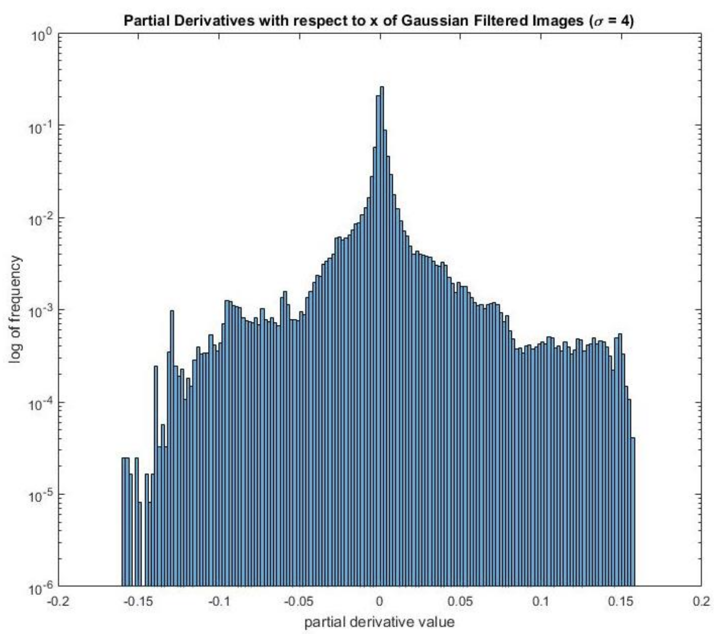
The reason behind this difference in the tails of a Gaussian and our histogram is created by the image smoothing performed during convolution. Our original image had a few examples of extreme local differences in pixel intensities, which would have translated into a histogram with tails more similar to a Gaussian. However, smoothing the image removed these cases of massive local differences and brought them to a more reasonable balance (i.e. smaller gradients). Smoothing our image effected small local differences very little, but had a great impact on the magnitude of large local differences, thus the gradient values in the tail got squeezed together and became less Gaussian like in shape.

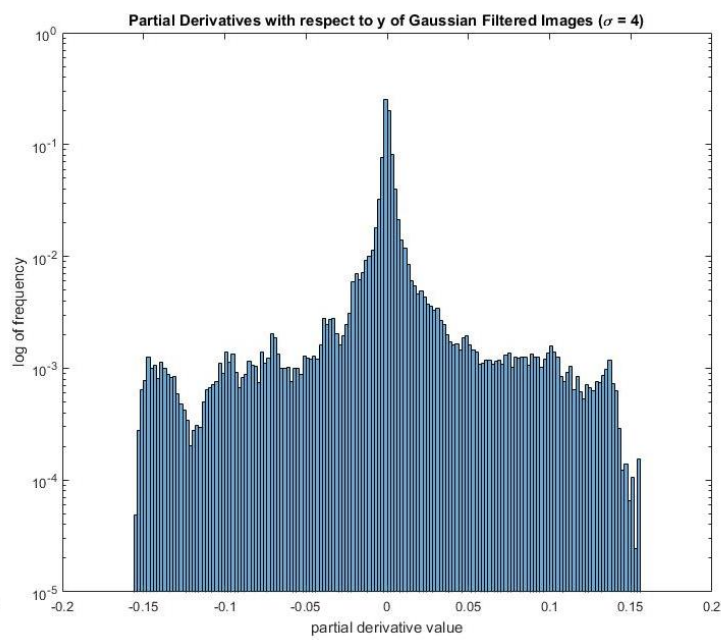
Histograms of partials derivative with σ = 2 (left: x, right: y)





Histograms of partials derivative with σ = 4 (left: x, right: y)





The other sets of histograms also do not follow an exact Gaussian distribution. Near the mean the histogram is Gaussian like however away from the mean the frequencies are still relatively constant and do not tend to 0 as you would expect if it followed the Gaussian. The drop off at the ends of the histograms becomes more sudden and severe as σ increases.

Looking at the difference between histograms we see that the 'end point' of the histograms (i.e. where the frequency becomes 0) is different in each histogram. We notice that as σ increases the range of gradient values with non-zero frequency decreases. The reason for this is that as σ increases the image becomes more blurred since the filter weights are more uniform and thus significant changes in intensities are smoothed out significantly more than in images with lower values of σ. This leads to a smaller range of partial derivative values as σ increases.