UNIVERSITY OF BIRMINGHAM

Project Report

Assessed Assignment

Computational Vision
School of Computer Science

Abid Abdul Gafoor - 1987680

13th December 2018

Declaration

I confirm that I have read and understood the University's definitions of plagiarism and collusion from the Code of Practice on Assessment. I confirm that I have neither committed plagiarism in the completion of this work nor have I colluded with any other party in the preparation and production of this work. The work presented here is my own and in my own words except where I have clearly indicated and acknowledged that I have quoted or used figures from published or unpublished sources (including the web). I understand the consequences of engaging in plagiarism and collusion as described in the Code of Practice on Academic Integrity (Appendix A).

Table of Contents

Table of Figures

1 Aim

The assignment is divided into 2 parts: The 'Laplacian of Gaussian' of the 'shakey.gif' image and the efficacy of different edge detectors on the three 'fluorescing cells' images. A mask to implement the Laplacian of Gaussian (second derivative of the Gaussian noise function) is created to understand the relationship between two 2-dimensional convolution and four 1 dimensional convolutions in sequence [1]. For the second part, edge detectors such as Roberts, Sobel, First-order Gaussian, Laplacian and LoG are tested on the given images where the edge points are measured against the labelled image [5].

2 Method

2.1 Laplacian of Gaussian (LoG)

The Laplacian is the second derivative of the inputs x and y added together i.e. the second spatial derivative of an image. It is used to highlight the regions of rapid intensity change such as edges. In this case, the Laplacian is applied to a binary or greyscale image that is initially smoothed using a Gaussian smoothing filter [4]. The filter reduces the sensitivity of the image to noise and the Laplacian finds its edges. The equati$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right]e^{-\frac{x^2+y^2}{2\sigma^2}}$ah blah blah which was the basis for the function shown below to determine LoG in an image. Three parameters were changed: standard deviation, size and the threshold to find the best LoG for the 'shakey' image [2]. The pseudocode below helped create a LoG mask that was then applied onto the shakey image. Before, the mask was applied, the image was converted from a 3D matrix to a 2D matrix i.e. from RGB to greyscale [1]. Using MATLAB, the standard deviation and size were changed to help determine, the best parameters for edge detection. Finally, the threshold of the image was altered to help visualise the edges a lot clearer. In terms of standard deviation, when all the edge values are tightly bunched together and the bell-shaped curve is steep in a normal distribution graph and the standard deviation is small but when they are spread apart, the bell curve is relatively flat, meaning that the data points have a relatively large standard deviation [2].

```
function m = create_log(size,sigma) %inputting the size and standard deviation
    w = (size+1)/2;
    for i=1:size
        for j=1:size
            x = i - w;
            y = j - w;
            m(i,j)= (x^2+y^2-2*sigma^2) * exp(-(x^2 + y^2)/(2*sigma^2)) /sigma ^
4;
        end
    end
end
```

Figure 1 - Pseudocode showing how to calculate the Laplacian of Gaussian

2.2 Cell Detection

The main idea behind cell detection is to compare the relationship between the manually altered cell edges with edge detecting algorithms. For this, three example images (9343 AM, 10905 JL, 43590 AM) were used with varying degrees of brightness. After converting each of the images from a 3D to a 2D convolution i.e. to greyscale, the edge detector masks were placed on the

images to find the edges of the cells [1]. A threshold was set on each of the edge detectors to find the best value for optimal edge detection.

First the images were converted to greyscale using rgb2gray. The edge detector function was then used to input the x and y values from the Roberts and Sobel matrices [1]. For the rest of the edge detectors, the mask was convoluted onto the image itself as shown below.

```
shakey1 = rgb2gray(imread('10905 JL.bmp'));
m1 = convn(shakey1 , A, 'same'); %for everything else
% m1 = get_edge(shakey1 , A , B); %for sobel and Robert where A&B represented
the x and y values
% show_image(m1>20) %greyscale image
% show_image(m1<20) %inverse image
```

Figure 2 - Pseudocode showing how to display images after filtering with an edge detector

2.2.1 Edge Detectors

The three different images were first masked with the Roberts filter as seen in figures 16, 17 and 18. A threshold was set on all of the images: 10, 15 and 20. An inverse of the masked image was plotted in addition to the masked image as seen in those figures. This process was repeated for all the other edge detector but with different thresholds. For Sobel, the experimental thresholds were 30, 40, 60 (figures 19, 20, 21); for first-order Gaussian, 2, 3 and 5 (figures 22, 23, 24); for Laplacian, 1, 5, 8 and 10 (figures 25, 26, 27 and 28); and for Laplacian of Gaussian, 1, 3, 4 and 5 (figures 28, 29, 30 and 31) [5]. For Laplacian and Laplacian of Gaussian, one extra threshold was experimented upon as the edges could not be as easily detected compared to the other edge detectors [5].

2.2.2 Receiver Operator Characteristic (ROC)

ROC is the plot between the true positive rate and the false positive rate. In this case, the plot is between the true edges of the image and the acquired edges from the edge detectors. To plot the ROC, every single point was iterated from 1 to 100 in 1 step intervals [3]. The true positive values were from the manually edited images and the false positive values from the edge detector. They were then plotted against each other where the true positive values remained constant and the false positive values depended on the edge detector used: Roberts, Sobel, First-order Gaussian, Laplacian and Laplacian of Gaussian [3][6].

```
for i = 1:1:100
   celltemp = cell1_roberts>i;
   [tpr,fpr] = roc(cell9343_model_threshold,celltemp);
   tprlist(i) = tpr;
   fprlist(i) = fpr;
end
```

Figure 3 - Pseudocode showing how to calculate the ROC for each edge detector

3 Results

3.1 Laplacian of Gaussian (LoG)

The values for the size of the filter was changed and experimented between 7 and 9 (as seen from figures 4 and 12). For both, changing the standard deviation size affected the overall image quality [2]. For 7 and 9, the standard deviation was tested between 0.5 and 2 and the best one for 7 was determined to be 1.45 and for 9 was 1.9. In addition, a threshold was set on the

'Laplacian of Gaussian' mask on the image. This caused the image to only accept values above the threshold, reducing noise. However, when the threshold was increased, the image started losing detail, so a trade-off was determined between high and low thresholding. After vigorous testing, two instances were perfect for the LoG problem: a filter size and standard deviation of '7 and 1.45' and '9 and 1.9' [2]. As they were very similar, it was concluded that 7 and 1.45 would be used for the cell detection part of the experiment. In addition, to further improve the edge detection a threshold value of 30 was set.

3.2 Cell Detection

3.2.1 Edge Detectors

While experimenting, an interesting observation was seen. For every single filter, the higher the threshold was set, the noise level decreased and for each of the images, the optimal threshold value was different for each of the filters. For example, as seen from the figures, when applying the Roberts mask, the best threshold for the '9343 AM' image was 15 and for both '10905 JL' and '43590 AM' image, it was 10. The brightest image '10905 JL' had the best edge detection compared to the other two images. The best parameter was threshold 30 for the darkest image '43590 AM' when the Sobel filter was applied. The 'Laplacian of Gaussian' at 4 was the best threshold for the image '9343 AM'. For each individual mask, there were optimal thresholds. In Sobel, the best threshold for both '9343 AM' and '43590 AM' was 40 but for '10905 JL', it was 60. In First-order Gaussian, the best threshold for both '9343 AM' and '10905 JL' was 3 and for '43590 AM', it was 2. In Laplacian, the best threshold for all three '9343 AM', '10905 JL' and '43590 AM', the optimal was 8. Unfortunately, the Laplacian filter didn't detect edges at all, deeming it completely useless. In Laplacian of Gaussian, the best threshold for both '9343 AM' was 4 and for both '10905 JL' and '43590 AM', it was 3. From just visual observation without any comparison, the worst two edge detectors are the 'Laplacian' and the 'Laplacian of Gaussian'.

3.2.2 Receiver Operator Characteristic (ROC)

According to Zweig and Cambell, the closer the ROC curve is to the upper left corner, the higher the overall accuracy of the test i.e. the closer, the graph (in figure 33) to (0,1), the more accurate the data [3][7]. Here, the best edge detector that has the highest correspondence to the manually edited cells is Roberts, quickly followed by Sobel. This is further justified by the detection of edges in the darkest image by the Sobel filter. The 'Laplacian' filter is the worst for edge detection as stated above but with the addition of Gaussian smoothing, the 'Laplacian of Gaussian' model performed insignificantly better [3][4]. The reason that 'Laplacian' filters were not able to detect edges accurately is due to the fact that it approximates the second derivative, which is a lot more prone to noise. Therefore, the filter would be able to detect noise more than the edges themselves [6].

4 Conclusion

The experiment was successful in determining the best edge detector for detecting edges in images (Roberts) by conducting various experiments on the threshold values of the detectors. A ROC graph was created to prove this visualisation [3]. Finally, an algorithm was created to work out the Laplacian of Gaussian for each image given [5].

5 References

[1] Machinelearninguru.com. (2017). *Image Convolution*. [online] Available at: http://machinelearninguru.com/computer_vision/basics/convolution/image_convolution_1.html [Accessed 10 Dec. 2018].

[2] Niles, R. (2018). *Statistics: Definition of Standard Deviation*. [online] Robert Niles. Available at: https://www.robertniles.com/stats/stdev.shtml [Accessed 8 Dec. 2018].

[3] Schoonjans, F. (2018). *ROC curve analysis with MedCalc*. [online] MedCalc. Available at: https://www.medcalc.org/manual/roc-curves.php [Accessed 9 Dec. 2018].

[4] Homepages.inf.ed.ac.uk. (2003). *Spatial Filters - Gaussian Smoothing*. [online] Available at: https://homepages.inf.ed.ac.uk/rbf/HIPR2/gsmooth.htm [Accessed 14 Dec. 2018].

[5] Signal Processing Stack Exchange. (2017). *What is the difference between Difference of Gaussian, Laplace of Gaussian, and Mexican Hat wavelet?*. [online] Available at: https://dsp.stackexchange.com/questions/37673/what-is-the-difference-between-difference-of-gaussian-laplace-of-gaussian-and [Accessed 10 Dec. 2018].

[6] Metz CE (1978) Basic principles of ROC analysis. Seminars in Nuclear Medicine 8:283-298.

[7] Zweig MH, Campbell G (1993) Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine. Clinical Chemistry 39:561-577.

6 Appendix

A Images

A.1 Laplacian of Gaussian of the 'shakey' image

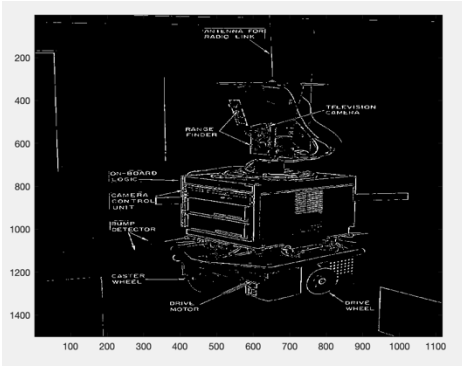

Figure 4 - Threshold>30

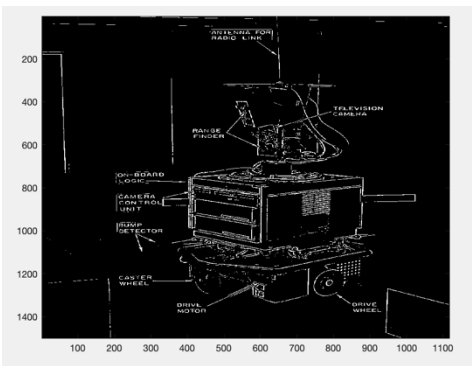

Figure 5 - Threshold>20

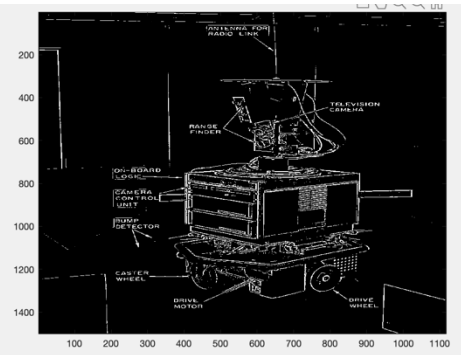

Figure 6 - Size: 7, Sigma: 1.45

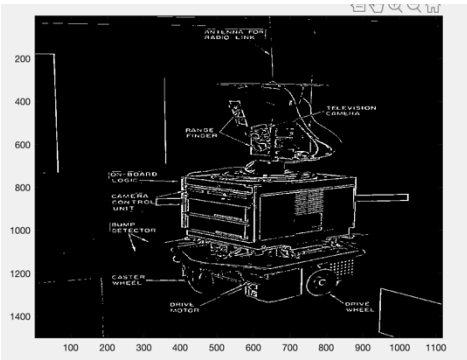

Figure 7 - Size: 9, Sigma: 1.95

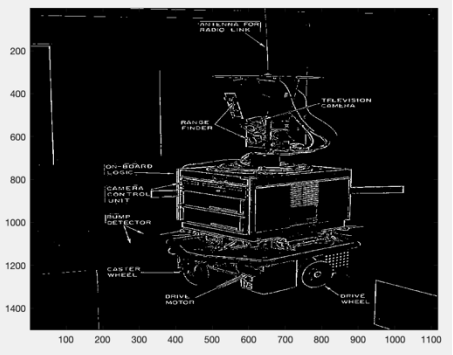

Figure 8 - Size: 7, Sigma: 1.5

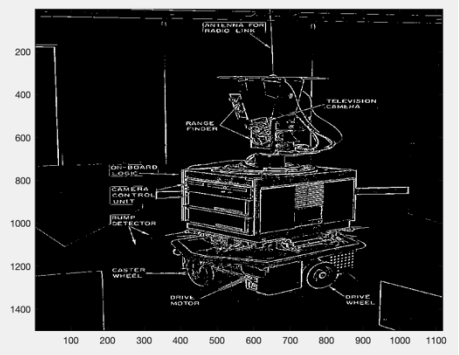

Figure 9 - Size: 7, Sigma: 1.4

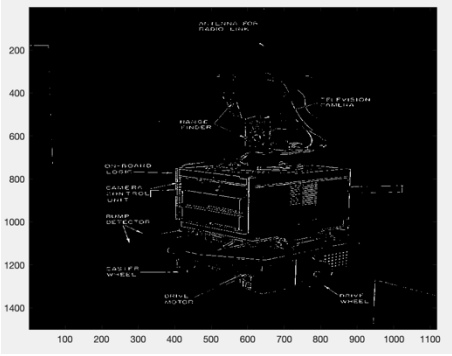

Figure 10 - Size: 7, Sigma: 1.7

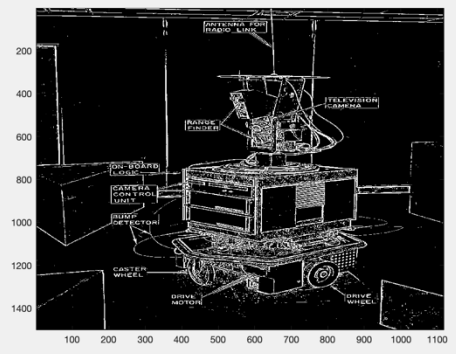

Figure 11 - Size: 7, Sigma: 1.2

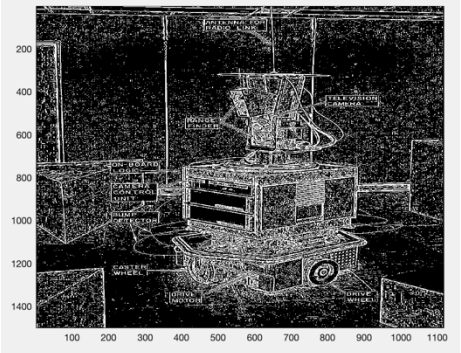

Figure 12 - Size: 7, Sigma: 1

A.2 Fluorescing cells with their manually detected edges

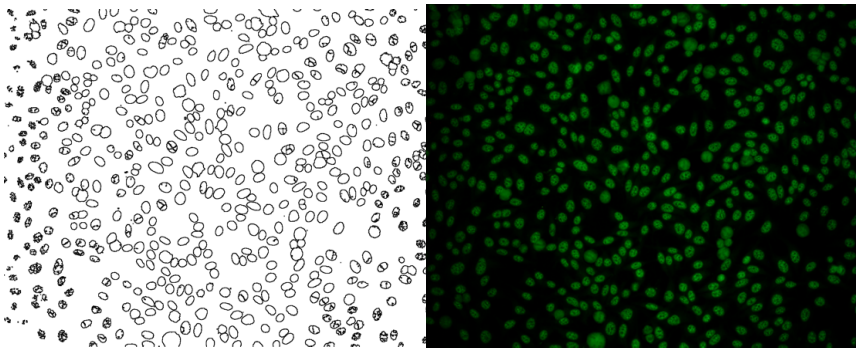

Figure 13 - Image (9343 AM) with manually detected edges.

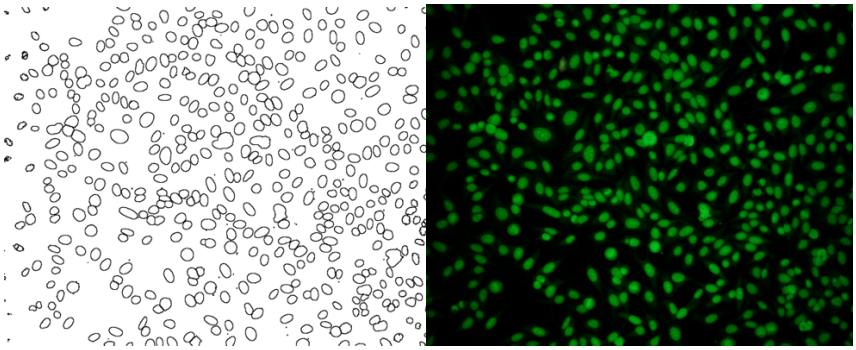

Figure 14 - Image (10905 JL) with manually detected edges.

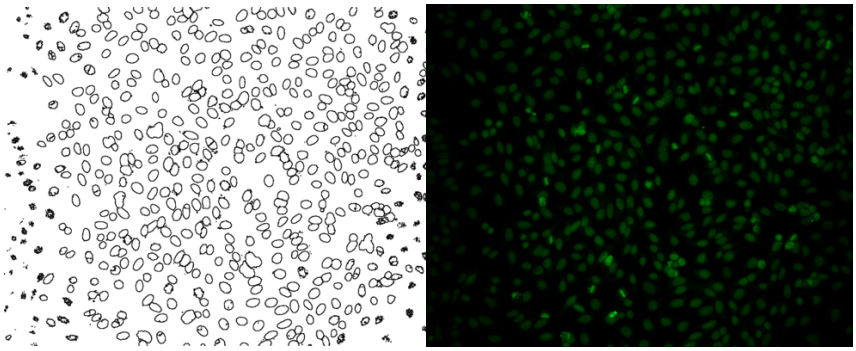

Figure 15 - Image (43590 AM) with manually detected edges.

A.3 Fluorescing cells with their manually detected edges

A.3.1 Roberts

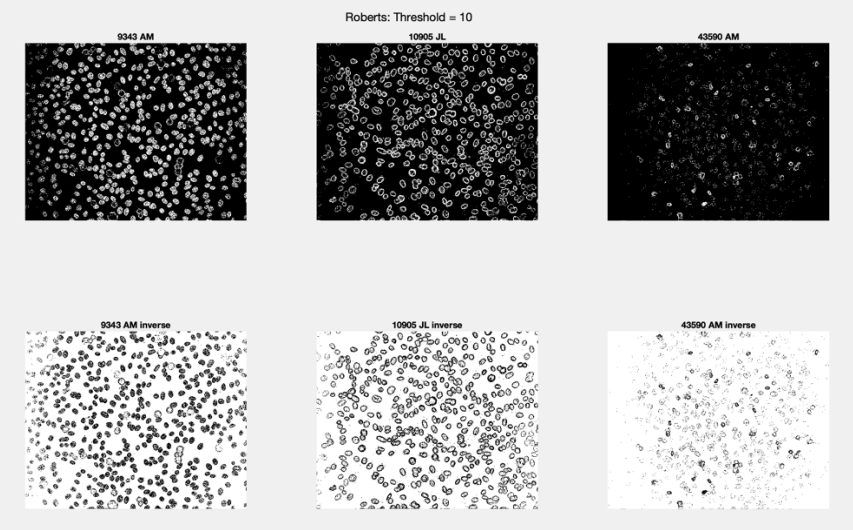

Figure 16 - Roberts: Threshold = 10 for each type of image

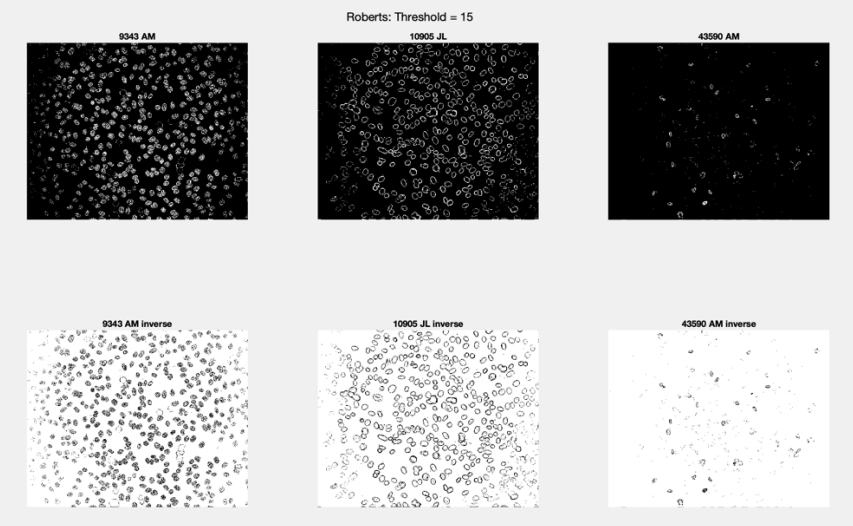

Figure 17 - Roberts: Threshold = 15 for each type of image

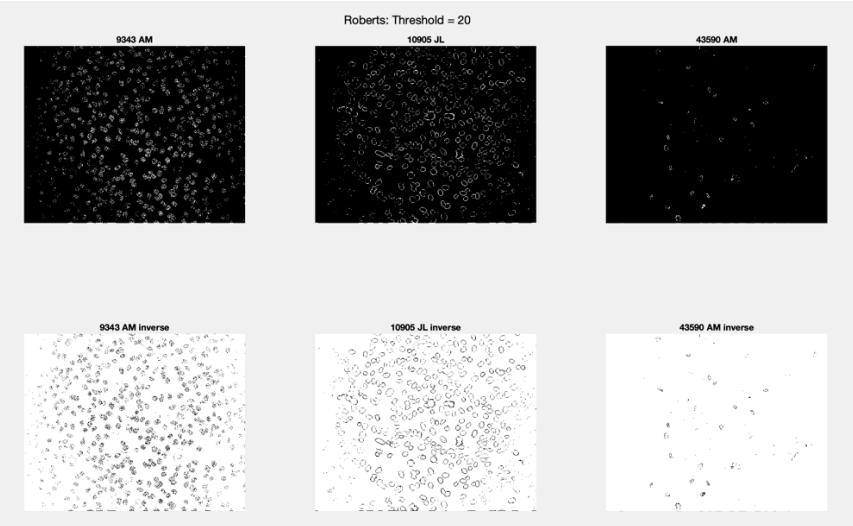

Figure 18 - Roberts: Threshold = 20 for each type of image

A.3.2 Sobel

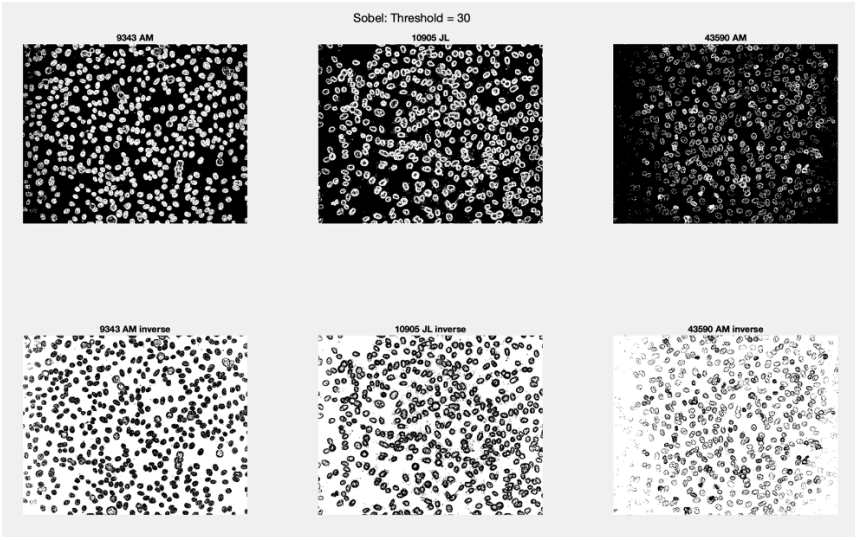

Figure 19 - Sobel: Threshold = 30 for each type of image

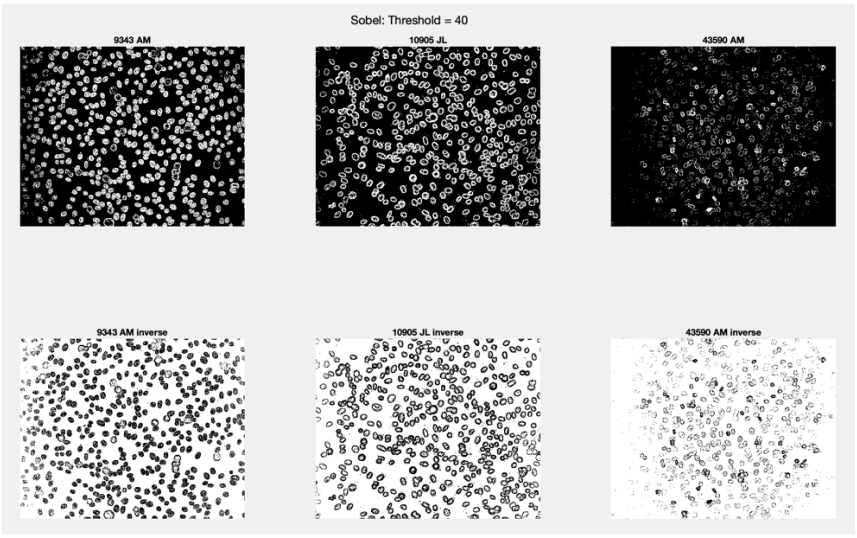

Figure 20 - Sobel: Threshold = 40 for each type of image

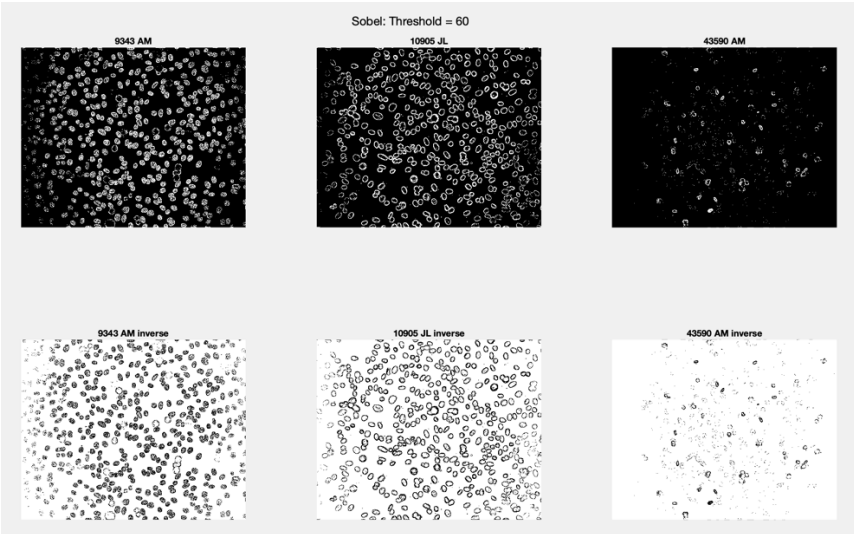

Figure 21 - Sobel: Threshold = 60 for each type of image

A.3.3 First Order Gaussian

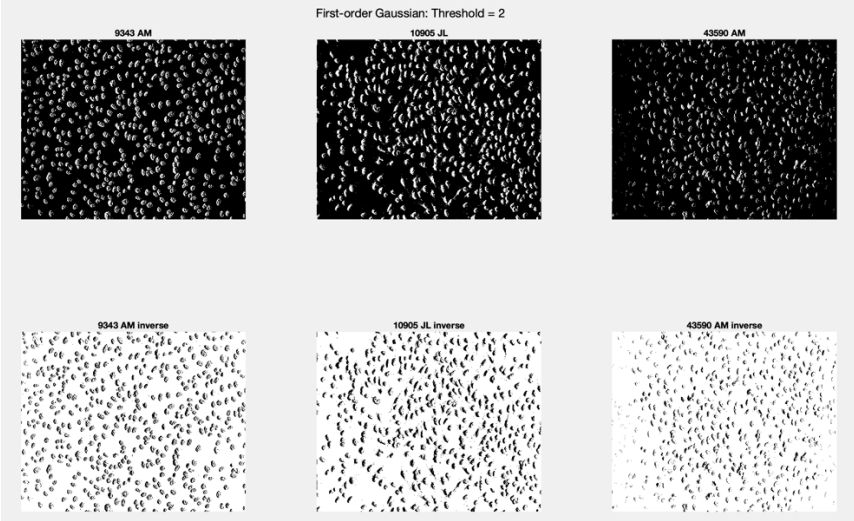

Figure 22 – First-order Gaussian: Threshold = 2 for each type of image

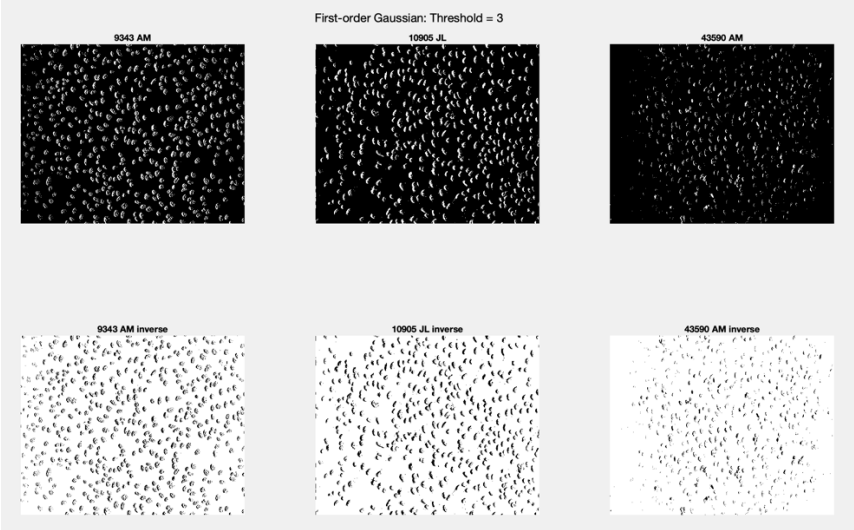

Figure 23 – First-order Gaussian: Threshold = 3 for each type of image

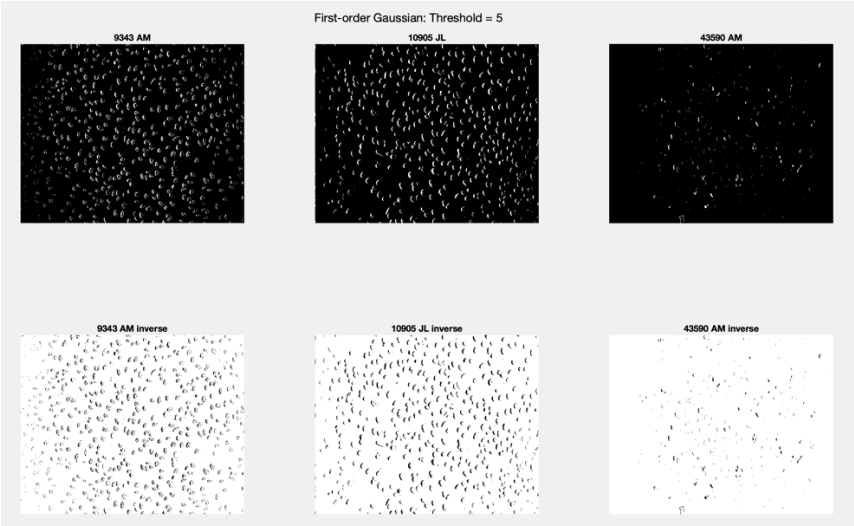

Figure 24 – First-order Gaussian: Threshold = 5 for each type of image

A.3.4 Laplacian

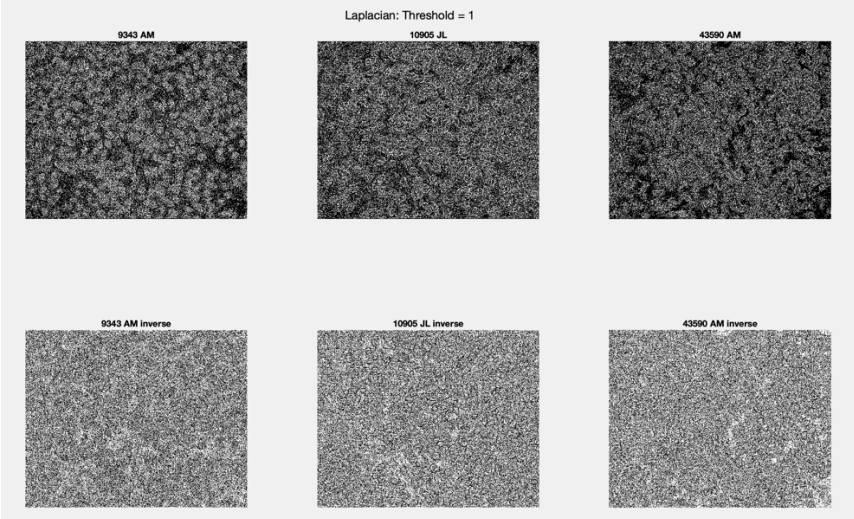

Figure 25 – Laplacian: Threshold = 1 for each type of image

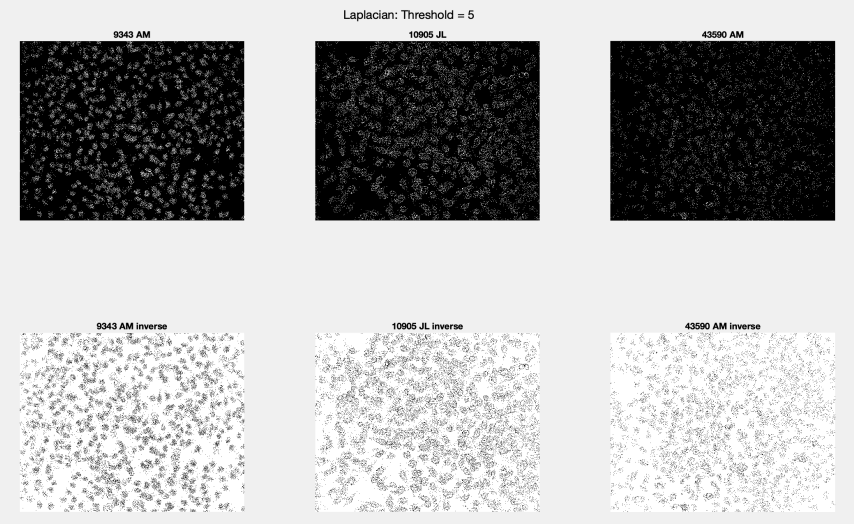

Figure 26 – Laplacian: Threshold =5 for each type of image

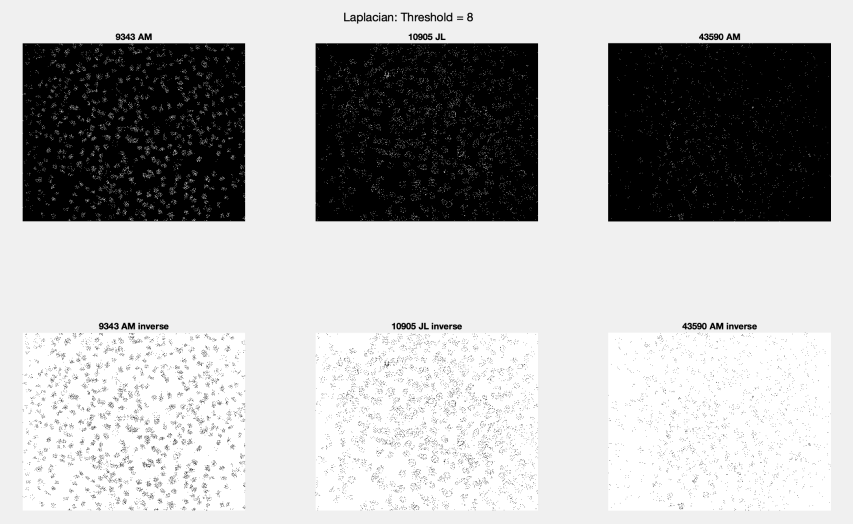

Figure 27 – Laplacian: Threshold = 8 for each type of image

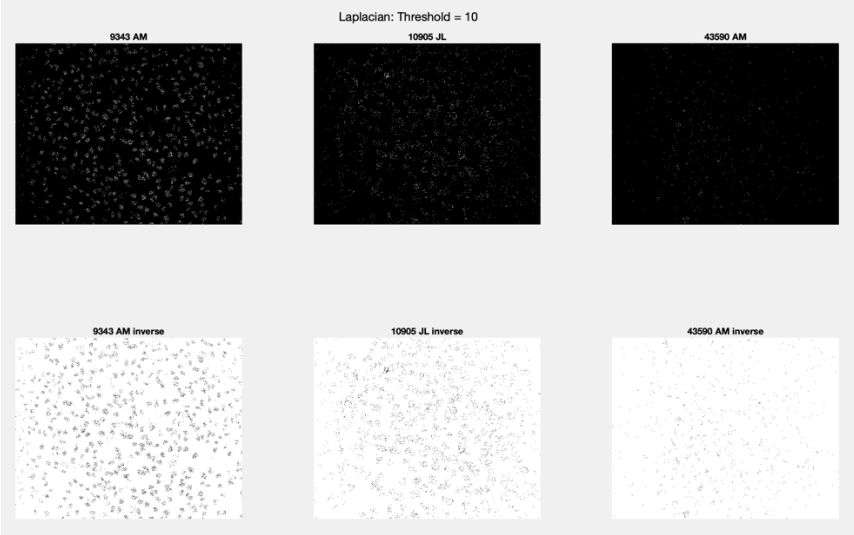

Figure 28 – Laplacian: Threshold = 10 for each type of image

A.3.5 Laplacian of Gaussian

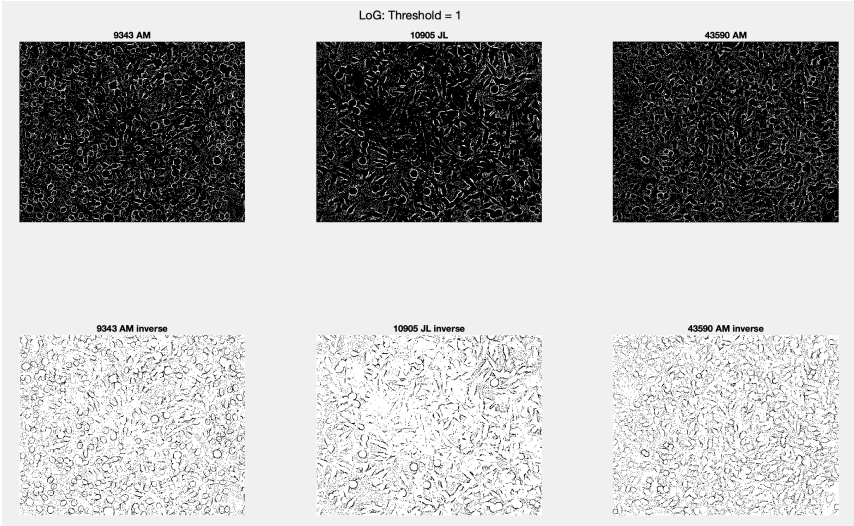

Figure 29 – Laplacian of Gaussian: Threshold = 1 for each type of image

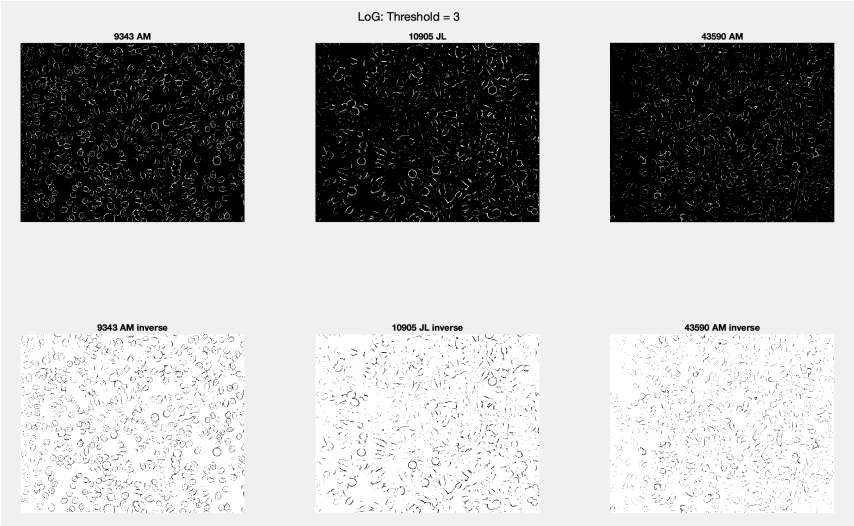

Figure 30 – Laplacian of Gaussian: Threshold = 3 for each type of image

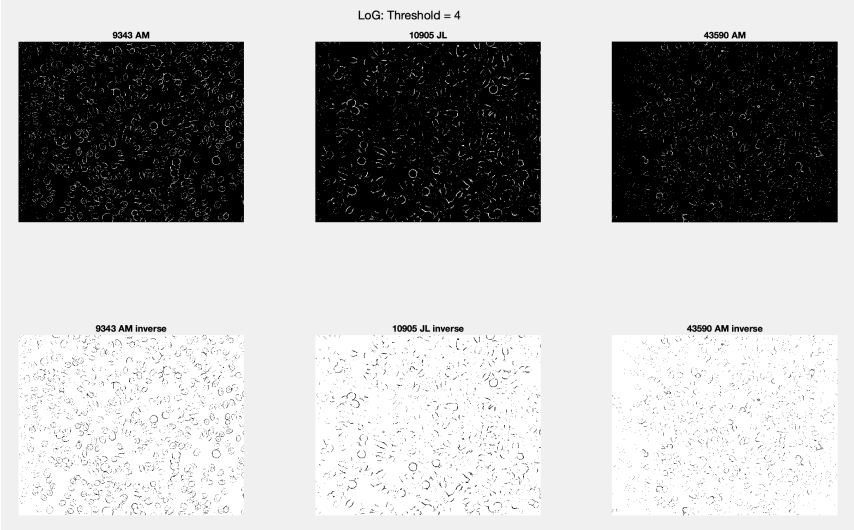

Figure 31 – Laplacian of Gaussian: Threshold = 4 for each type of image

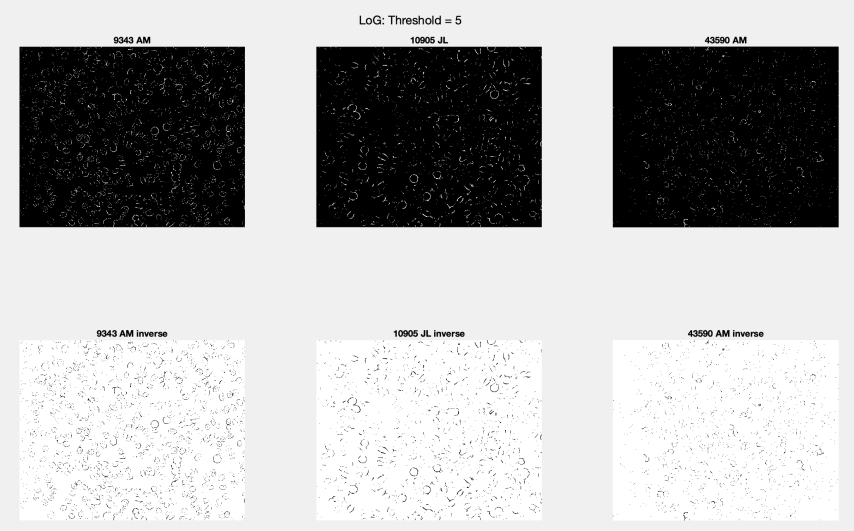

Figure 32 – Laplacian of Gaussian: Threshold = 5 for each type of image

A.4 ROC graph showing the different filters

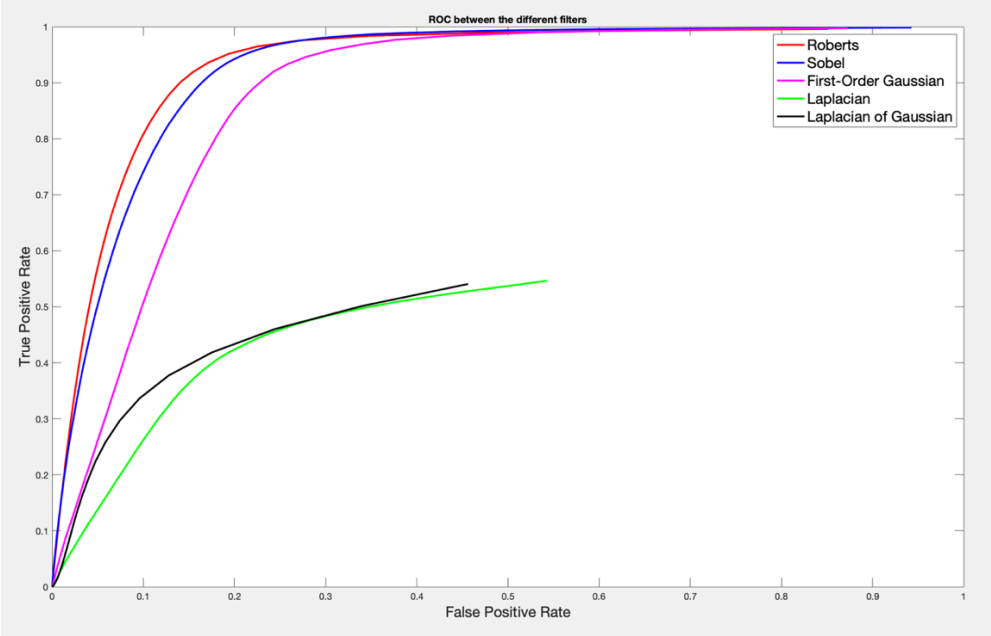

Figure 33 – ROC of the different filters