CALIFORNIA STATE UNIVERSITY, LONG BEACH

College of Engineering
Department of Computer Engineering and Computer Science
Dr. Thinh V. Nguyen
Spring 2007
CECS-553/653: Machine Vision

PROJECT 2

Name: _____Foss, Shannon_____
Last, First

- **Dates:** Date assigned: Tuesday March 13, 2007. Date due: Tuesday April 10, 2007. Late submissions will receive penalty at 10% per day. This project is worth 35% of the project grade.

- **Objectives**: The objectives of this project includes: (1) to study connected components labeling algorithms, (2) to perform edge detection and study the effects of various parameters, and (3) to perform morphological operations.

- **Project Description:**

A. Programming Project:

Write a computer program to read in an image named "image.bmp" where *image* is the input to the program and is selected from the image database. Save the image in grey level as "image_grey.bmp" Note that the word "image" should be replaced with the appropriate image file name. Cut out a sub-image having size of NxN with center at coordinates (x,y) of the "image_grey.bmp" image. Save this cut-out image as "image_spatialN.bmp". For images with no cutout, use the original grey level image. Perform these operations for the following images with the corresponding NxN and (x,y):

Image	NxN	(x,y)
Ziyi Zhang	no cutout	
circles and lines	no cutout	
arteries	256x256	(140, 140)
pattern1	512x512	(310, 460)
pattern3	512x512	(460, 310)

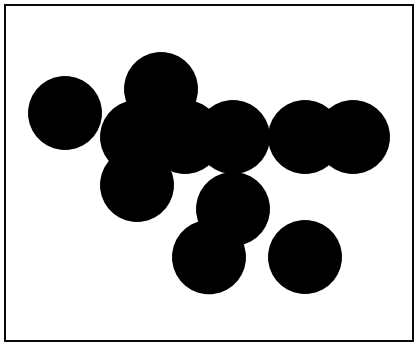
Perform the following operations: **(for individual projects, do only questions 1 through 3, question 4 is for extra credit. For group projects, do questions 1 through 4.. There is no extra credit question for group projects.)**

1) Edge detection using Roberts and Sobel operators: (20 points) Apply the Roberts and Sobel operators on the above images. Show the resulting output. Then, threshold the output images to obtain the edge images. The edge pixels are black and the non-edge pixels are white. Discuss how you select the threshold to obtain the edge images.
2) Edge detection using Laplacian of Gaussian: (30 points) Apply the Laplacian of Gaussian with mask sizes of 7x7 and 21x 21 on the above images. Show the values of the masks. For each mask size, select various values of σ. Use zero crossings to detect the edges. Discuss the effects of mask size and σ on the resulting edge images.
3) Edge detection using morphological operators: (20 points) Apply morphological operators with appropriate structuring elements for edge detection on "circles and lines".
4) Line counting: (30 points) Use morphological operators with appropriate structuring elements on the image "circles and lines" to produce an image containing only the lines. Then, apply a connected components labeling algorithm to label the lines and determine the number of lines.

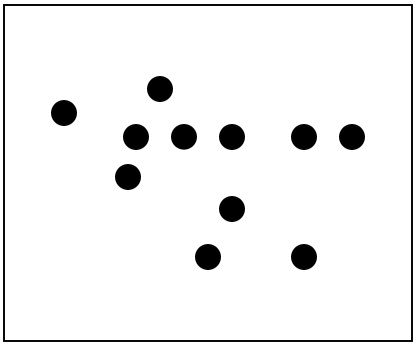
B. *Non-programming project:*

This project is to develop algorithms to apply mathematical morphology in handling occlusion problems. Provide pseudocode for the algorithms. An example of the problem is given below.

Suppose an image of a number of disks is given. The disks may be occluded or partially overlapped. The objective is to count the number of disks (e.g., coins). Suppose the size of the disks is known. A possible result is to produce an output image B. Discuss how to handle the severe occlusions or when there are disks of different but known sizes.



Original image A



Output image B

☐ **Project Report:** Follow the required format. Attach this sheet as the cover sheet for the report. Attach printouts of the results for the above images. Discuss the results and specific implementations

- *Project Description:* A brief description of the problem statement. Discuss any relevant assumptions if necessary.
- *Project Background:* Provide a brief summary of background or theory of the techniques used to implement the project.
- *Algorithm Description*: Discuss the specific algorithms or techniques used to implement the project. Pseudo code should be used where relevant.
- *Results and Analysis*: Provide the results of the work. Analyze the results. Determine if the results are as expected. Discuss. The results should include outputs of the program (e.g., processed images, tables, useful data).
- *Conclusion:* Provide a brief conclusion of the project.

Project #2 – Edge Detection
Shannon Foss
April 10, 2007

Introduction –

In this project we used several different methods for finding the edges of an object within an image. One method was to use the Roberts and Sobel's operators, which are kernels that are convoluted with an image to find its edges. Laplacian is a calculated mask that is then applied to the image to detect edges. Other methods also include different types of morphological operators which may be used to manipulate the image to get the information that is needed. This report will cover all these operators and their results.

Description –

Roberts Cross and Sobel Operators -

Roberts and Sobel operators are both easy and fast edge detectors to create and to run. These kernels are convolved with the image to calculate a gradient between the grey levels in the image, and that gives a value to the pixel that is being calculated. The more the gradient becomes steeper, the more the value of the pixel will go up.

The Roberts Cross operator consists of two 2 by 2 kernels, one for the X direction and the other for the Y direction.

+1	0
0	-1

Gx

0	+1
-1	0

Gy

Figure 1 – The Roberts Cross operator in the X and Y directions.

The operator is placed on the image with the top left corner corresponding to the image pixel being evaluated. Convolution is then performed (convolution consists of multiplying each pixel of the image with the matching kernel element and then summing those together, that sum is the new value of the pixel in the image. Then the next pixel is evaluated the same way) on each pixel in turn until all of them have been evaluated. Then the operator for the opposite direction is applied in the exact same way. These two values are then added together. The resulting data is the new image that highlights the edges of the object in the picture.

(a) (b)

Figure 2 – (a) original image - Flowers_grey.bmp, (b) Image that has had the Roberts Cross operator applied to it – Flowers_Roberts.bmp

The Sobel operator is applied in exactly the same way, the only difference is that it is two 3 by 3 kernels rather than 2 by 2s, and this time, the center element of the kernel is placed over the image pixel being evaluated rather than the top left corner.

-1	0	+1
-2	0	+2
-1	0	+1

Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

Figure 3 – The Sobel operators for the X and Y directions

As you can see here, the Y kernel is just a 90° counterclockwise rotation of the X kernel. The same is true for the Roberts Cross operator as well. You may also threshold the output so it shows up better and possibly better represent what you would like to view.

(a) (b)

Figure 4 – (a) original image – Flowers_grey.bmp, (b) Image that has had the Sobel operator applied to it – Flowers_Sobel.bmp

Laplacian of Gaussian –

Gaussian Blur is a convolution operator where the mask is determined by the equation:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Equation 1 – the Gaussian Equation – an equation used to create blur in images to reduce noise.

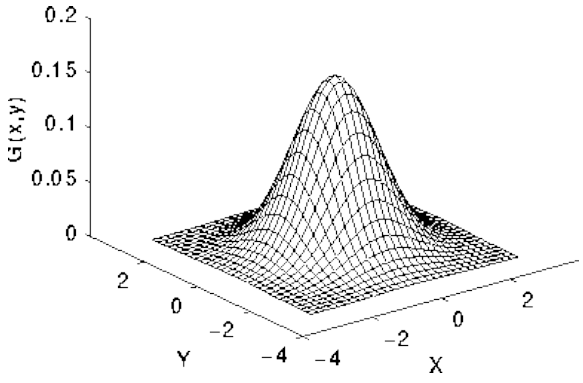
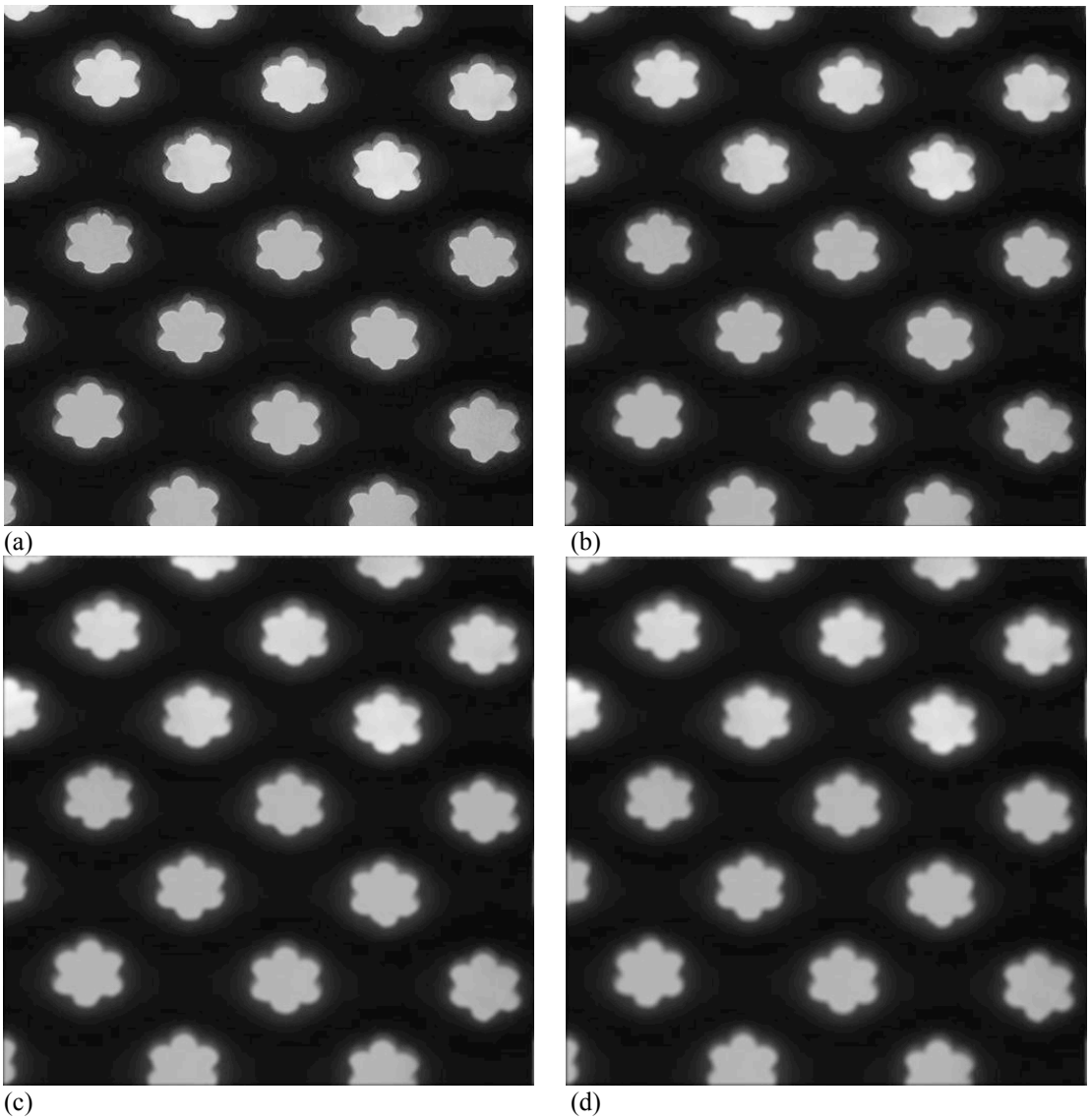


Figure 5 – (right) The representation of the Gaussian Equation – mean at (0,0) and σ = 1.0.

where σ is the number of standard deviations included under the curve, and x and y are the number of pixels away from the element being evaluated in the X and Y directions. There is more than one way to create the Gaussian kernel though, you may also use the one dimensional equation to create a 1-D array of Gaussian values and then matrix multiply it by itself to create the 2-D kernel. This is possible because the Gaussian surface is circularly symmetrical with itself.

Gaussian Blur is used for removing excess noise and blur out random stray pixels from a picture. Many second derivative filters, such as Laplacian, are sensitive to such noise, and so it needs to be minimalized as much as possible.

When implementing Gaussian blur, you must first create the convolution kernel that will be applied to the image. This is accomplished by deciding on a σ value and a kernel size, then applying it to the equation. The σ represents the number of standard deviations from the mean will be under the curve, the higher the value of σ, the blurrier the image will turn out to be, also, the lower the value of σ the sharper the image will be. Interestingly enough, if σ is less than 1, the new picture will be sharper than the original. In the figures you will notice that once σ gets to a certain value, it doesn't produce a great deal more blur than the previous σ (Figure 6 - (c) and (d)). This is because as σ extends out, the values at those points become very small, and so they do not effect as much of the image.

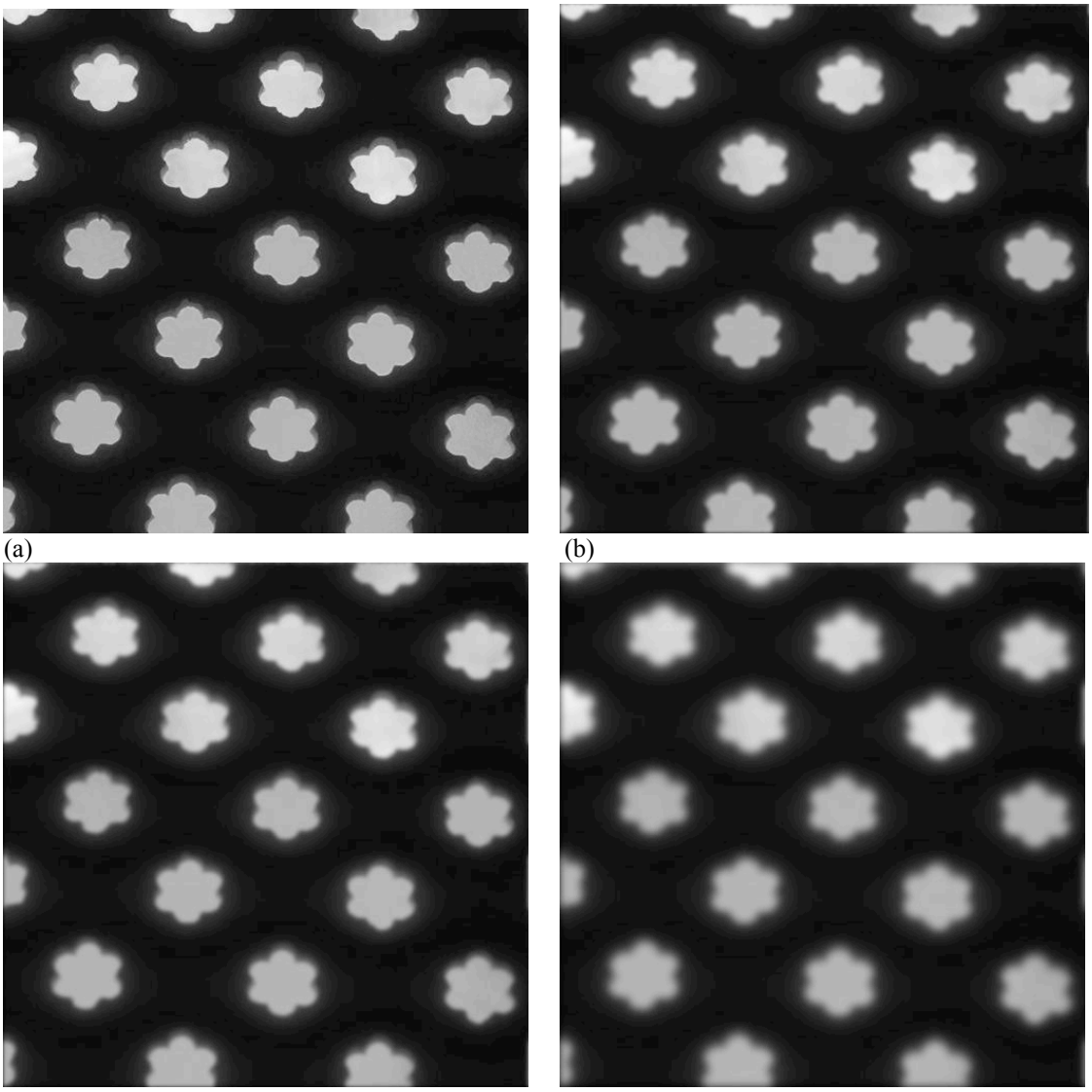


(a) (b)

(c) (d)

Figure 6 – Image with varying degrees of blur but all with the same kernel size, (a) original image – pattern1_spacial512.bmp, (b) Blur with σ = 1.0 – pattern1_Gauss7-1.bmp, (c) Blur with σ = 2.0 – pattern1_Gauss7-2.bmp, (d) Blur with σ = 4.0 – pattern1_Gauss7-4.bmp.

The size of the kernel being applied also has an effect on how much blur is created. For instance, a larger kernel will create much more blur than a smaller sized kernel. This is because the kernel is applied over a larger area and so there is more influence from neighboring pixels on determining its value. As you can see below in Figure 7, the difference between image (b), which uses a 7x7 size kernel, and image (c), which uses a 21x21 size kernel, is somewhat substantial.



(a) (b)

(c) (d)

Figure 7 - Image with varying σ and kernel size values, (a) original image – pattern1_spacial512.bmp, (b) Blur with σ = 2.0 and kernel size = 7x7 – pattern1_Gauss7-1.bmp, (c) Blur with σ = 2.0 and kernel size = 21x21 – pattern1_Gauss7-2.bmp, (d) Blur with σ = 4.0 and kernel size = 21x21 – pattern1_Gauss7-4.bmp.

Laplacian is the 2nd spatial derivative of an image. It is used for edge detection because it is easily able to pick up on sharp color level changes, and so, distinguishing between an object and the background becomes trivial and in the resulting image, it stands out very well with bright clean edges. However, it does not do well with gradual color level changes, they tend to blend together and so it can't always pick up on them and the resulting image usually has very broad edges.

There are three kernels that are principally used that approximate the Laplacian filter:

0	1	0
1	−4	1
0	1	0

(a)

1	1	1
1	−8	1
1	1	1

(b)

−1	2	−1
2	−4	2
−1	2	−1

(c)

Figure 8 – Three common Laplacian kernels

these are convolved with the image using the center point of the kernel for the placement on the pixel being calculated.

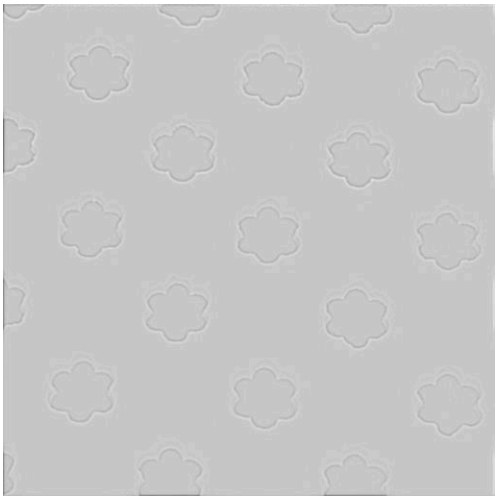


Figure 9 – Laplacian 3x3 approximation kernel (b), applied to a previously Gaussian blurred image - pattern1_GL21-1.bmp

Laplacian Of Gaussian –

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right]e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Equation 2 – Laplacian of Gaussian Equation

This is the equation to use if you wish to produce a kernel for both Gaussian and Laplacian all at once.

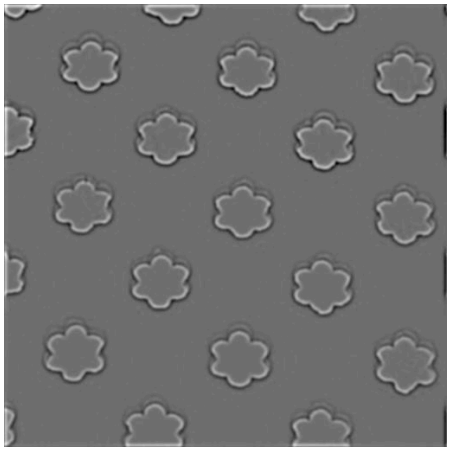
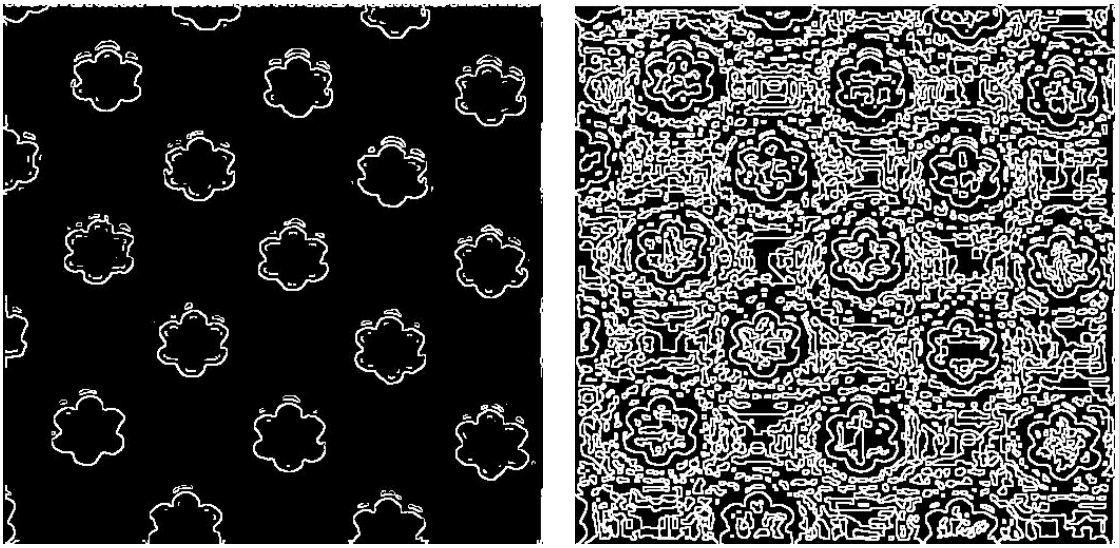


Figure 10 – Laplacian of Gaussian applied on an image, - pattern1_LoG21-2.bmp

Zero Crossings –
The place where Laplacian crosses from negative to positive is called a zero crossing, these are easy to detect, and as such, they can sometimes give great edges.



(a) (b)

Figure 11 – (a) pattern1_Log7-2.bmp, (b) pattern1_Log21-2.bmp

Morpholigical Edge Detector –

By using morphological kernels you can manipulate the image, ANDing or ORing the image with a morphological kernel you can dilate or erode the object in the image. In the Figure below, you can see the effect of dilation followed by subtraction of the original image.

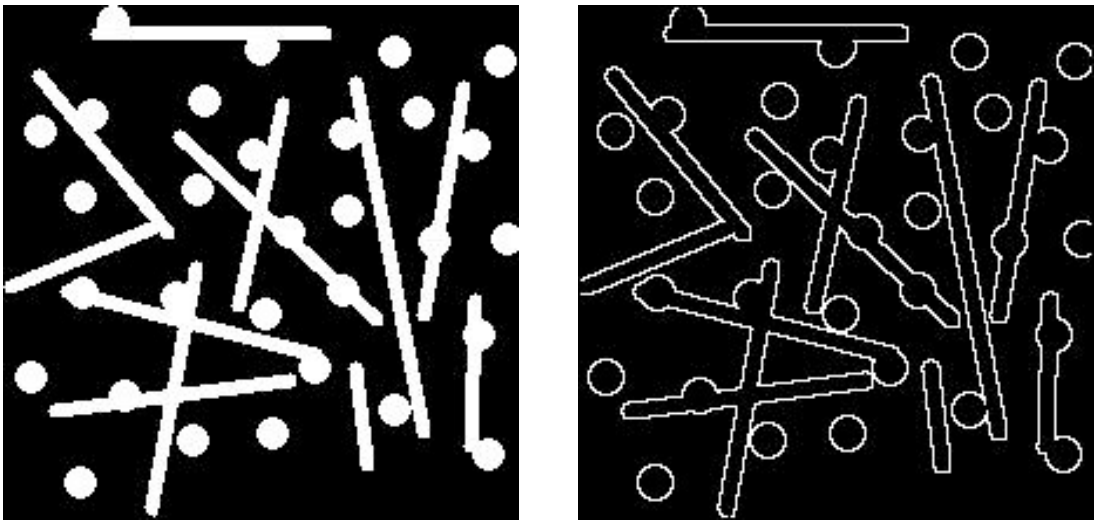
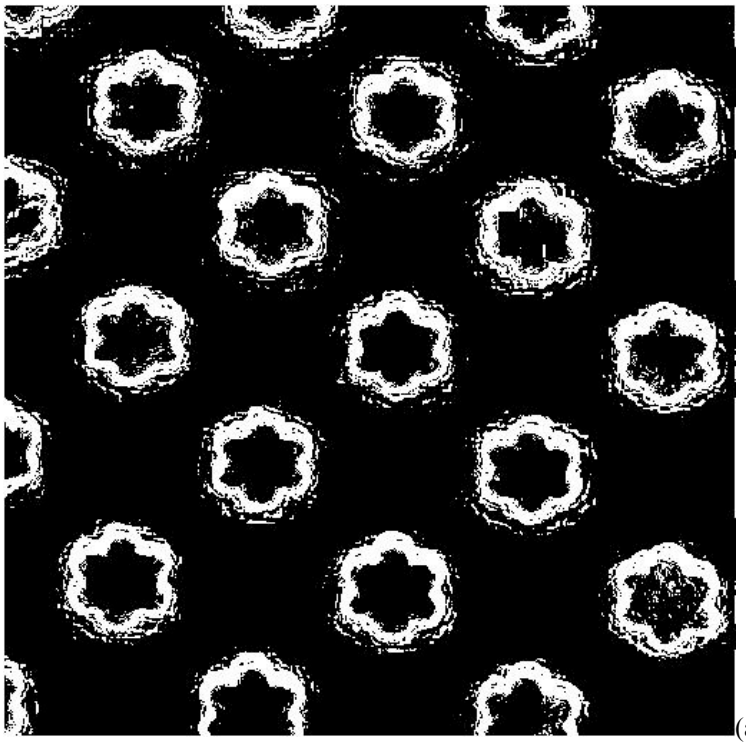


Figure 12 – before and after morphological operations on circles_and_lines.bmp when creating an edges only picture

Thinning –

Thinning is a series of morphological operators that when applied have the effect of minimizing a previously applied edge detectors output down to single pixel thin lines.


(a)

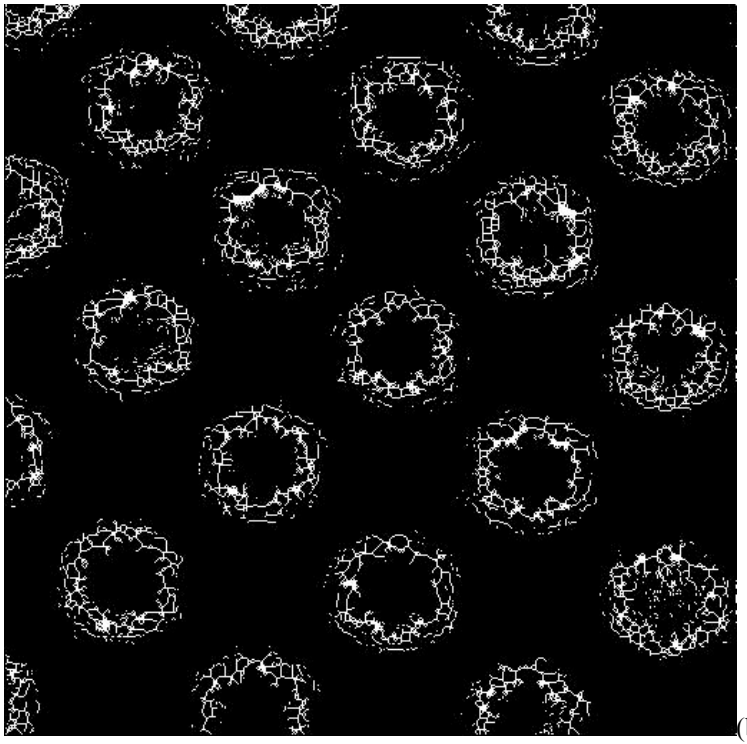

(b)

Figure 13 - Here you can see the effect of thinning of the Sobels (a) edges on pattern1_Sobel.bmp (b)

Line Counting –

This is the effect of a disk shaping morphological operator of size 14 being used to get rid of the circles. Unfortunately, I was not able to dilate the lines enough so that they would stay together when erosion was used on them.

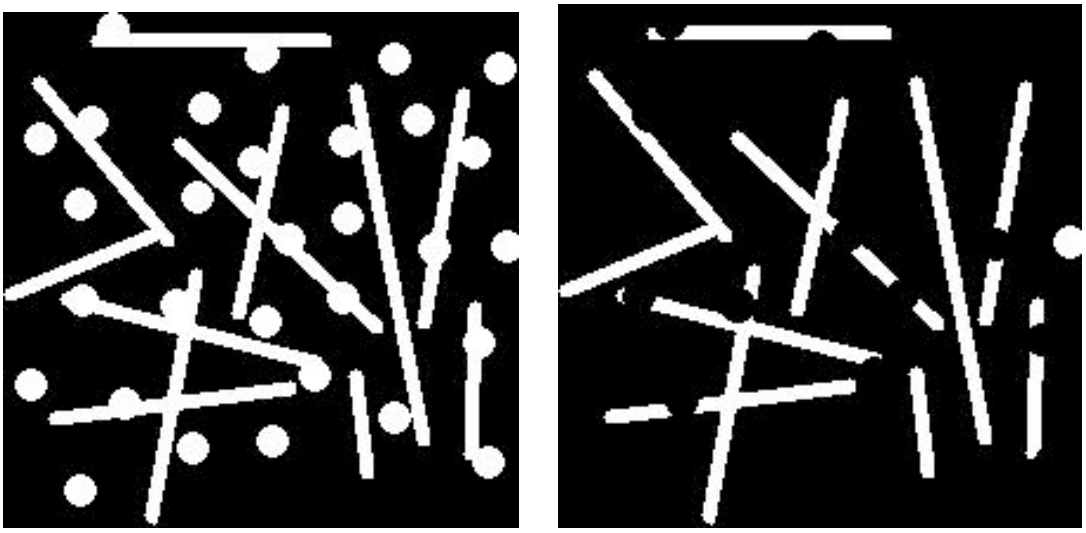
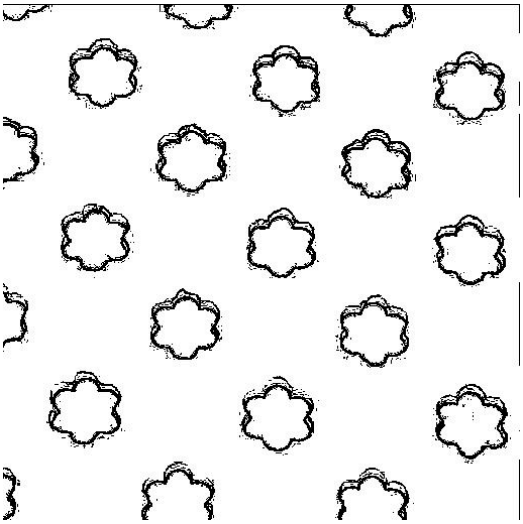


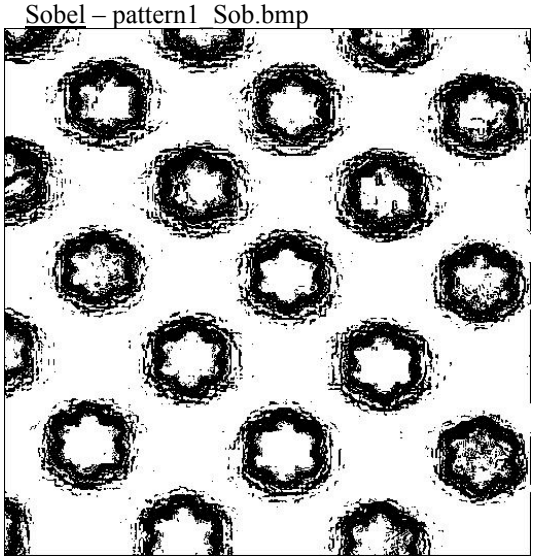
Figure 14 – before and after images of circles_and_lines.bmp when attempting to get an image of only the lines.

Extra Images –

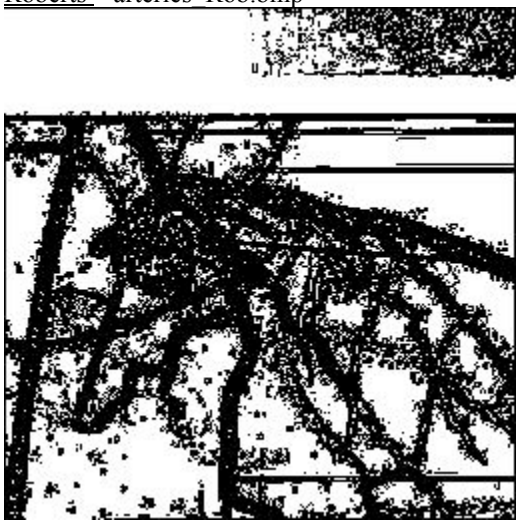
Roberts – pattern1_Rob.bmp



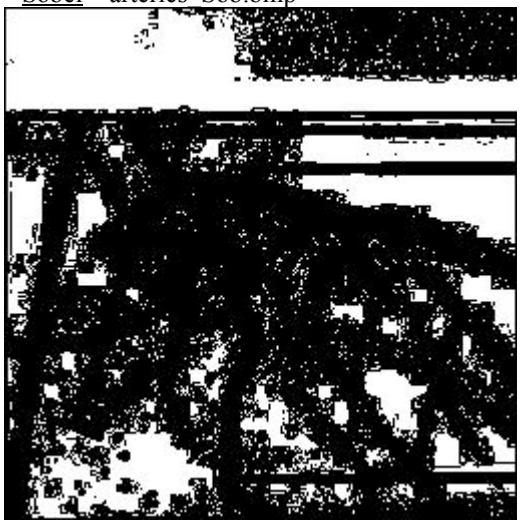
Sobel – pattern1_Sob.bmp



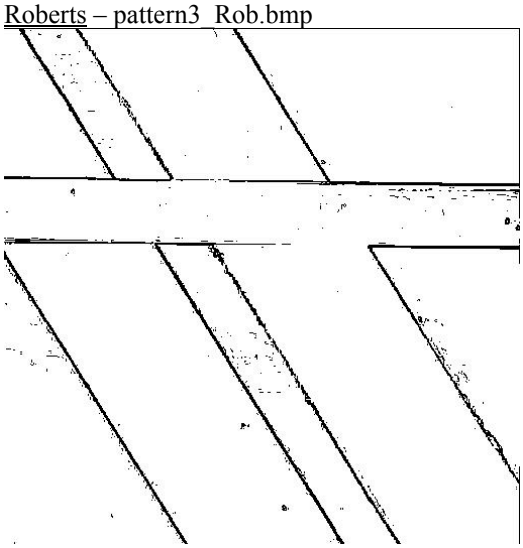
Roberts – arteries_Rob.bmp



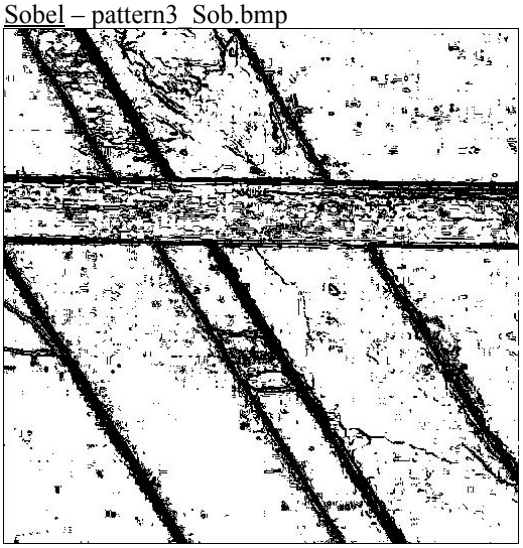
Sobel – arteries_Sob.bmp



Roberts – pattern3_Rob.bmp



Sobel – pattern3_Sob.bmp



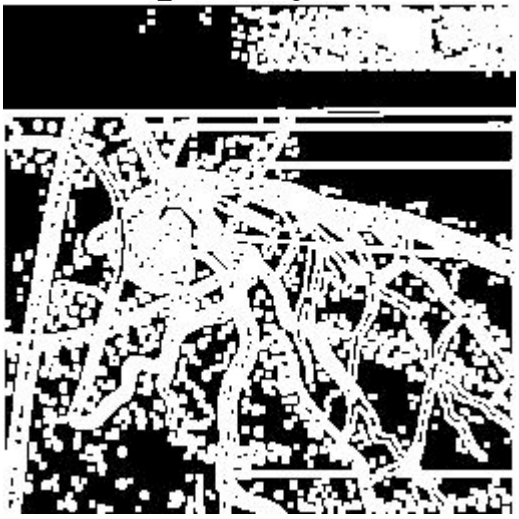
Roberts – ZiyiZhang_Rob.bmp

Sobel – ZiyiZhang_Sob.bmp

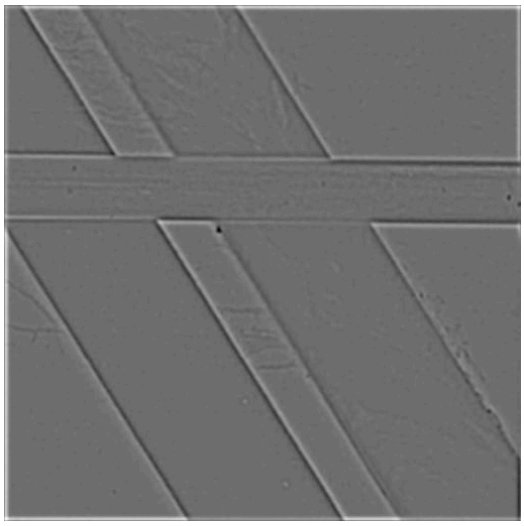
With the Roberts and Sobel operators, I chose the threshold so that all the pictures would be subjected to the same standard. I chose a threshold of values that were less than 25 would be white and any other value would be black.

Laplacian of Gaussian –
LoG - arteries_LoG7-1.bmp

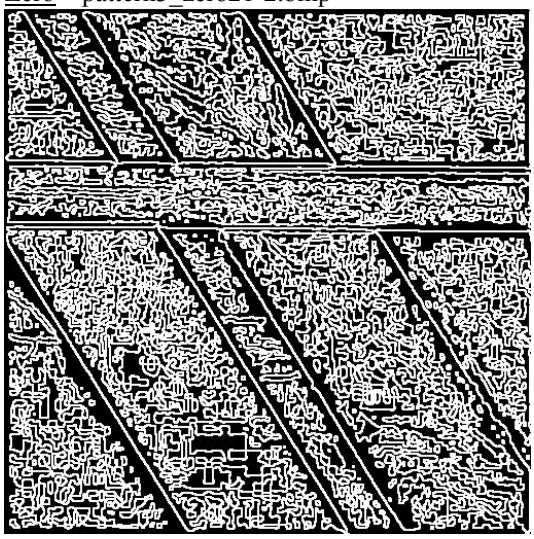
Zero - arteries_zero7-1.bmp



LoG – pattern3_LoG21-2.bmp



Zero – pattern3_zero21-2.bmp



LoG – ZiyiZhang_LoG7-4

Zero – ZiyiZhang_zero7-4