EECS: 332 Intro to Computer Vision, MP #5

Canny Edge Detection, Due on 11/08/2016

Pradyoth Hegde, Student Id: 2997017

Introduction

Canny Edge Detection is an Edge detection technique used to detect edges in an image while suppressing noise. It was developed by John F Canny in 1986. It can be used to extract structural information from different vision objects and dramatically reduce the amount of data to be processed. The main advantages about Canny edge detection are as follows:

- Low error rate
- Good localization
- Minimal Response

Algorithm for Canny Edge Detection:

Initially, we convert the RGB image into a Grayscale Image and then begin the analysis

Canny Edge Detection can be implemented using a set of simple steps

1. **Gaussian Smoothing: S = GaussSmoothing(I, N, Sigma)**

We need to filter out any noise in the grayscale image before trying to locate any edge. In order to perform this operation, we need to create a Gaussian Mask and then use it to filter the input image. We can use different masks and different "sigma" values

2. **Calculating Image Gradient: [Mag, Theta] = ImageGradient(S)**

After we get the smoothed image, we need to calculate the image gradient of it. We can use Roberts Cross or Sobel types to obtain the Gradient, however for this example, we have used Sobel as it is the default type. We have calculated the magnitude in both the X and Y directions, along with the Magnitude and Theta of the entire image

3. **Selecting High and Low Thresholds: [T_low, T_high] = FindThreshold(Mag)**

We need to calculate the Lower and Higher threshold values from the Magnitude matrix which can be used later. To do this, we first scale the "Mag" matrix to (0-255) and then create a PDF and CDF matrix for this. Using the CDF vector, we can assume that T_high is at 0.8 and T_low is at 0.4. These values are then returned

4. **Suppressing Nonmaxima: Mag = NonmaximaSuprress(Mag, Theta)**

Finding Local Maxima of the Image gradient is very essential. We can use either Interpolation or LUT. For this example, Interpolation has been used as it is more accurate than LUT. Non maximum suppression works by finding the pixel with the maximum value in an edge. This occurs when pixel q has an intensity that is larger than both p and r where pixels p and r are the pixels in the gradient direction of q. If this condition is true, then we keep the pixel, otherwise set the pixel to zero (make it a black pixel).

5. **Thresholding and Edge Linking: E = EdgeLinking(Mag_low, Mag_high)**

Once we have suppressed the Nonmaxima, thin edges are obtained in the image. Now, we have to link the edges based on the thresholds we had obtained in the previous steps. The high threshold produces strong edges and the low threshold produces weak edges. The entire purpose of canny edge detector is to recursively link the weak and strong edges. When a strong edge ends, keep growing the weak edge to fill gaps between strong edges.

Comparison of Different Parameters: The Gaussian Smoothing parameters(N, Sigma) and T_high and T_low values can be changed to obtain different accuracies. This will be discussed in result analysis

Comparison with Different Edge Detectors: We can check if Canny Edge Detector works better/worse than other edge detectors. This is also discussed in result analysis

Result Analysis

Our Canny Edge Detector was used on “lena.bmp” and the following results were obtained

Fig 1

Fig 2

Fig 3

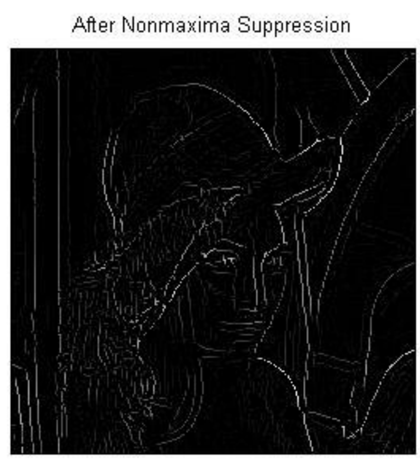


Fig 4

Fig 5

Fig 6

The results can be analyzed as follows:

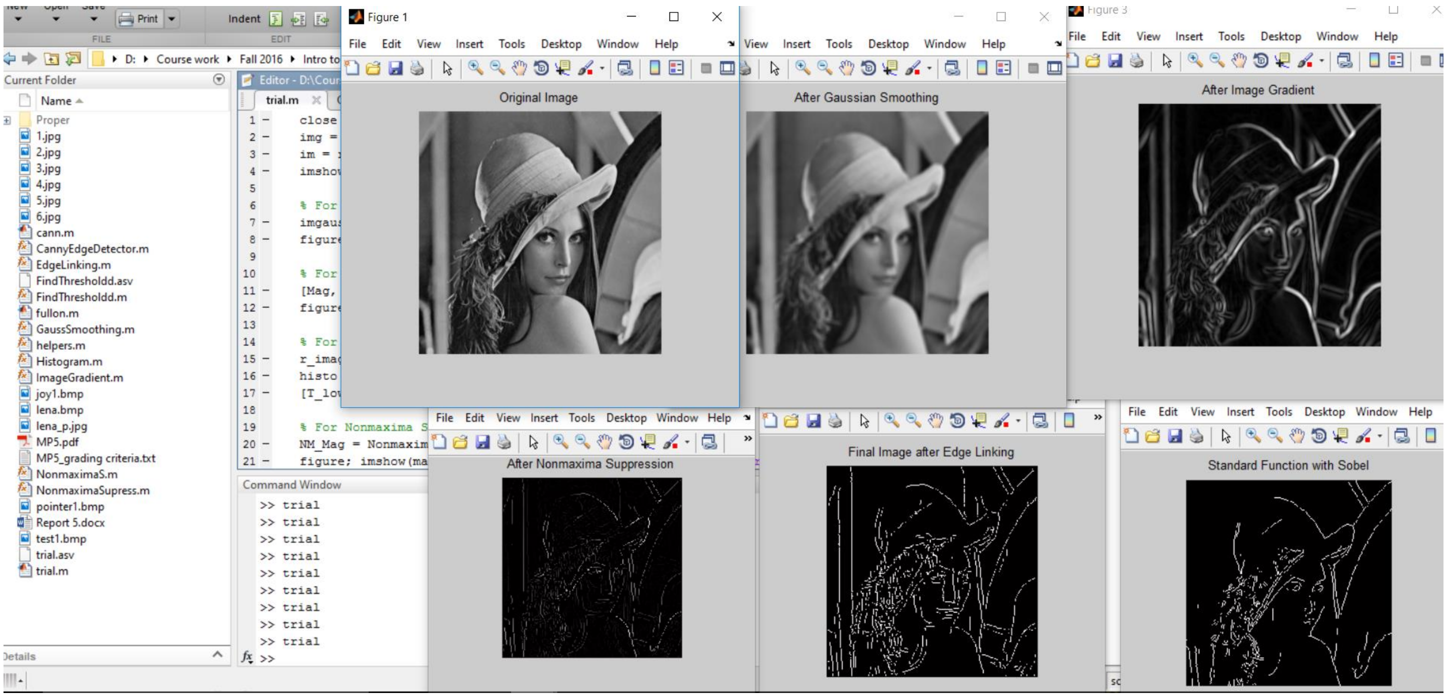
- Fig 1 is the Input Image ("lena.bmp") It looks like a grayscale image, but it is not
- Fig 2 is the Input Image after being converted to grayscale and Gaussian Smoothed. As you can see in the image, the image is a little blurred and the sharp edges have been removed using the Gaussian mask in this step
- Fig 3 is the output of the Image Gradient function. The output is the square root of the sum of squares in both the X and Y directions. This function produces both the Magnitude matrix and the Angle matrix, however, we are only interested in the Magnitude matrix
- Fig 4 is the output of Non Maxima Suppression using Interpolation. When compared to Fig 3, you can see that Fig 4 has much thinner edges. This has happened due to the process of Non Maxima Suppression. This can be used in order to perform edge linking
- Fig 5 is the final output after Edge Linking. The strong edges are first shown, then the weak edges connected to the strong edges are also shown to get the Final Image with only edges. This is the final out of the Canny Edge Detector
- Fig 6 is the output for the same image but using a standard edge detector available in MATLAB using a "Sobel" model

Results for Test Images

This is a screenshot of the results obtained when our canny edge detector was used to detect edges in the given image. There are 6 images, namely:

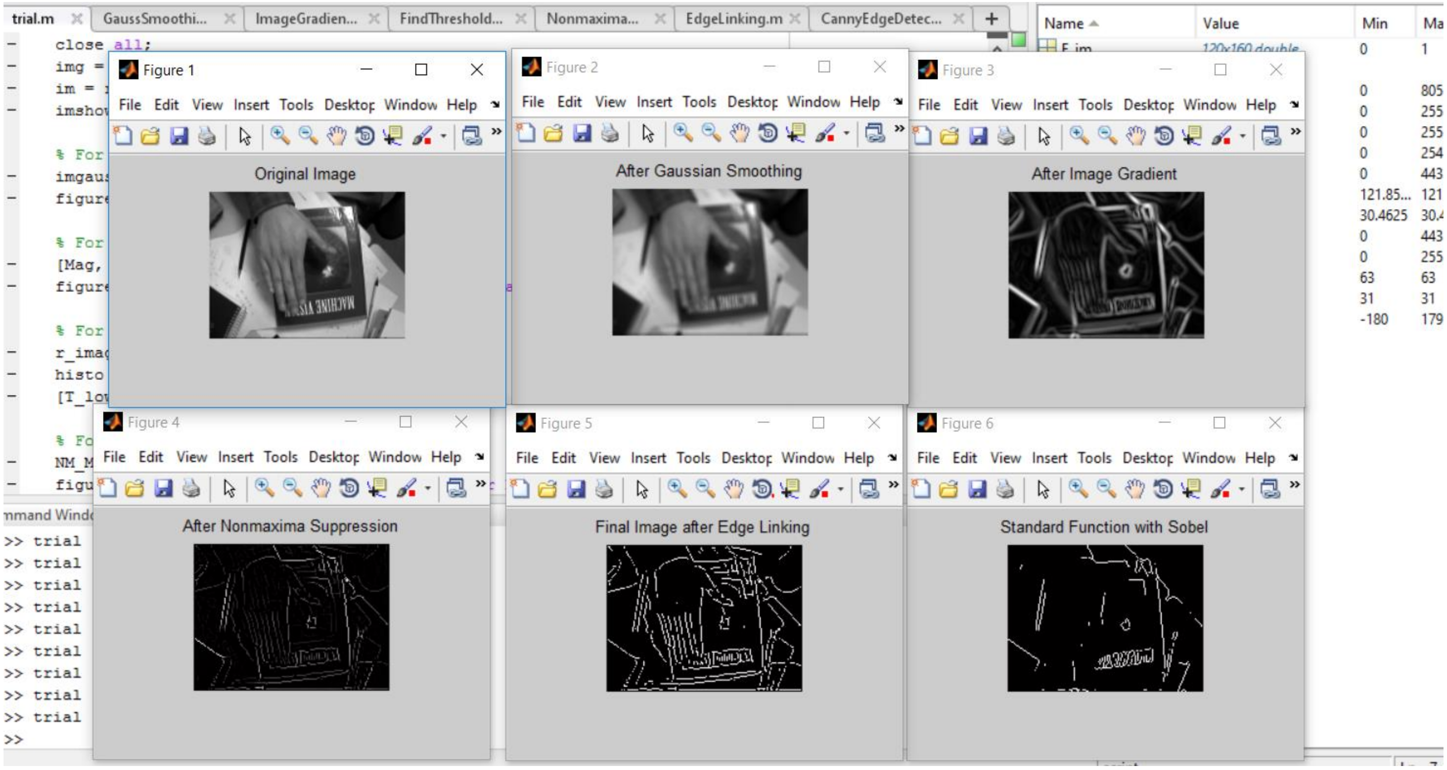
- Original Image
- Image after Gaussian Smoothing
- Image after Image Gradient
- Image after Non Maxima Suppression
- Final Image after Edge Linking
- Comparison with standard Sobel Function

1. **Lena.bmp**



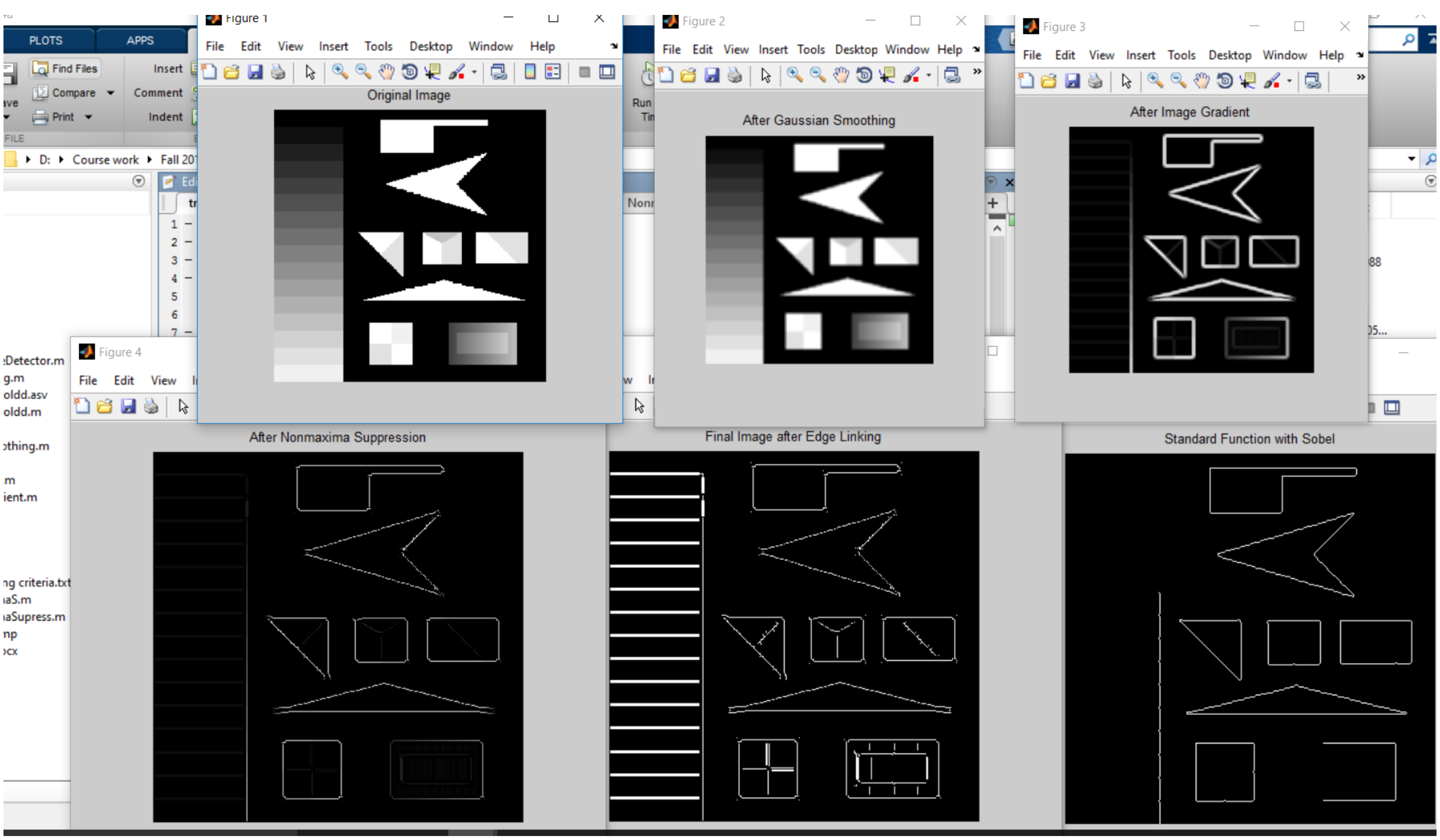
For this image, A Gaussian mask of N = 5 and Sigma = 3

2. **Joy1.bmp**



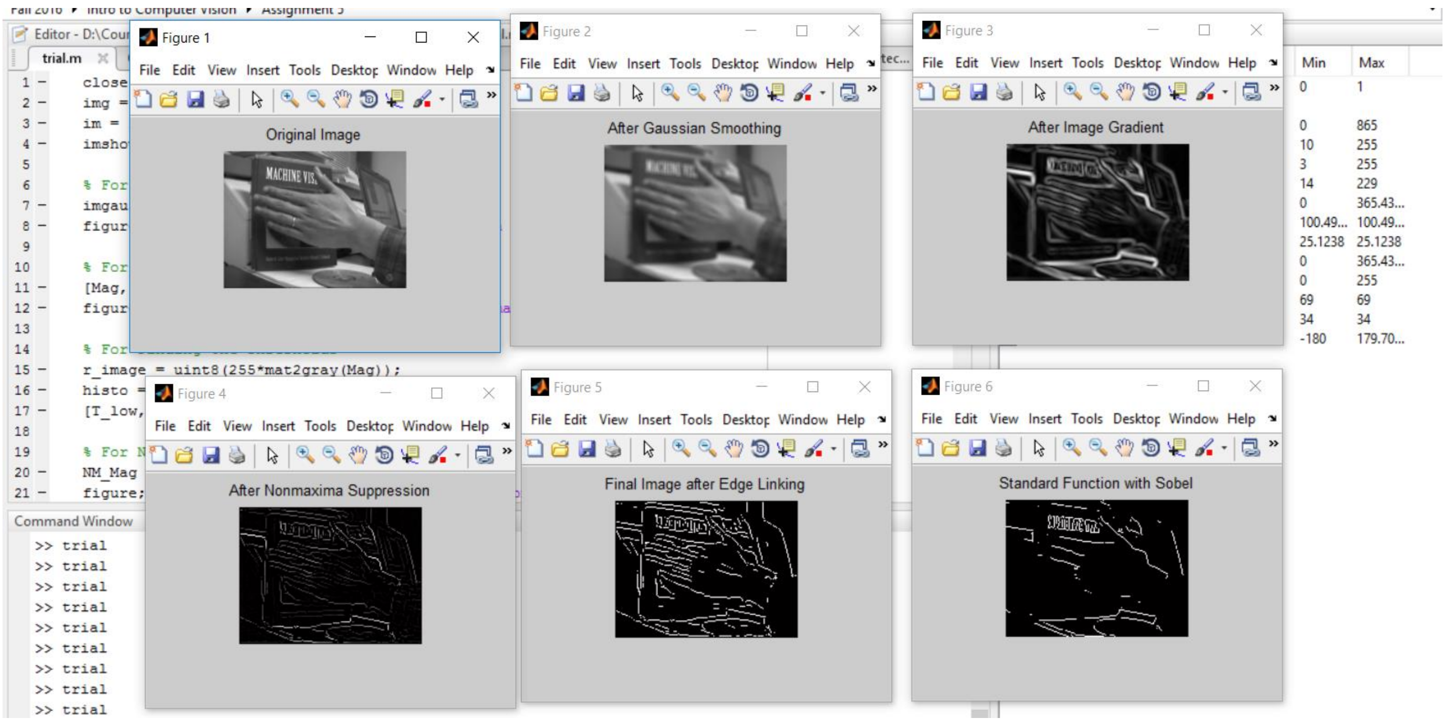
For this image, A Gaussian mask of N = 4 and Sigma = 3

3. Test1.bmp



For this image, A Gaussian mask of N = 6 and Sigma = 3

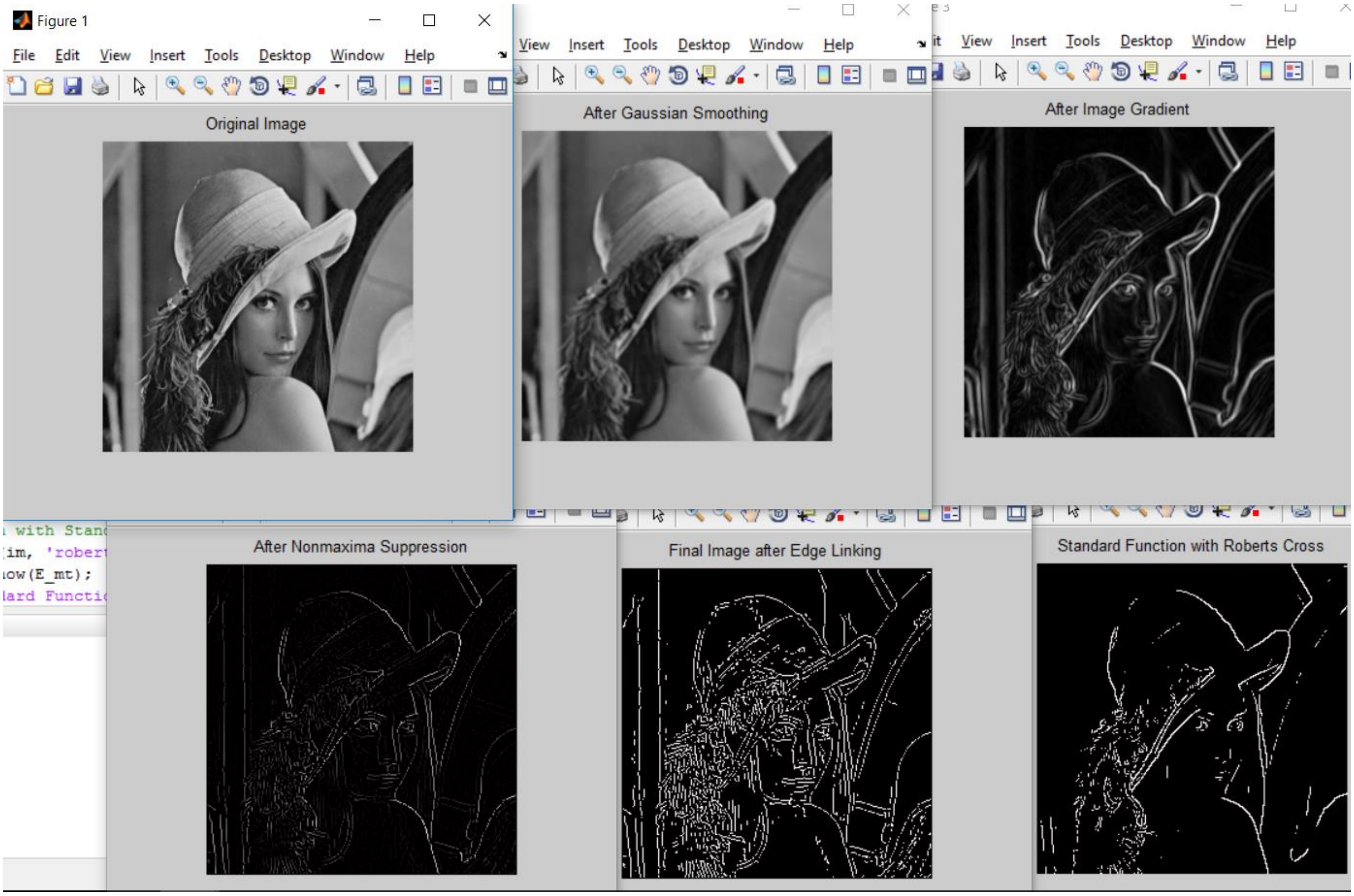
4. Pointer1.bmp



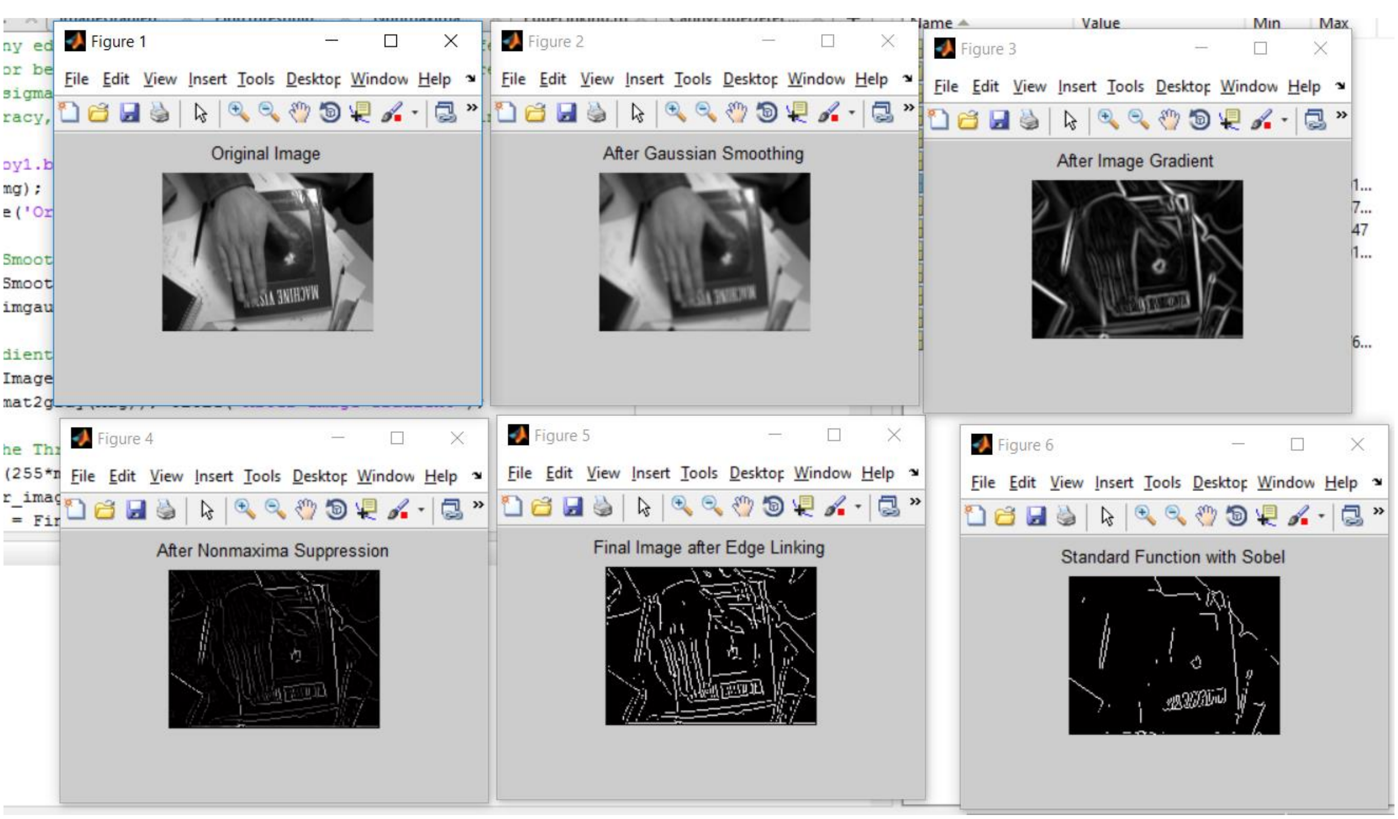
For this image, A Gaussian mask of N = 4 and Sigma = 3

Testing with other parameters (Changing the values of N, sigma and PercentageofNonEdge)

1. **For Lena (N=3, Sigma =3) (Here it is compared with Roberts Cross standard edge detector)**



2. **For Joy1 (N=3, Sigma = 3)**



3. For Pointer1(N=3, Sigma = 3)