Efe Tarhan

22002840

EEE

CS484 - Introduction to Computer Vision Homework 1 Report

Question 1:

I have written functions for dilation and erosion that take binary images and applies the described operation. The function takes the images and converts them to a Boolean array. Following that the image is padded with zeros with an amount adaptable to the size of the structuring element. The 'OR' or 'AND' operation is applied to the image for dilation and erosion using the structuring element respectively. Output of the operation is converted to a full-scale binary image where the pixel values are 0 for black pixels and 255 for white pixels. The results of applying these functions on the given image can be seen from figures, Figure 1-2, given below.

Fig 1. The dilation operation applied on the binary image.

Fig 2. The erosion operation applied on the binary image

The “cross” structuring element with the following matrix structure has been used for both morphological operations.

$$\bar{\bar{S}} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

After completing the objective functions, the image is being tried to be denoised and regenerated using a series of operations. Closure can be used as a method to thicken the lines in the image while removing noise which can be a good way for denoising. The effect of the closure operation can be seen from Figure 3.

Fig 3. Original image and its final version after applying closure operation.

Question 2:

In the second question, the task is to write a histogram function that can create the histogram of any given greyscale image and compare it with the MATLAB’s original histogram function. There are two sample images that are provided by the instructor. The results of applying the histogram function can be seen below in Figures 4,5,6 and 7.

Fig.4 First sample greyscale image for question 2

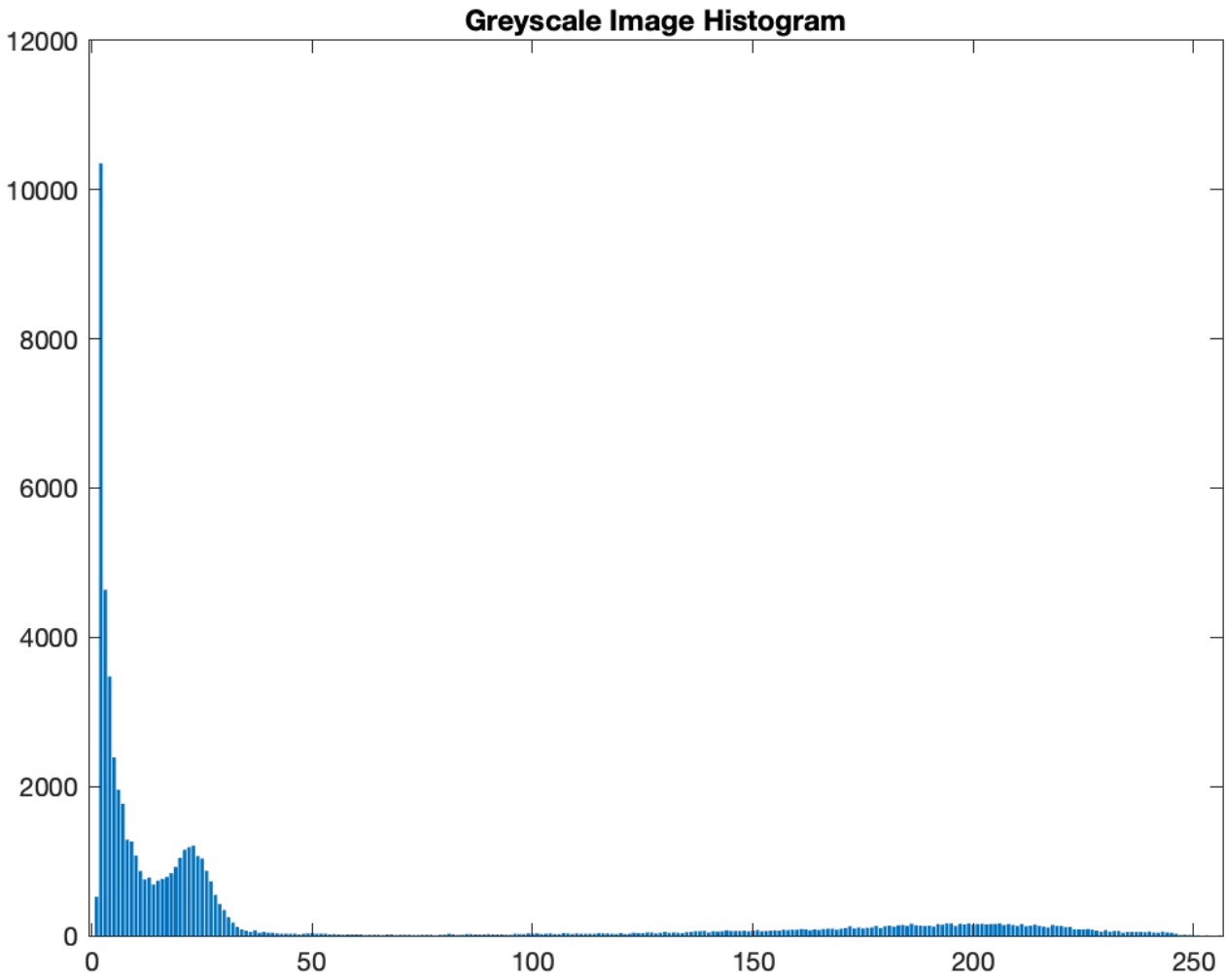

Fig.5 Histogram of the first greyscale sample image

Fig.6 Second sample greyscale image for question 2

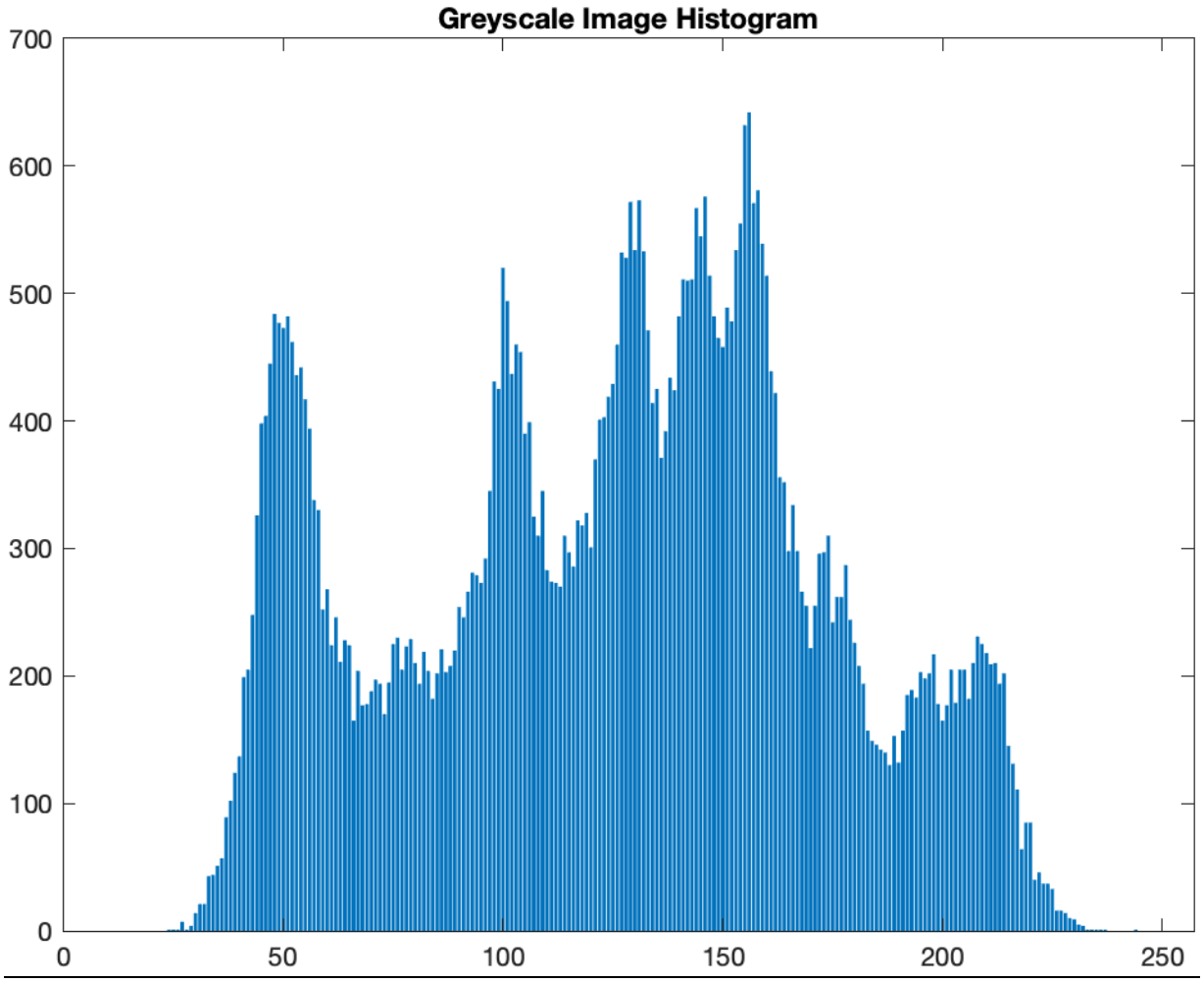

Fig.7 Histogram of the second greyscale sample image

Also, the results are compared with the MATLAB's histogram function and the results can be seen from Figure 8. By observing the shape of the histogram and pixel values it can be stated that the results are the same.

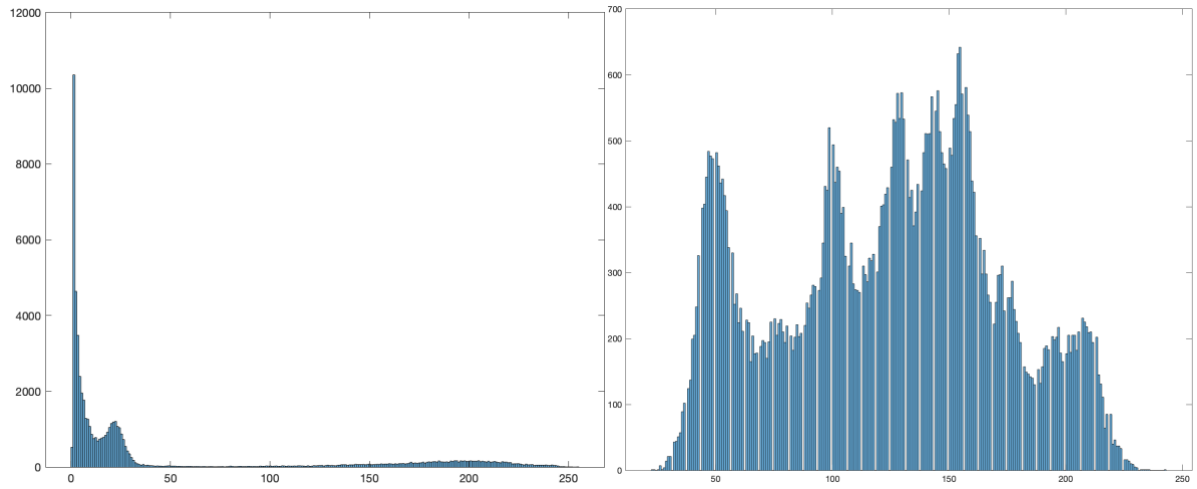

Fig.8 Results of the MATLAB's histogram function for both sample images. The histogram for the first image is on the right and second is on the left.

Question 3:

For this part of the assignment the main goal is to implement a histogram equalization function that can approximate the histogram of a given greyscale image to a uniformly distributed values of pixels. For discrete models the histogram equalization is done by using the discrete CDF of the image:

$$y = T(x_k) = 255 * \sum_{j=1}^{k} p_r(x_j)$$

Results of applying this operation to the given sample images can be seen from Figures 9 and 10.

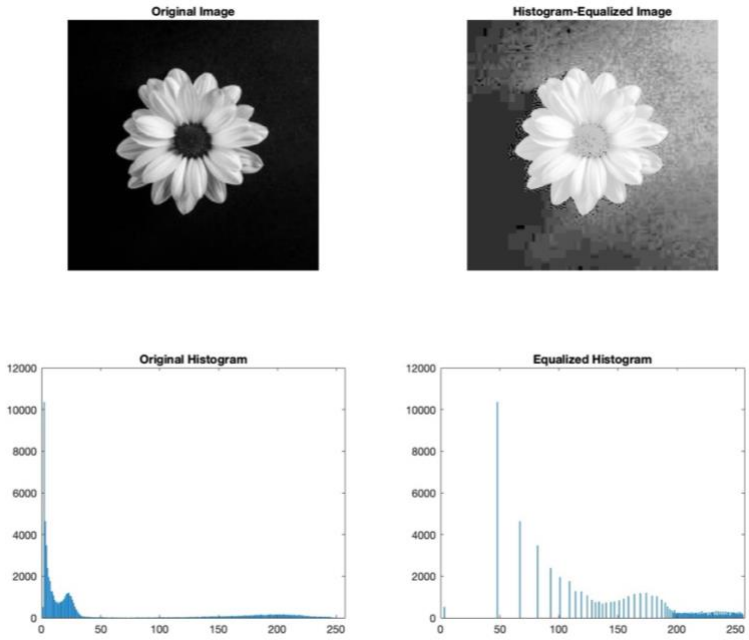

Fig.9 Results of the histogram equalization on the first image.

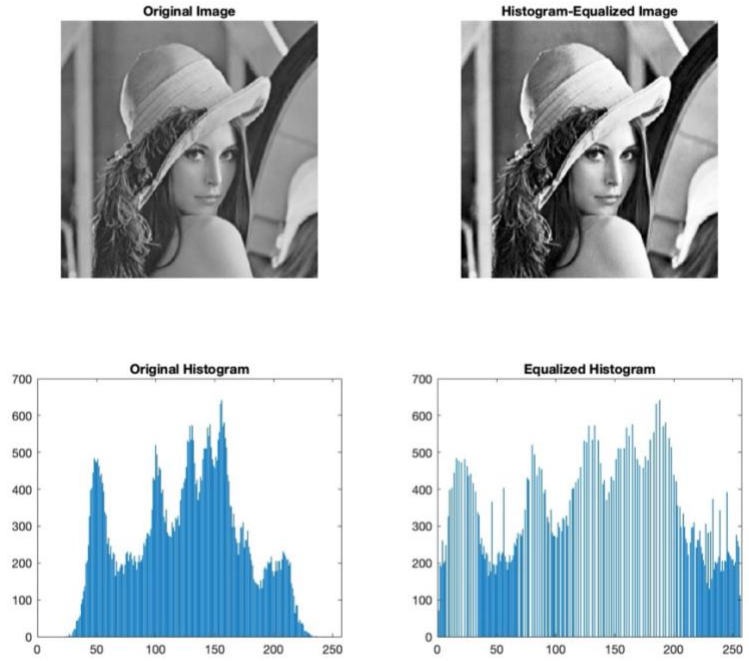

Fig.10 Results of the histogram equalization on the first image.

From the results grey level depth and details in the second image has been strengthened and hence the histogram equalization proved to be useful as the pdf of the image has become more uniform and more details can be visualized. For the first image the histogram equalization did not work as good as the second image where some distortions occurred in the image. This is since the histogram of the first image has become more uniform compared to the second image. Also, the full histogram normalization is not possible in the discrete case since the continuous differentiation operation is changed with the difference operation which can create a sparse histogram or a nonuniform one depending on the initial variance of the image. The reason why histograms are not exactly uniform is because the images are digital and hence restricted to 8-bit data storage per pixel and the equalization operation is also not continuous.

Question 4:

In this part of the homework a dynamic thresholding method called Otsu's thresholding has been applied to the given images to separate the background and foreground. Otsu's thresholding tries to find a pixel level where the interclass variances maximized and therefore intraclass variance is minimized. To find this level a search algorithm is being used and the following results are gathered. Results of applying Otsu's threshold to images can be seen from Figure 11.

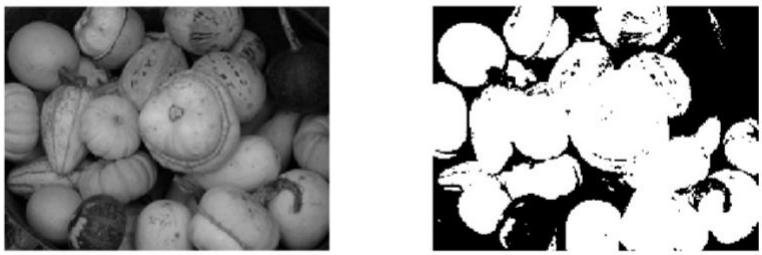

Fig.11 Original image and the image obtained by using Otsu's thresholding.

The histogram of the original image and the threshold that minimizes the interclass variance is determined and shown in the plot given in Figure 12. The optimal threshold divides the histogram into two distinguishable regions like a binary classification problem. For the first

image sample the Otsu's thresholding method is perfectly able to extract the foreground from the image.

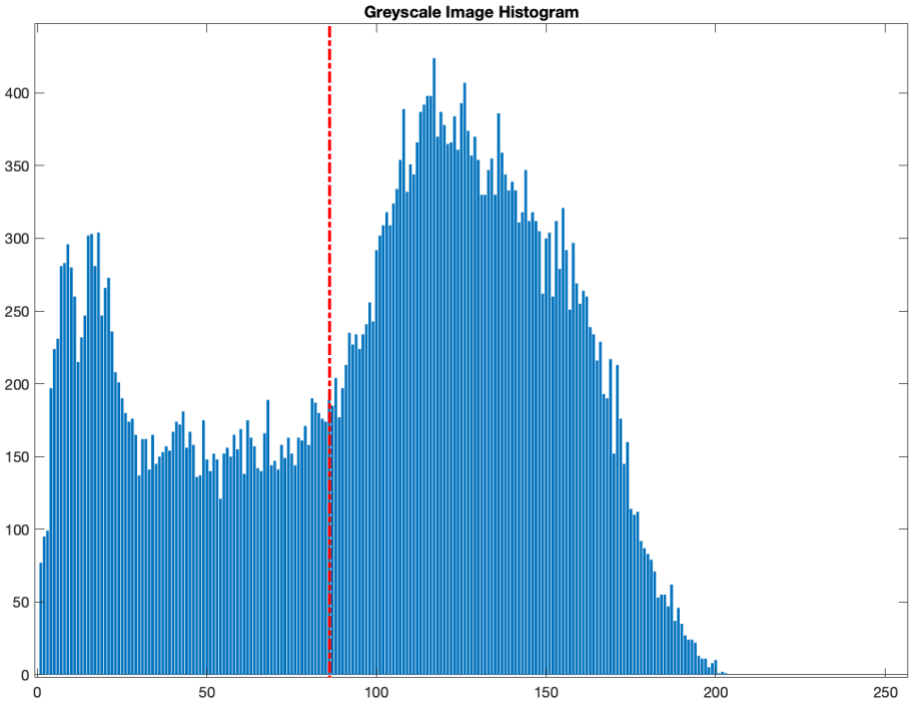

Fig.12 Histogram and optimal threshold level for the given greyscale image.

For the second example, the image is more mixed in terms of the locations and clustering of greyscale pixel values. Because of this, the Otsu thresholding does not successfully separate the foreground and background but became more sensitive to shadows and shaded regions in the image. The results can be observed from Figures 13 and 14.

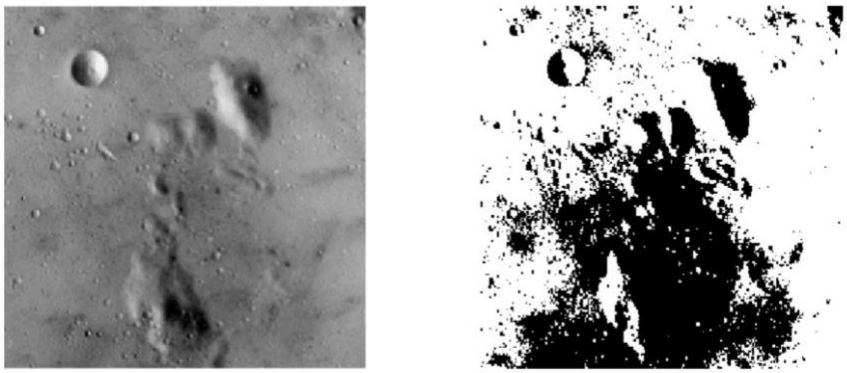

Fig.13 Original image and the image obtained by using Otsu's thresholding.

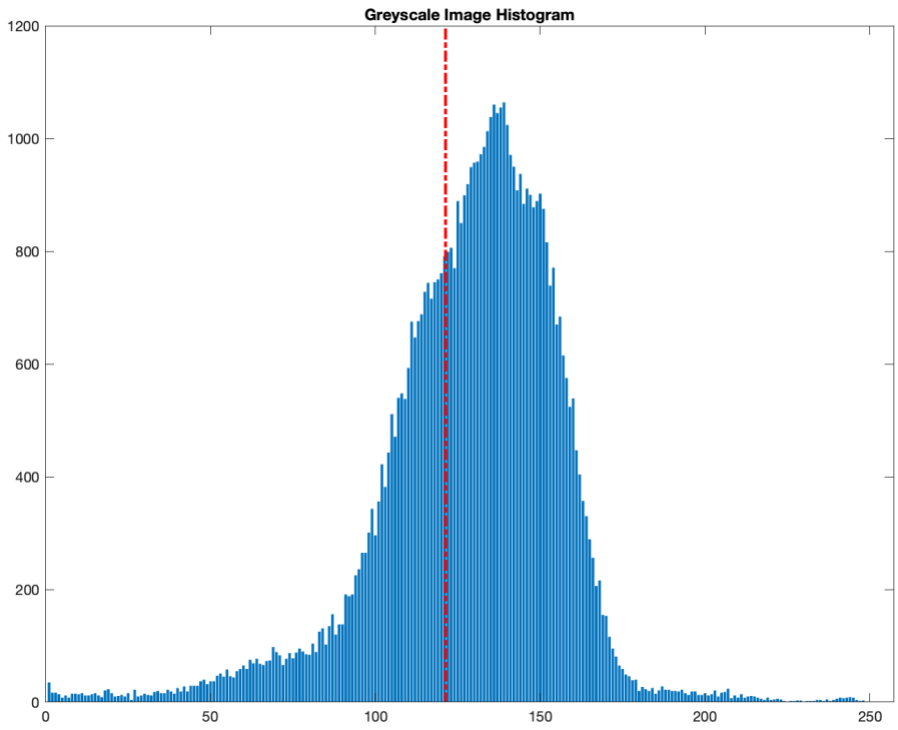

Fig. 15 Histogram and optimal threshold level for the given greyscale image.

For Otsu's method to be operatable there must be distinct peaks in the histogram, the lighting must be uniform and equal along the image and the contrast of the surfaces that must be separated must be high.

Question 5:

For this part of the homework given image will be processed through several computer vision operations and number of distinct objects will be found using several techniques. Original image that will be used for this part is shown in Figure 16.

Fig.16 The image where the count of distinct objects will be found.

First operation must be converting this image to binary format by applying a proper thresholding since the objects can be easily distinguished from the background due to high contrast. Otsu's Thresholding can be used for this task where it is highly practical for separation tasks with high contrast. The image after thresholding is shown in Figure 17.

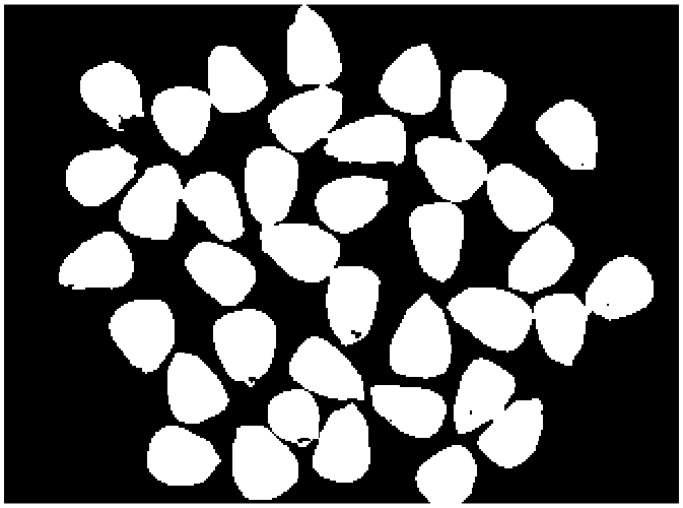

Fig.17 Binary image obtained by applying thresholding to the original image.

Following the thresholding, erosion can be used for removing the connections between the adjacent objects to be able to apply CCA (Connected Component Analysis). The erosion operation has been applied to the image for 4 times. The result of each one can be seen below in Figure 18.

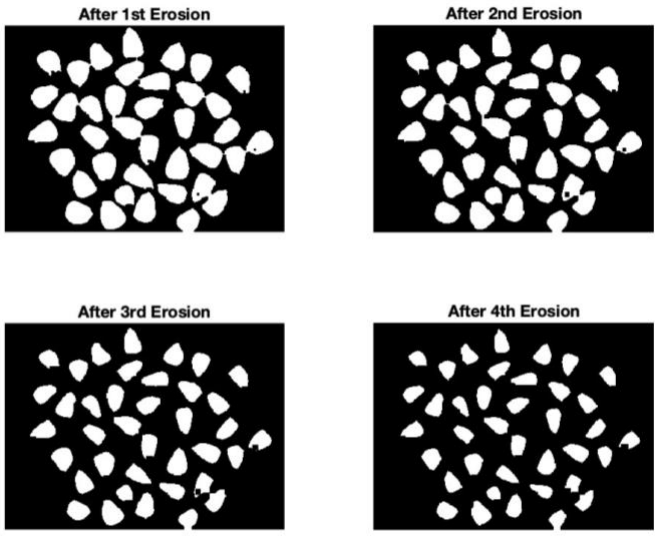

Fig.18 The binary image after four erosions.

The erosion operation has been done using a 3 by 3 matrix full of ones as structure element of the operation. Following the erosion a handwritten connected component analysis code can be applied to the binary image and number of distinct objects can be found. The result of the image after applying CCA is shown in Figure 19.

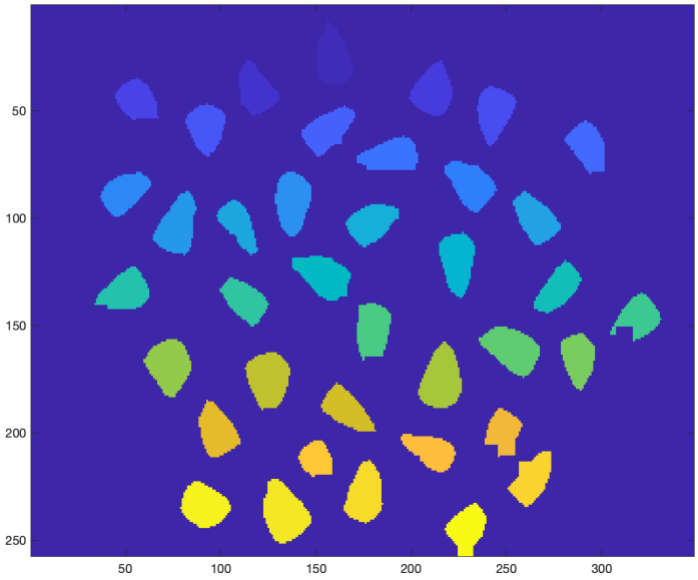

Fig.19 The result of applying connected component analysis to the binary image.

From the image each color represents a distinct object where the method numbers each distinct element from 1. By using the CCA algorithm it has been found out that there are **38** distinct objects in the image.

Question 6:

Mathematical notation of the 2D convolution operation for discrete-time data can be seen below. This equation has been implemented in MATLAB inside a function for creating a generic tool for 2D convolutions.

$$y[m,n] = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} h[i,j] \,.\, x[m-i, n-j]$$

The 2D convolution function written in MATLAB has been used for implementing edge detection on the given image in Figure 20.

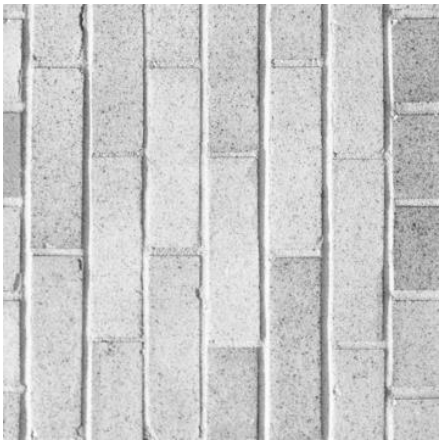

Fig.20 Sample image to be used for edge detection.

The edge detection algorithm has been used by first applying Sobel or Prewitt filters to the image to obtain the gradients S_x and S_y. Magnitude of the gradient has been obtained by using the following operation.

$$Grad_S = \sqrt{S_x^2 + S_y^2}$$

After obtaining the magnitude of the gradient the pixels corresponding to edges are filtered out using mean and variance of the magnitude of gradient. The edges are detected by setting them to 1 and non-edge pixels to 0.

$$E = Grad_S > \ mean(Grad_S) + 0.5 * std(Grad_S)$$

As a result of this operation the images in Figures 21 and 22 are the detected edges from the given image using Sobel and Prewitt kernels respectively.

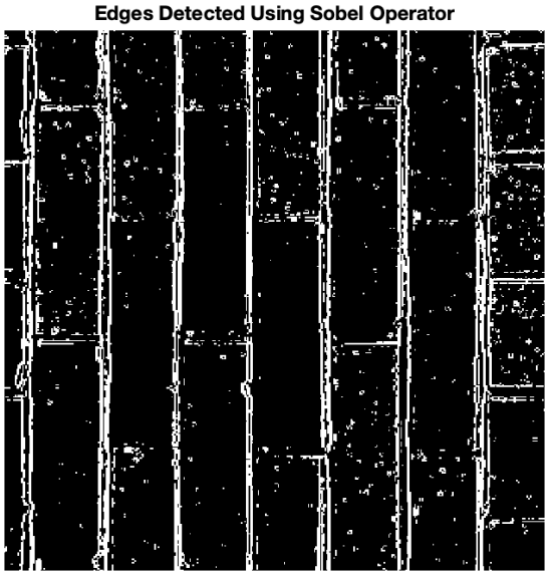

Fig.21 Edges detected using Sobel Operator

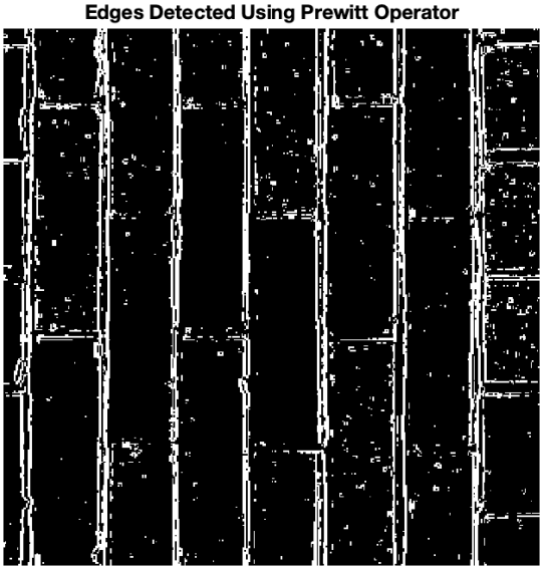

Fig.22 Edges detected using Prewitt Operator

There is dot like shapes in the image which occurs due to the sharpness of the details in the image. Performance of edge detection algorithm can be improved by applying a gaussian blur filter before applying the gradient operators. For further improvement canny edge detector can be used by involving a Laplace operator along the gradients.