Digital Image Processing Project

Submitted to: Dr. Aloke Datta

Name: SIDDHARTH RAJESH JIYANI

Roll Number: 22UCS211

Contents

Overview

My Image :

img1.jpg (Dimension : 2592 x 1728)

For processing this image to obtain the largest connected components, we will go through various steps and processes to obtain the final image shown below :

Step 1: Reading image in Matlab and storing image in img variable.

```
img = imread("./img3.jpg");
```

Step 2: Convert the image from RGB to Gray Scale

Here we convert the RGB image to grayscale using the formula:

```
Gray = 0.29 * R + 0.59 * G + 0.11 * B
```

Result of conversion from RGB to Gray Scale:

Step 3: Add 30% Salt and Pepper noise to the image.

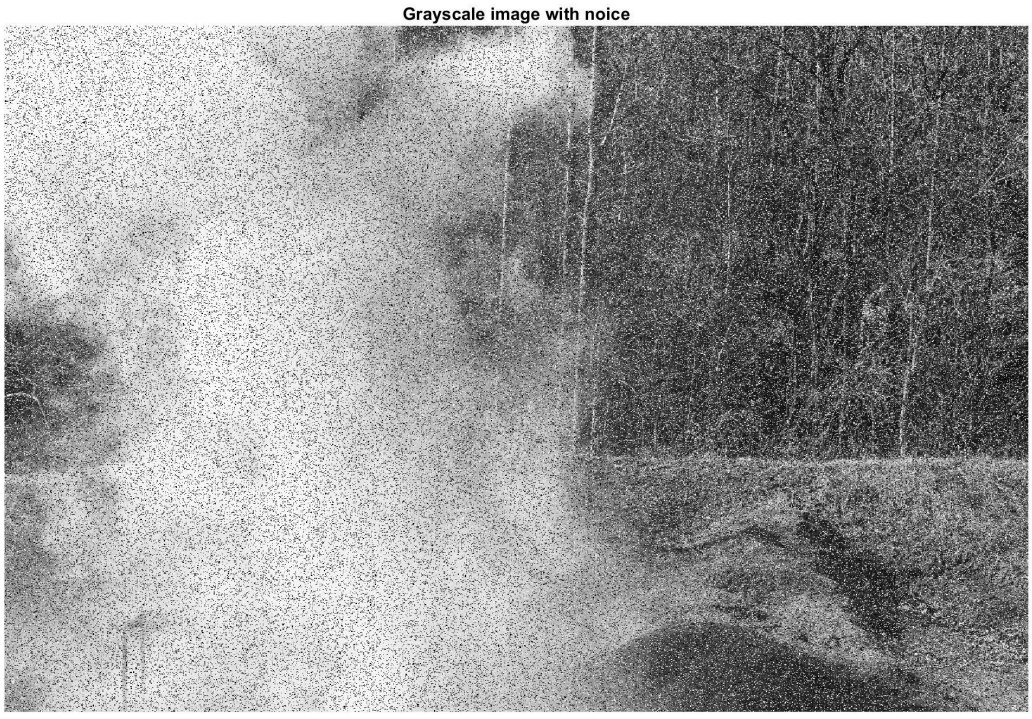
The PDF of salt-and-pepper noise is given by:

$$p(z) = \begin{cases} P_s & \text{for } z = 2^k - 1 \\ P_p & \text{for } z = 0 \\ 1 - (P_s + P_p) & \text{for } z = V \end{cases}$$

Our aim is to add 30% Salt and pepper noise so for that purpose we will be adding 15% Salt (**P_s = 0.15**)noise and 15% Pepper noise (**P_p = 0.15**) and hence **1 - (P_s +P_p) = 0.70**

Now according to the probabilities we assign values as maximum intensity (black i.e., 0) and minimum intensity (white i.e., 255) which will eventually add noise to out grayscale image.

Image after adding noise:



Step 4: Remove noise using Adaptive median filter from the previous image.

Note: We here chose adaptive mean filter because this filter performs well in images with salt and pepper noise.

To perform this filtering, we follow the given algorithm:

Declaring variables:

$$z_{\min} = \text{minimum intensity value in } S_{xy}$$
$$z_{\max} = \text{maximum intensity value in } S_{xy}$$
$$z_{\text{med}} = \text{median of intensity values in } S_{xy}$$
$$z_{xy} = \text{intensity at coordinates } (x, y)$$
$$S_{\max} = \text{maximum allowed size of } S_{xy}$$

The adaptive median-filtering algorithm uses two processing levels, denoted level A and level B, at each point (*x,y*):

$$\begin{aligned}
\text{Level } A: \quad & \text{If } z_{\min} < z_{\text{med}} < z_{\max}, \text{ go to Level } B \\
& \text{Else, increase the size of } S_{xy} \\
& \text{If } S_{xy} \leq S_{\max}, \text{ repeat level } A \\
& \text{Else, output } z_{\text{med}}. \\
\text{Level } B: \quad & \text{If } z_{\min} < z_{xy} < z_{\max}, \text{ output } z_{xy} \\
& \text{Else output } z_{\text{med}}.
\end{aligned}$$

where S_{xy} and S_{max} are odd, positive integers greater than 1.

Image obtained after filtering:

Denoised Image

Step 5: Using Otsu's thresholding technique to make the binary image of the denoised image from Step 4.

Otsu's approach is entirely driven by the normalized histogram of the image. Follow the below steps:

It can be observed that $P_1 m_1 + P_2 m_2 = m_G$ and $P_1 + P_2 = 1$

- Let $\{0,1,2,\dots,L-1\}$ denote the set of L distinct integer intensity levels in a digital image with MxN pixels.
- The normalized histogram has components $p_i = \frac{n_i}{MN}$ which follows $\sum_{i=0}^{L-1} p_i = 1$ with $p_i \geq 0$.
- Suppose we select a threshold $T(k) = k, 0 < k < L-1$ and use it to threshold the input image into two classes c_1 and c_2. i. e c_1 has pixels in the intensity range $[0,k]$ and c_2 has pixels in the intensity range $[k+1, L-1]$
- The probability that a pixel is assigned to class c_1 is given by $P_1(k) = \sum_{i=0}^{k} p_i$ and that of class c_2 is given by $P_2(k) = \sum_{i=k+1}^{L-1} p_i = 1 - P_1(k)$
- The mean intensity values of the pixel in c_1 is given by $m_1(k) = \frac{\sum_{i=0}^{k} i\, p_i}{\sum_{i=0}^{k} p_i}$ and c_2 is given by $m_2(k) = \frac{\sum_{i=k+1}^{L-1} i\, p_i}{\sum_{i=k+1}^{L-1} p_i}$
- Average intensity of entire image is given by $m_G = \sum_{i=0}^{L-1} i\, p_i$

σ_B^2 is the *between-class variance*, defined as

$$\sigma_B^2 = P_1 (m_1 - m_G)^2 + P_2 (m_2 - m_G)^2$$

Now further simplifying the above formula we get,

$$\sigma_B^2 = P_1 P_2 (m_1 - m_2)^2$$

Our aim in Otsu's thresholding is to maximize the σ^2_B value. Then, the optimum threshold is the value, k*, that maximizes $\sigma^2_B(k)$:

$$\sigma_B^2 (k^*) = \max_{0 \le k \le L-1} \sigma_B^2(k)$$

Once k* has been obtained, input image f (x,y) is segmented as before:

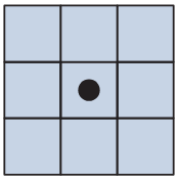
$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > k^* \\ 0 & \text{if } f(x,y) \le k^* \end{cases}$$

Binary Image obtained after thresholding:

Step 6: Using Morphological Analysis to remove small objects.

In this step, we perform two morphological operations **(1) Erosion** and then **(2) Dilation** after performing erosion.
We use a structuring element and perform erosion to remove small unwanted objects and then using the same structuring element we perform dilation to recover a few lost details in the image.

Structuring element :



Erosion:

$$A \ominus B = \left\{ z \middle| (B)_z \subseteq A \right\}$$

Erosion reduces the foreground pixels and removes the small components in the image. (Our foreground pixels are white and background pixels are white)

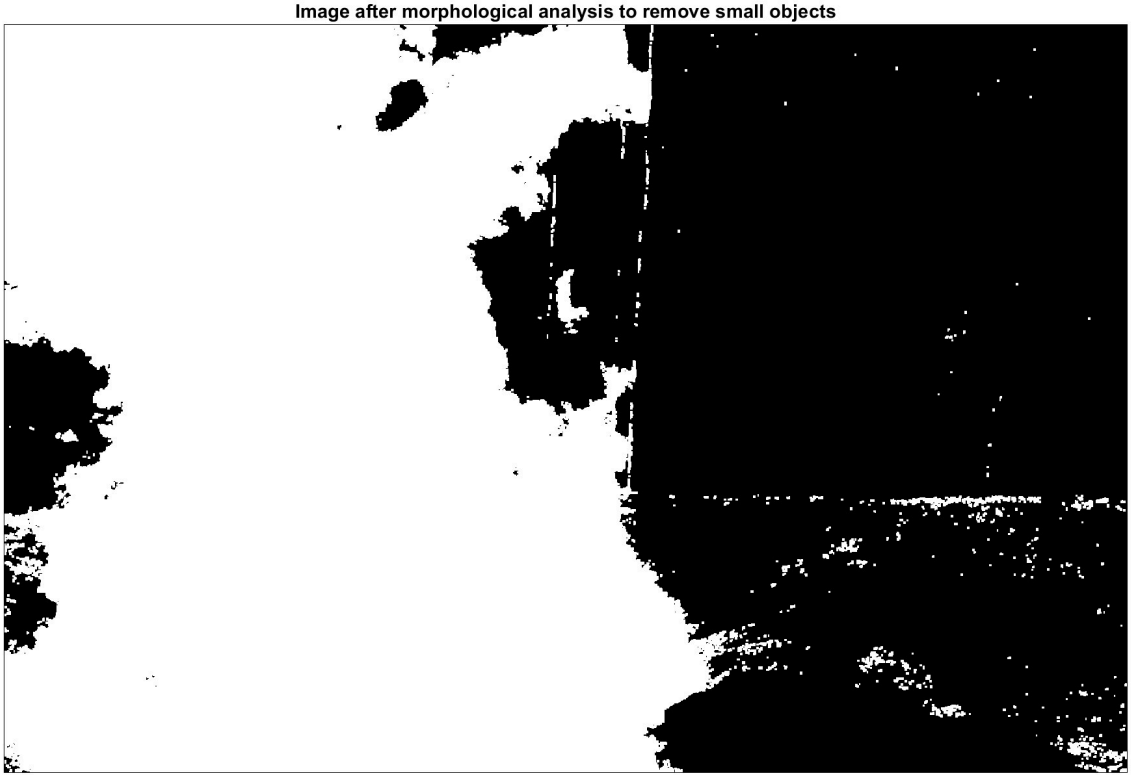
Image after performing erosion:

Dilation:

$$A \oplus B = \left\{ z \middle| (\hat{B})_z \cap A \neq \varnothing \right\}$$

Unlike erosion, which is a shrinking or thinning operation, dilation “grows” or “thickens” objects in a binary image. The manner and extent of this thickening are controlled by the shape and size of the structuring element used.

Image after perforation Dilation on the eroded image:



Step 7: Use Morphological Analysis to find the largest connected components.

The objective is to start with $\mathbf{X_0}$ and find all the connected components in image. The following iterative procedure accomplishes this:

$$X_k = \left(X_{k-1} \oplus B\right) \cap I \qquad k = 1, 2, 3, \ldots$$

The algorithm follow:

Initialize: $X_0 = p$,

where p belongs to any pixel in desiged region

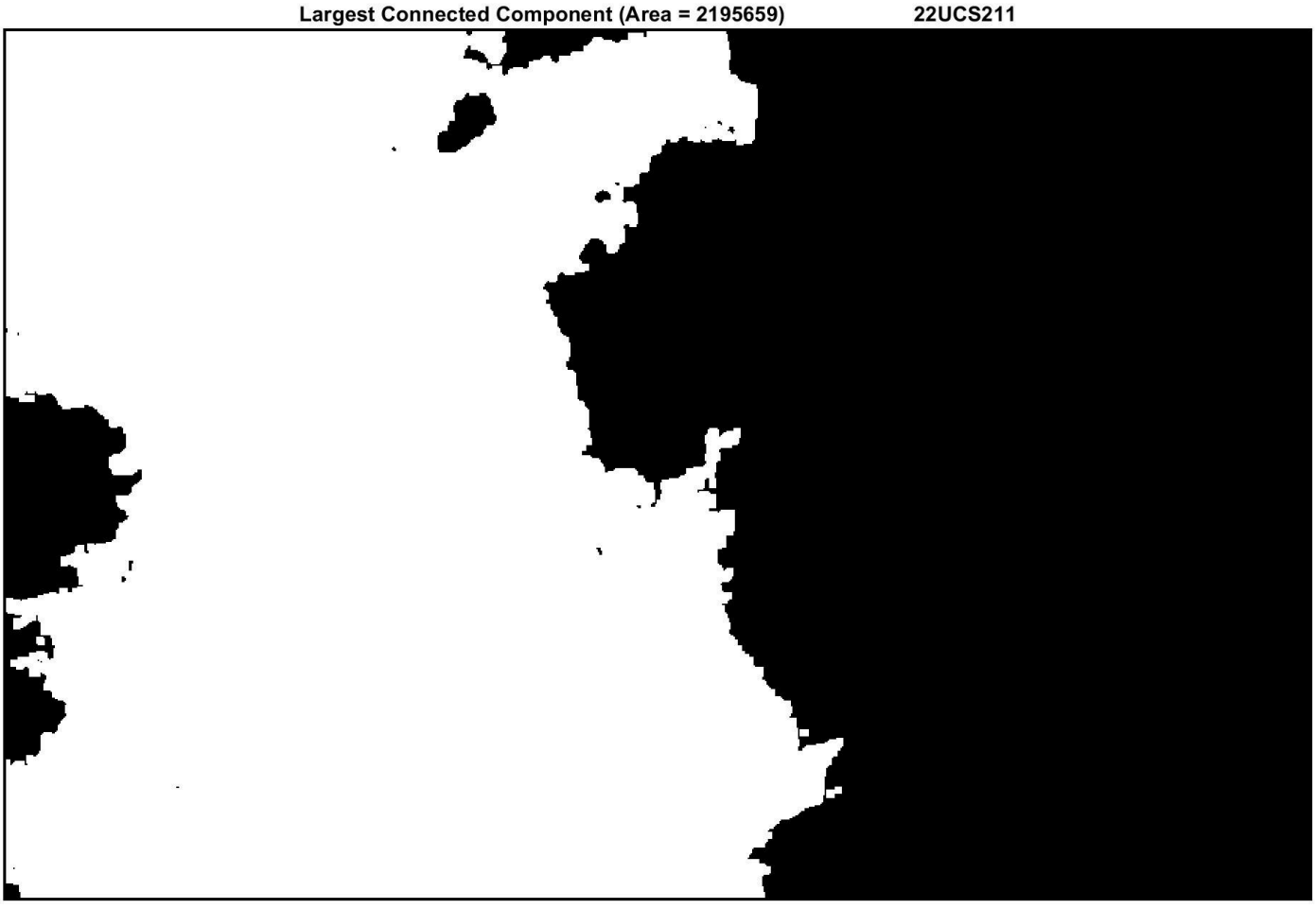
Followed by performing repetitively:

$$X_k = (X_{k-1} \oplus B) \cap A$$

Until:

$$X_k = X_{k-1}$$

The resultant image :



Step 8: Find the area of the largest connected component

The pixel count of the largest connected component gives the desired area completed all the steps of the project.
We traverse through the whole image and add 1's in the image to obtain the desired area.

Original image size = 2592 x 1728 = 4478976
The area obtained = **2195659 pixels**.

********** Thank you**********

References used:

1. Digital Image Processing, fourth edition by **Rafael C. Gonzalez • Richard E. Woods.**
2. Lecture slides **Dr. Aloke Datta** provided.