Lab 1: Basic Image Handling

Kamala Rai Danuwar

13 [20070291]

BE Computer

Everest Engineering College

16/02/2024

Summary

In this lab we learnt about the basic handling of the image. We read the color image and converted it to a gray-scale image and saved it and extracted the grayscale values to a 2D integer array and applied appropriate algorithms on the array to solve our queries.

I created a binary image from a grayscale image by thresholding using different threshold values:
1) r =128

2) mean intensity as threshold

3) median intensity as threshold

I. PROBLEM DESCRIPTION

In this practical, we are required to write a code in any code editor and language of our own preferences to read a color image, convert it to gray-scale by averaging the RGB values and save the resulting image. I used Net Beans IDE for writing the Java code. Initially we extracted the grayscale values to 2D integer array and applied the appropriate processing algorithms on the array and converter the array into an array. In this lab, I attempted three questions: 1) creation of a binary image from a grayscale image by thresholding: using an arbitrary threshold of 128. 2) conversion the grayscale image to binary using mean intensity as the threshold. 3) modification of program for question 2 using median intensity as the threshold. I solved the questions by creating different functions for each of the questions and then called the function from the main.

II. THEORETICAL BACKGROUND

- Thresholding transformation can be written in a simple form as:

$$s = \begin{cases} c_l & \text{if } r \leq \theta \\ c_h & \text{otherwise} \end{cases}$$

where r and s are input and output pixel intensities, θ is the threshold, c_l and c_h are constants.

- The main methods used in this lab are as follows:
    - threshold(int[][] f, int r) : it applies thresholding operation to a 2D integer array represented image ‘f’ taking parameters ‘f’:original image and ‘r’:threshold value as inputs. Pixels with intensity greater than ‘r’ are set 255(white) and pixels with intensity values less than or equal to ‘r’ are set to 0(Black) and the thresholded image ‘F’ is returned to main for display.

    - meanIntensityAsThreshold(int[]][] f) : this method calculates the mean intensity of the input image ‘f’ and uses this mean intensity as the threshold value for creating a binary image. It calculates the sum of all the pixels in the input image and divides it by the total number of pixels to find mean intensity. Then each pixels are compared with the mean intensity. Pixels with intensity greater than mean are set 255(white) and pixels with intensity values less than or equal to mean are set to 0(Black) and the thresholded image ‘F’ is returned to main for display.

    - medianIntensityAsThreshold(int[]][] f) : this method calculates the median intensity of the input image ‘f’ and uses this median intensity as the threshold value for creating a binary image. It first copies all the pixel values of the input image into 1D array and sorts to find the median intensity. Then each pixels are compared with the median intensity. Pixels with intensity greater than median are set 255(white) and pixels with intensity values less than or equal to median are set to 0(Black) and the thresholded image ‘F’ is returned to main for display.

- main(String[] args): in this method, ‘Image’ object is created using a file path pointing to an image file and the pixel array of the image is obtained using the ‘getPixelArray()’ method.

III. RESULTS AND LESSON LEARNT

To test my program I compared the original image and the converted image side by sides and then concluded using the methods theoretical significances.

The result for three operations were visually different from each other and thus learnt that the different in the threshold value results in different visual impacts.

PROGRAM LISTING

Function for r=128 as threshold value

```
public static int[][] threshold (int[][] f, int r) {
          int [][] F = new int [f.length][f[0].length];
            for(int x=0;x<f.length;x++) {
                for(int y=0;y<f[0].length;y++) {
                    if(f[x][y]>r) {
                       F[x][y] = 255;
                    }
                    else {
                       F[x][y] = 0;
                    }
                }
            }
            return F;

        }
```

Function for using mean intensity as threshold

```
        public static int[][] meanIntensityAsThreshold (int[][] f) {
            int [][] F = new int [f.length][f[0].length];

           // calculation of mean intensity
            int sum=0;
            for (int x = 0; x < f.length; x++) {
                for (int y = 0; y < f[0].length; y++) {
                   sum += f[x] [y];
                }
            }

            int meanIntensity = sum / (f.length * f[0].length);

           //using mean intensity as threshold

            for(int x=0;x<f.length;x++) {
                for(int y=0;y<f[0].length;y++) {
                    if(f[x][y]>meanIntensity) {
                        F[x][y] = 255;
                    }
                    else {
                        F[x][y] = 0;
                    }
                }
            }

           return F;
        }
```

Function for using median intensity as threshold

```
        public static int[][] medianIntensityAsThreshold (int[][] f) {
          int [][] F = new int [f.length][f[0].length];

          //1D array to store pixel values

          int[] pixelValues = new int[f.length * f[0].length];
```

```
int index=0;

//copy pixel va;ues to 1D array

for (int x = 0; x < f.length; x++) {
    for (int y = 0; y < f[0].length; y++) {
        pixelValues[index++] = f[x][y];
    }
}

// array sort

Arrays.sort(pixelValues);

//finding median index

int medianIndex = pixelValues.length / 2;

//finding meddian intensity
int medianIntensity = pixelValues[medianIndex];

// using median intensity as threshold

for(int x=0;x<f.length;x++) {
    for(int y=0;y<f[0].length;y++) {
        if(f[x][y]>medianIntensity) {
            F[x][y] = 255;
        }
```

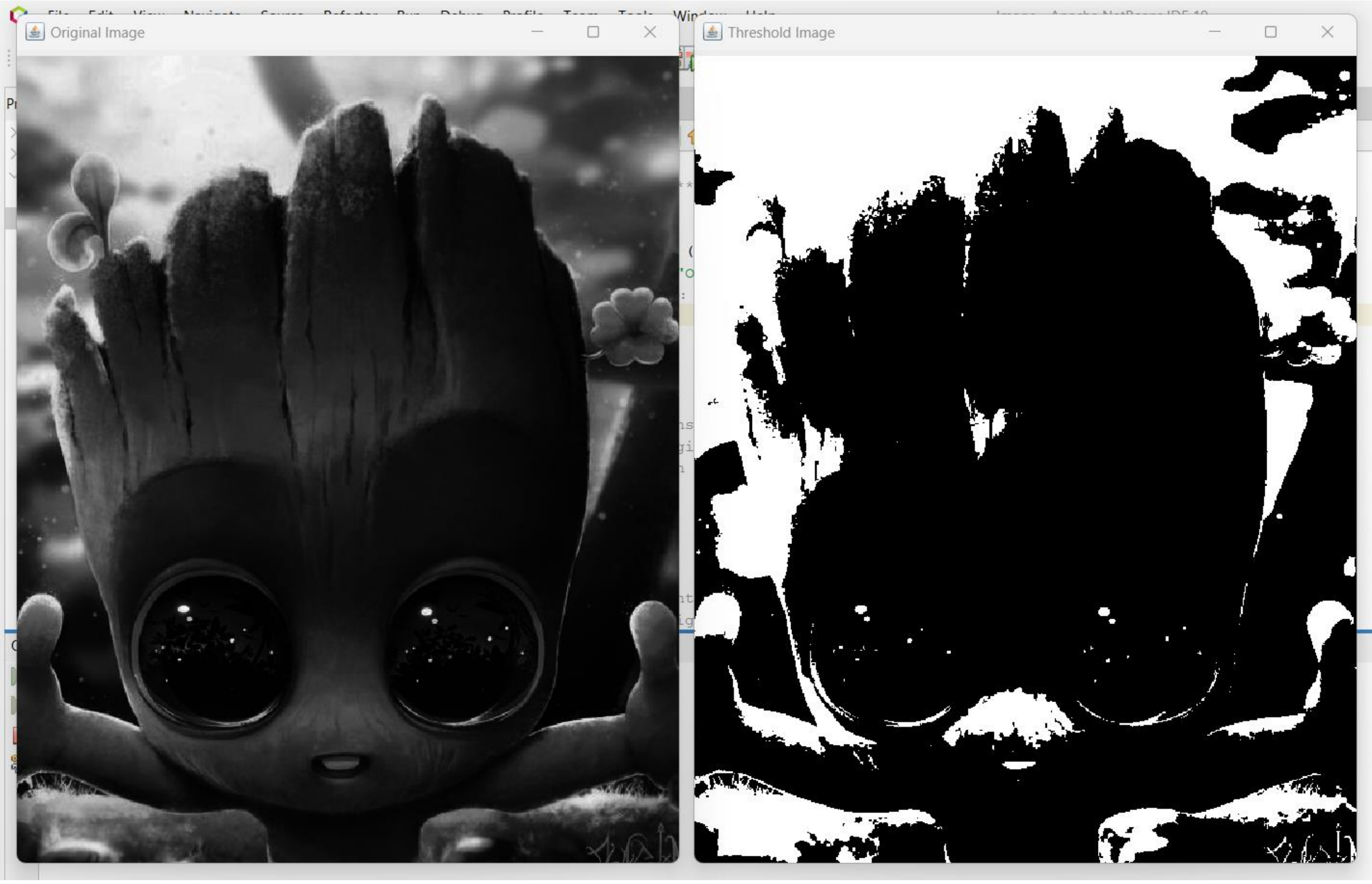
```
                    else {
                        F[x][y] = 0;
                    }
                }
            }
            return F;
        }
```

Main method

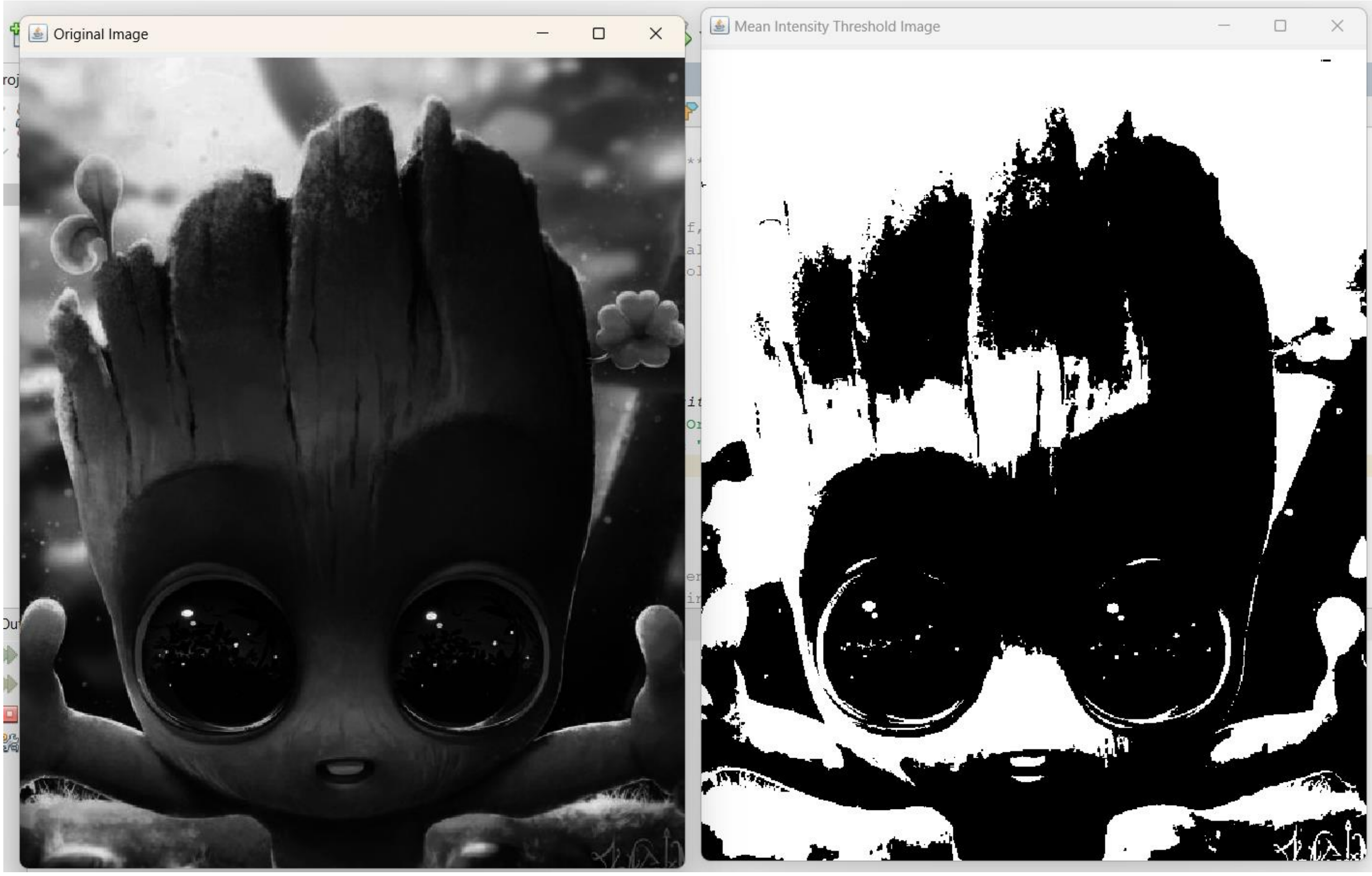
```
public static void main(String[] args) {


            Image img = new Image("C:\\Users\\Kamala Rai 
Danuwar\\Pictures\\Iam grut.jpg");
            int[][] f = img.getPixelArray();
    //QUESTION 1
    /*
           int [][] F = threshold (f,128);
           Image.display(f, "Original Image");
           Image.display(F, "Threshold Image");
            */



    //QUESTION 2
    /*
            int [][] F = meanIntensityAsThreshold(f);
            Image.display(f, "Original Image");
            Image.display(F, "Mean Intensity Threshold Image");
*/

    //QUESTION 3
    /*
            int[][] F = medianIntensityAsThreshold(f);
            Image.display(f,"Original Image");
            Image.display(F, "Median Intensity Threshold Image");
    */

    }
```

OUTPUTS

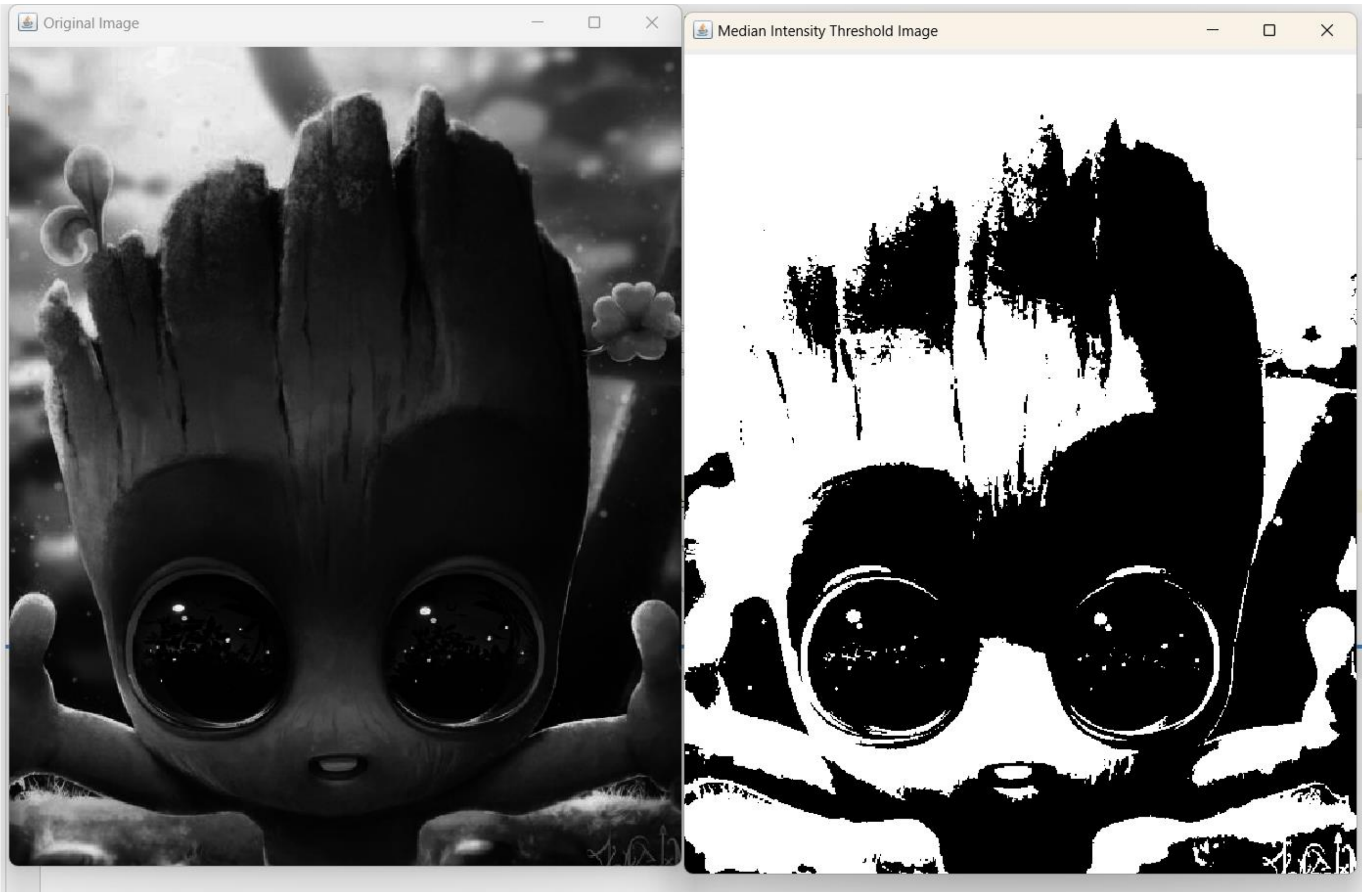
When using r=128 as threshold value:



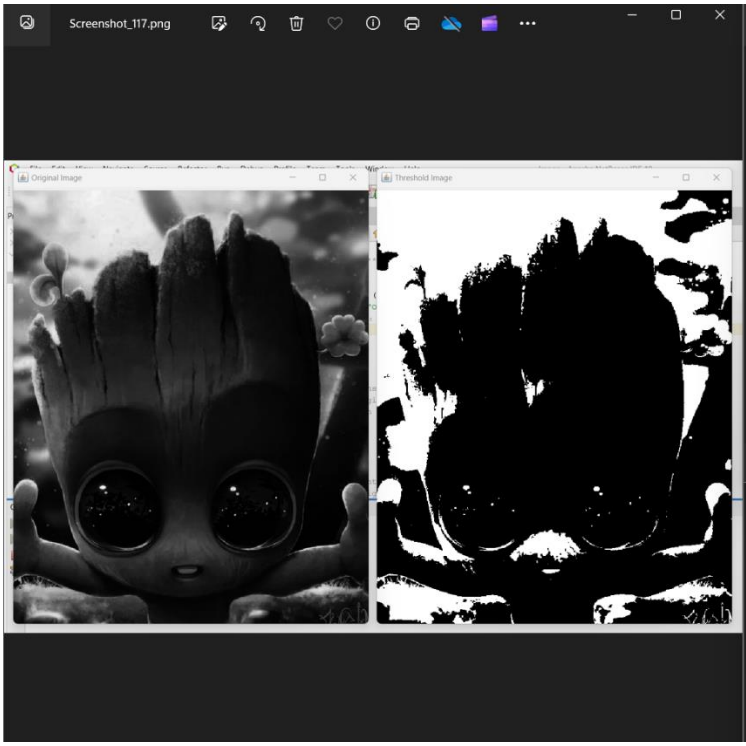
When using mean intensity as threshold:

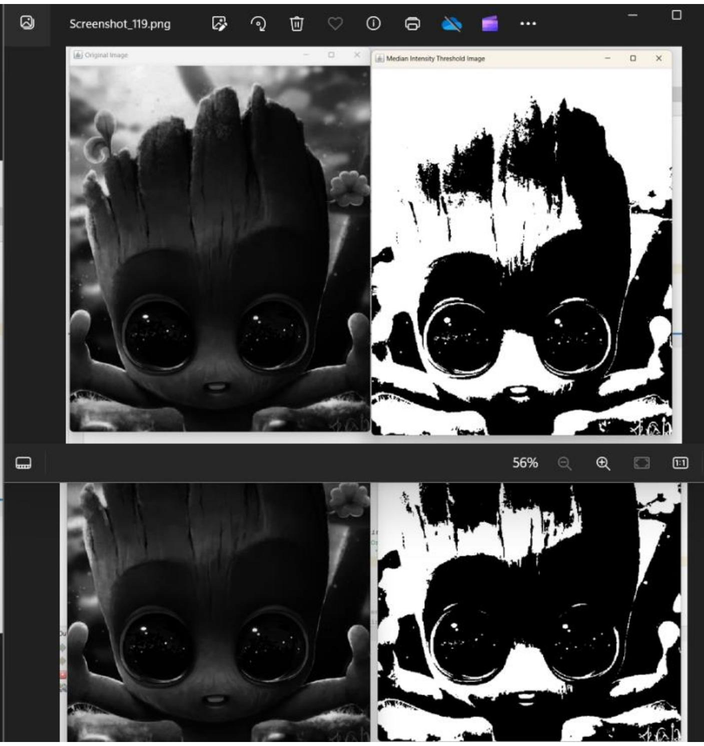


When using median intensity as threshold:

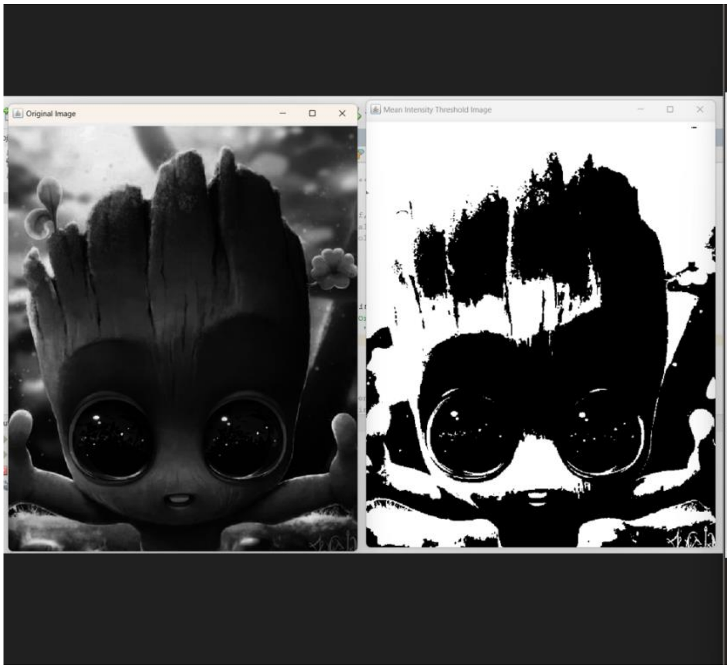


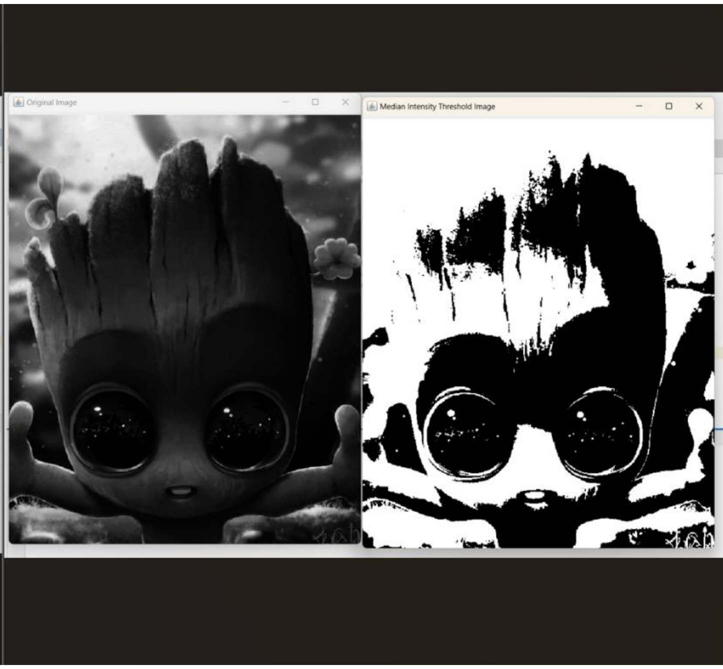
comparing the three images using different threshold values:





When mean and median values are as threshold values.


Original Image
Mean Intensity Threshold Image


Original Image
Median Intensity Threshold Image