**Name: Pravin Reddy Sereddy**

**Email: spravin1@binghmaton.edu**

**Purpose of the Project:**

The main purpose of the project is getting the knowledge of how the image data is stored in a computer and to get the knowledge of image enhancement. By image enhancement we can get the clear image from the bad images by histogram equalization. So, we can get reliable images for different types of analysis like x-ray images.

Another purpose of the project is to detect the regions in the image by the applying the image threshold value to the image for converting the grey image to binary image. From the binary image we use four connect algorithm to detect the regions in the image.

And all these methods, implementations and results are explained in the below section

**Programs, Required environment setup and Running the Programs:**

There are 3 program files in this zipped file

**P1.cpp** is for the image grey scale transformation to create the negative images and print out the histogram of original image and negative image

**P2.cpp** is for image enhancement by applying the histogram equalization for better visual quality

**P3.cpp** is for image conversion from original given image to binary image by applying the threshold algorithm and detecting the regions by the 4 connect component algorithm

Required Environment setup:

* Opencv 3.4
* Linux based system with g++ compiler for compiling the c++ programs

Running the program:

Select the above program required and compile the program in the terminal with above environment

For compiling:

**g++ -ggdb filename.cpp -o filename ‘pkg-config –cflags –libs opencv’**

For running:

**./filename image**

**Programming Design**

**1. Image grey scale transformation and enhancement**

**Methods used:**

1. Algorithm used for grey scale transformation

* First load the image into to the program
* Apply the greyscale transformation function which is **(x = 255 -y) where** x is the new intensity value and y is the old intensity value

1. Algorithm used for the histogram representation

* First load the image into program
* And make array with size of 255 and initialize every value with 0
* For every pixel in the image take the intensity value from the image pixel and add 1 to the previous array of the pixel value
* And plot the histogram graph from the array values starting from the 0 to 255 by drawing the lines form the bottom of the window to the array position value

1. Algorithm used for the image enhancement (Histogram equalization)

* First load the image into program
* From the above procedure get the image histogram of the image
* From the histogram array get the probability of each intensity value in the image and that can be done by dividing the total number of each pixel intensity value in the image with the total number of the pixels
* Now get the cumulative probability of the intensity values from the previous probability of intensity values
* Now multiply with 255(because we are having the pixel intensity from 0 to 255) to cumulative probability values and we get the intensity values
* Then map the new intensity values with the old intensity values and replace it in the image
* And the new image is the histogram equalized image which is the enhanced image

**2. Region detection: finding the largest region, medium region, and the smallest**

**region of the given image**

**Methods Used:**

1. Algorithm used for the threshold value **(Global threshold Value)**

* First assume a threshold value of an image is around 128 or the mean value of the pixel intensities of an image
* Now divide the intensities values into two groups called group1 for the intensity values less than the threshold value and group2 greater and equal intensity values
* And do the mean of the group1 and group2 and in each group mean denominator I have additionally taken all the zero intensities values and added to the denominator of the mean in each group for better thresholding value
* And do the mean again now for the group1 and group2 and we get the new threshold value
* Now compare the new threshold value with the old threshold value and if they are same taking the new threshold value as the final threshold value
* If the threshold value is not equal do this procedure again with the new threshold value is the assumption value. Do this until we get the same threshold values.

1. Binary Image conversion

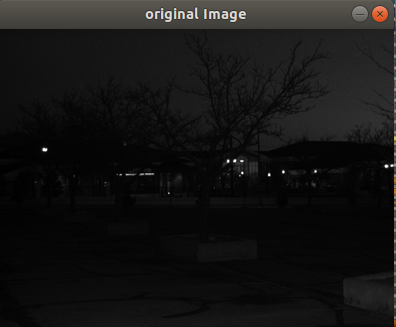
* First load the image in to the program
* From the above threshold algorithm get the threshold value
* Now for the all the pixels above the threshold value are converted to 255 and all the pixels below the threshold value are converted to 0

1. 4 Connect component algorithm and labelling

* In this algorithm I have taken row by row approach
* For the given image I have taken binary image form the threshold value
* Now I have traversed the image pixels by row by row with a counter variable
* If any of the pixel value is 255 then I have incremented the counter and replace the 255 with the counter value or the check if the above or previous element of the pixel not equal to 0 or 255 then if not, then replace the pixel value with value.
* If there is any conflict for the above value and with previous value, then replace the all the pixels before it with the minimum value of the pixel
* And traverse all the pixels like this
* For labelling the largest object in the image take the maximum number of pixels values have same value and fill the maximum object with the grey level of 200
* For labelling the largest object in the image take the minimum number of pixels values have same value and I have taken at least minimum number of pixels should be greater than or equal to the 100. So, that we can get rid of the noise data and fill the minimum object pixels with the grey level 60
* And fill all the objects between the minimum and maximum values with the grey level 120 as medium objects.

**Results:**

1. grey scale transformation



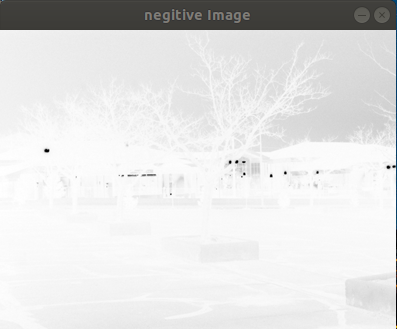
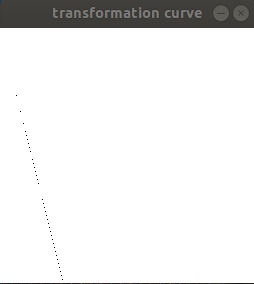


Image transformation Curve for the negative image vs original image

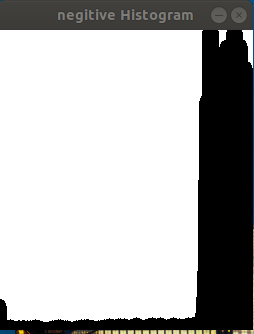


X axis is the original image

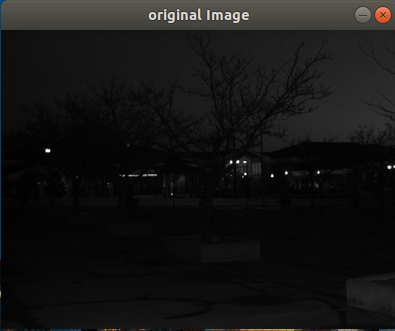
Y axis is the negative image

1. Histograms of the images



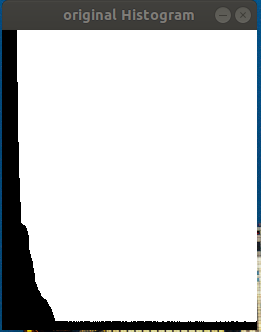


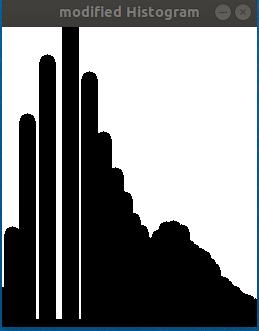
1. Histogram equalization





Histograms of images



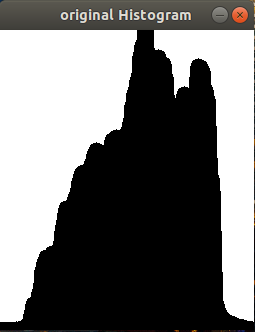


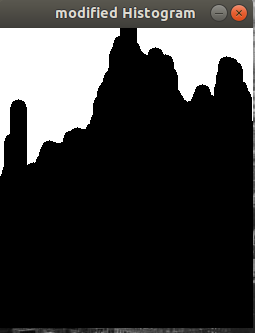
1. Histogram equalization to the images





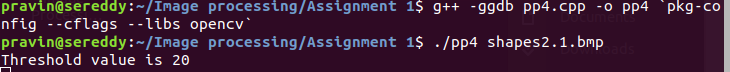
Histograms of the images



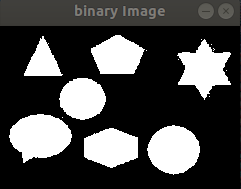


2. Region detection

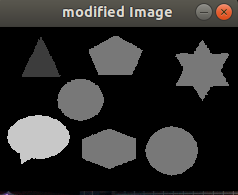
1. Threshold of the image for Figure 3 test image



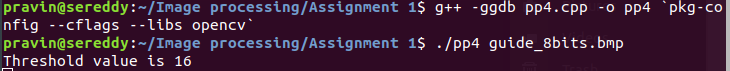
1. Binary image of the Figure 3



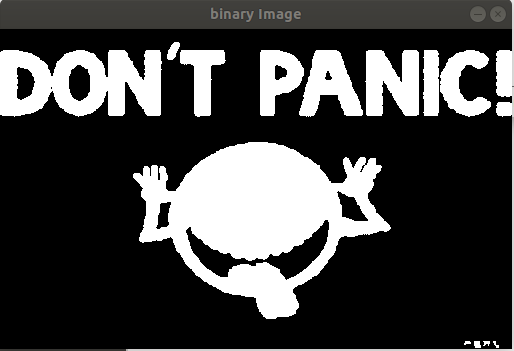
1. Region Detection after labelling



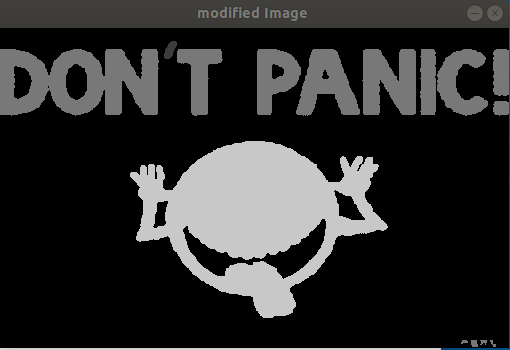
1. Threshold of the image for Figure 4 test image



1. Binary image Figure 4



1. Region detection after labelling



Bug Report:

No, bugs were detected to my knowledge