# Open Ended Experiment

# Basic Simulation Lab

# (ES 204)

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Submitted to:

Dr Shalini Shah

Ast. Professor

ECE Department, ASET

Submitted by:

Shaina Mehta

A2305219268

B.tech. C.S.E.

3CSE-4Y

AMITY SCHOOL OF ENGINEERING AND TECHNOLOGY

AMITY UNIVERSITY UTTAR PRADESH

NOIDA -201301

# Open Ended Experiment

**Aim:** (1) To brighten and darken an image and plot the original, bright and dark images.
(2) To obtain a negative of a grayscale image and plot the original image and its negative.

**Tools Used:** MATLAB/Octave Online I.D.E.

**Theory:** Image Processing Toolbox in MATLAB provides a comprehensive set of reference-standard algorithms and workflow apps for image processing, analysis, visualization, and algorithm development. You can perform image segmentation, image enhancement, noise reduction, geometric transformations, and image registration using deep learning and traditional image processing techniques. The toolbox supports processing of 2D, 3D, and arbitrarily large images.

Image Processing Toolbox apps let you automate common image processing workflows. You can interactively segment image data, compare image registration techniques, and batch-process large datasets. Visualization functions and apps let you explore images, 3D volumes, and videos; adjust contrast; create histograms; and manipulate regions of interest (ROIs).

You can accelerate your algorithms by running them on multicore processors and GPUs. Many toolbox functions support C/C++ code generation for desktop prototyping and embedded vision system deployment.

In this experiment we have used various image processing tools for performing various image processing operation i.e. to brighten and darken the image, to black and whiten the image and to make the grayscale image as negative etc.

The functions used in this experiment along with their description are:

(1) Imread(): It is used for reading the image from the graphic file.
Example: I=imread(‘India.png’);

(2) Imshow(): It is used for displaying the image.
Example: imshow(I); or imshow(‘India.png’);

(3) Imfinfo(): It is used for displaying the information of the graphic file.
Example: A=imfinfo(‘India.png);
display(A);

(4) Imhist(): it is used to create a histogram plot by defining n equally spaced bins,each representing a range of data values, and then calculating the number of pixels within each range. You can use the information in a histogram to choose an appropriate enhancement operation.
Example: imhist(I);

(5) Histeq(): It is used to transforms the grayscale image so that the histogram of the output grayscale image has 64 bins and is approximately flat.
Example: J = histeq(I);

(6) Rbg2gray(): It is used to convert the truecolor image to the grayscale image . The rgb2gray function converts RGB images to grayscale by eliminating the hue and saturation information while retaining the luminance.
Example: I = rgb2gray(RGB);

(7) Imadjust(): This function can be explained with the help of an example

(a) J = imadjust(I) maps the intensity values in grayscale image I to new values in J. By default, imadjust saturates the bottom 1% and the top 1% of all pixel values. This operation increases the contrast of the output image J.

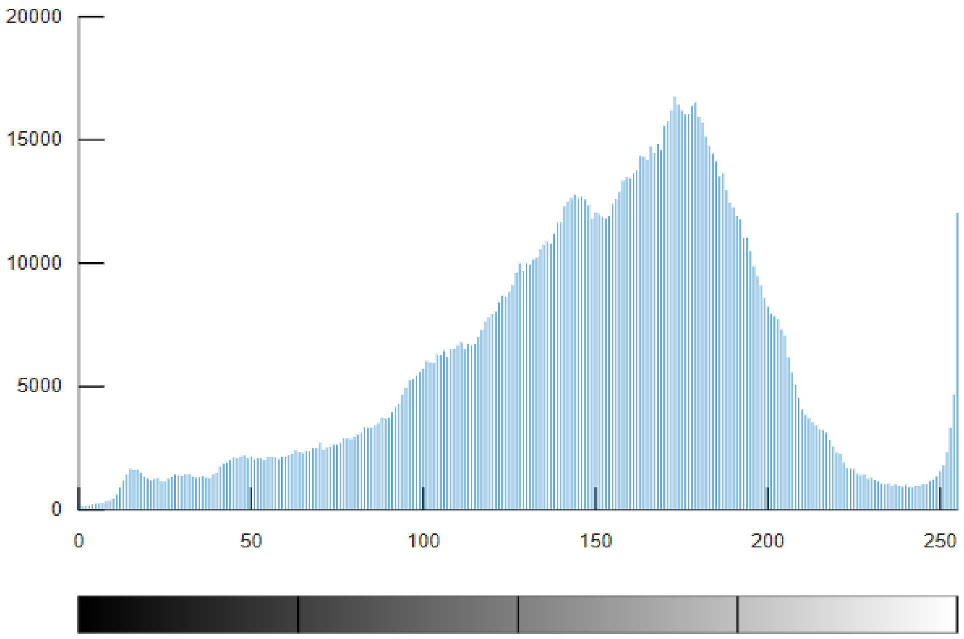
(b) J = imadjust(RGB,[low_in high_in]) maps the values in truecolor image RGB to new values in J. You can apply the same mapping or unique mappings for each color channel.
Example: J=imadjust(I,[0.3 0.4 0, 0.7 0.8 1]);

## Programs and Output:

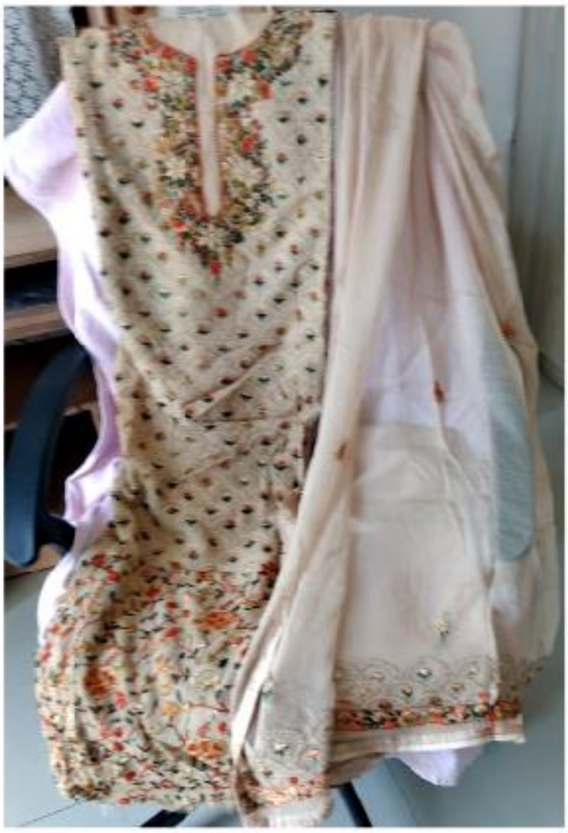
(1) Basic Image Operations:

**Input**>I=imread('suit11.jpg');
imshow(I);
**Output>**

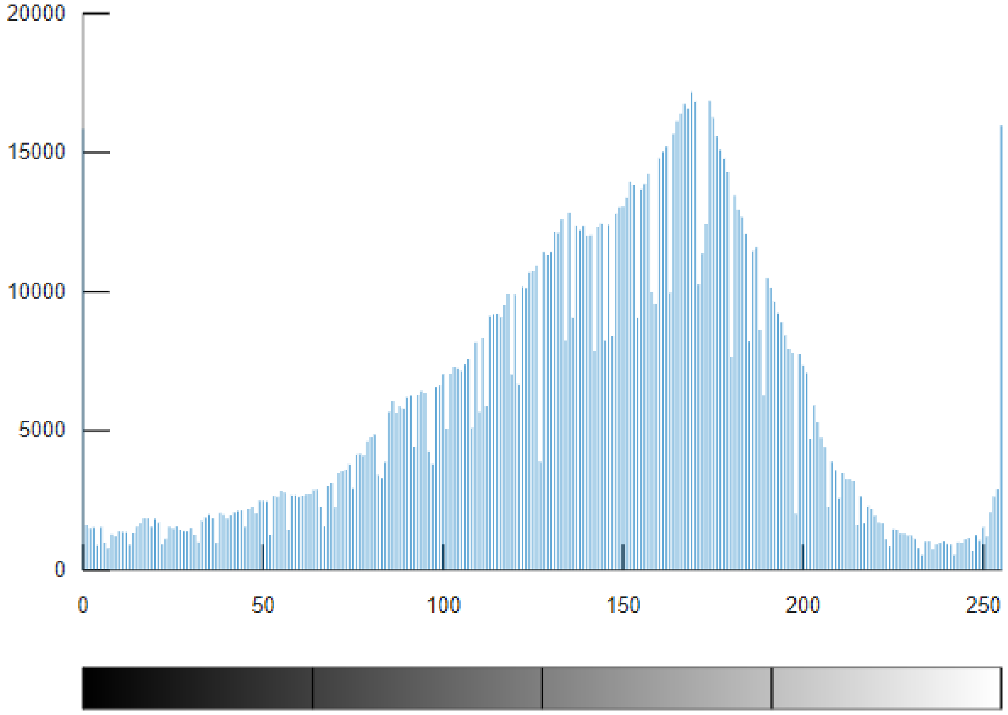
**Input**>imhist(I);
**Output>**



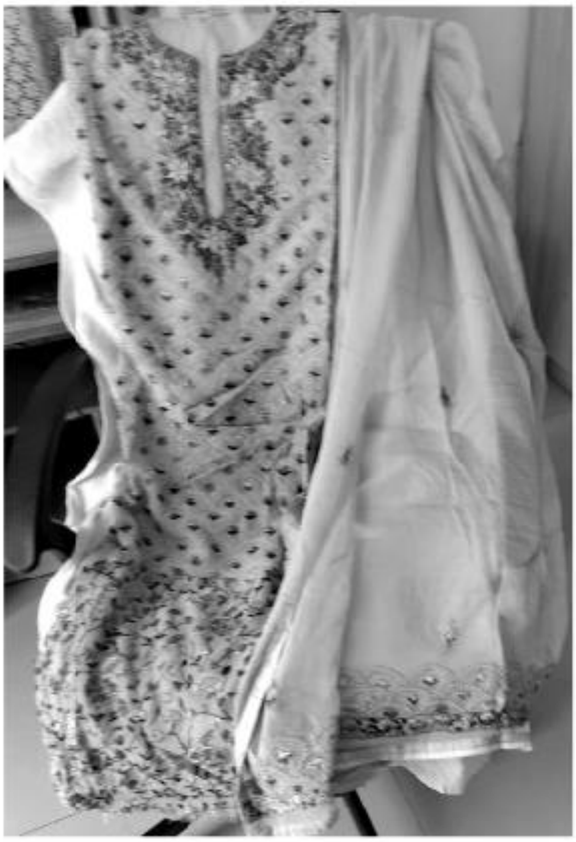
**Input**>imadjust(I);
imshow(I2);
**Output**>



**Input**>imhist(I2);
**Output**>

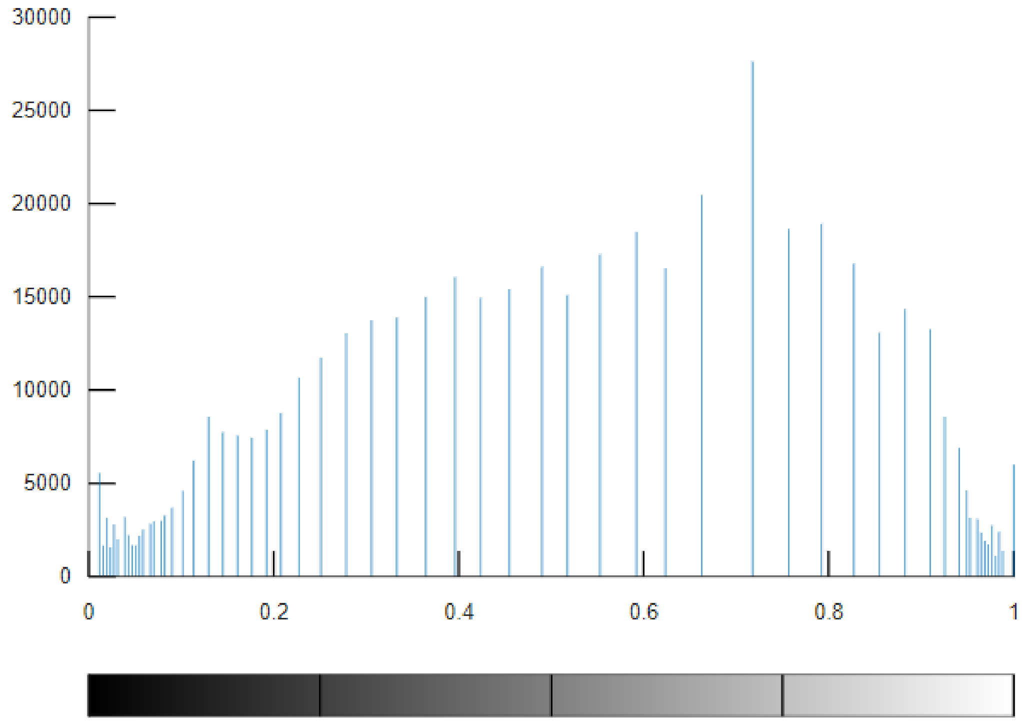


```
Input>I3=rgb2gray(I2);
imshow(I3);
Output>
```



**Input>**I4=histeq(I3);
imshow(I4);
**Output>**

**Input>**imhist(I4);
**Output>**



**Input**>A=imfinfo('suit11.jpg');
display(A);

**Output**>

```
A =

  scalar structure containing the fields:

    Filename = /home/oo/suit11.jpg
    FileModDate = 25-Sep-2020 12:30:17
    FileSize = 135915
    Format = JPEG
    FormatVersion =
    Width = 577
    Height = 869
    BitDepth = 8
    ColorType = truecolor
    DelayTime = 0
    DisposalMethod =
    LoopCount = 0
    ByteOrder = undefined
    Gamma = 0
    Chromaticities = [](1x0)
    Comment =
    Quality = 75
    Compression = undefined
```

```
Colormap = [](0x0)
Orientation = 1
ResolutionUnit = Inch
XResolution = 96
YResolution = 96
Software = Windows Photo Editor 10.0.10011.16384
Make =
Model =
DateTime = 2020:05:20 22:38:35
ImageDescription =
Artist =
Copyright =
DigitalCamera =

  scalar structure containing the fields:

    DateTimeOriginal I = 2020:05:20 22:16:56
    DateTimeDigitized = 2020:05:20 22:16:56
    SubSecTimeOriginal = 10
    SubSecTimeDigitized = 10
    ColorSpace = 1

GPSInfo =

  scalar structure containing the fields:
```

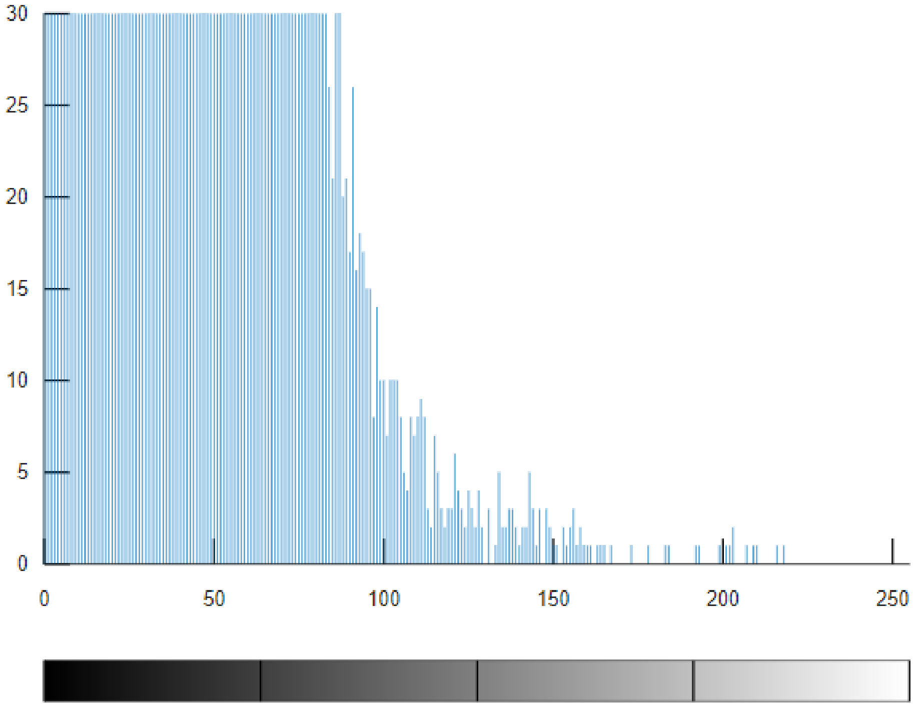
(2) Brightening of the Dark Image:

**Input**>I=imread('dark.png');
imshow(I);

**Output**>

**Input**>imhist(I);

**Output**>

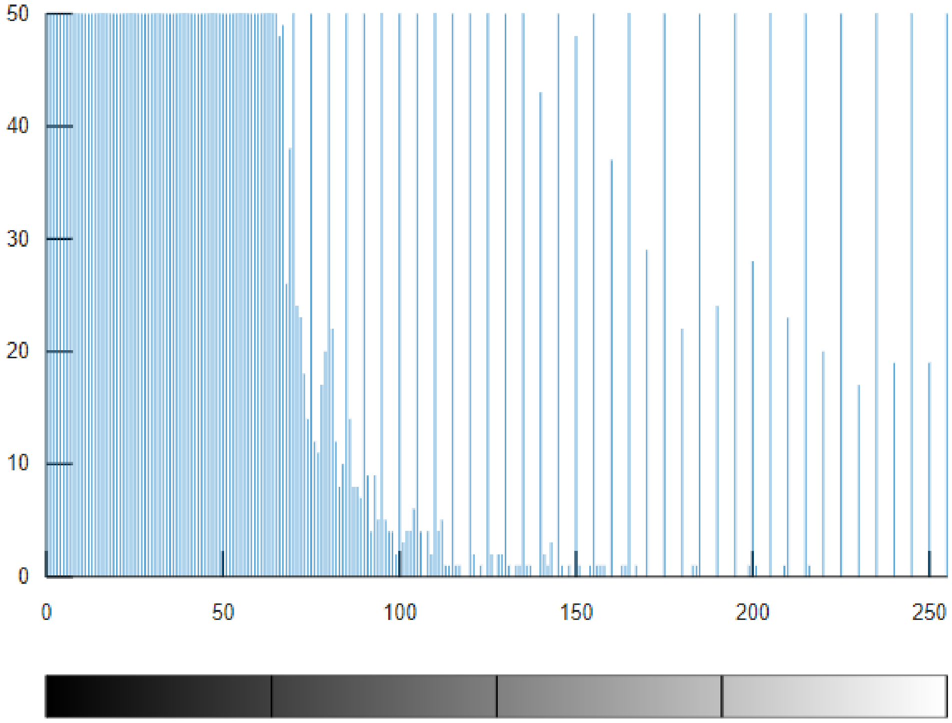


**Input**>I2=imadjust(I,[0.2 0.1 0;0.3 0.2 1]);
imshow(I2);

**Output**>

**Input**>imhist(I2);

**Output**>



(3) Darkening of the Bright Image:

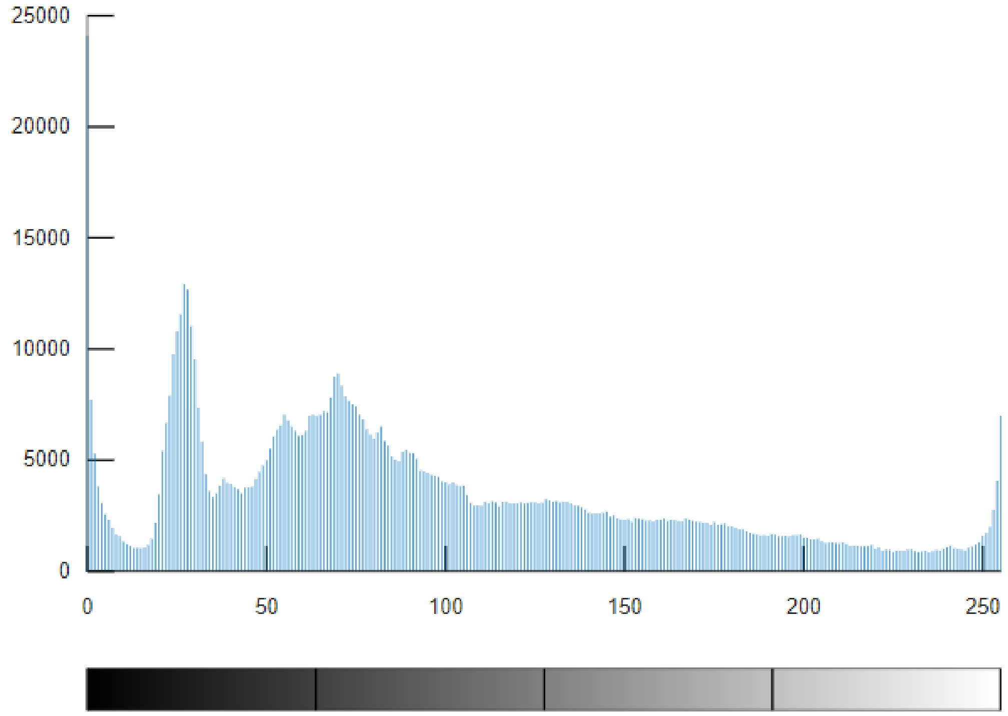
**Input**>I=imread('flower.png');

imshow(I);

**Output>**

**Input>**imhist(I);

**Output>**



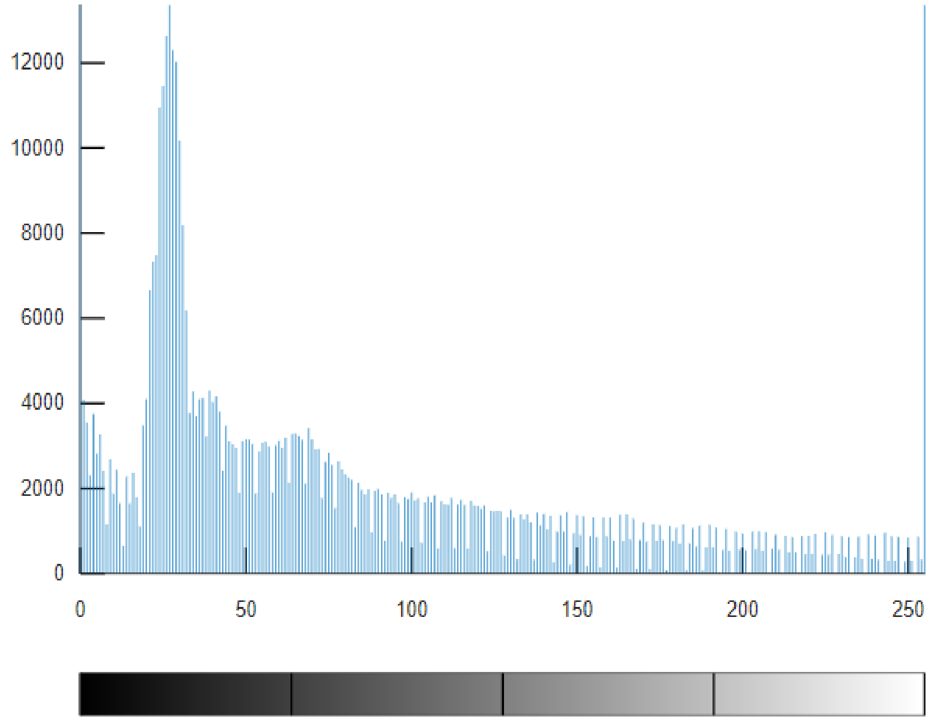
**Input>**I2=imadjust(I,[0.4 0.5 0;0.8 0.9 1]);

imshow(I2);

**Output>**

**Input**>imhist(I4);

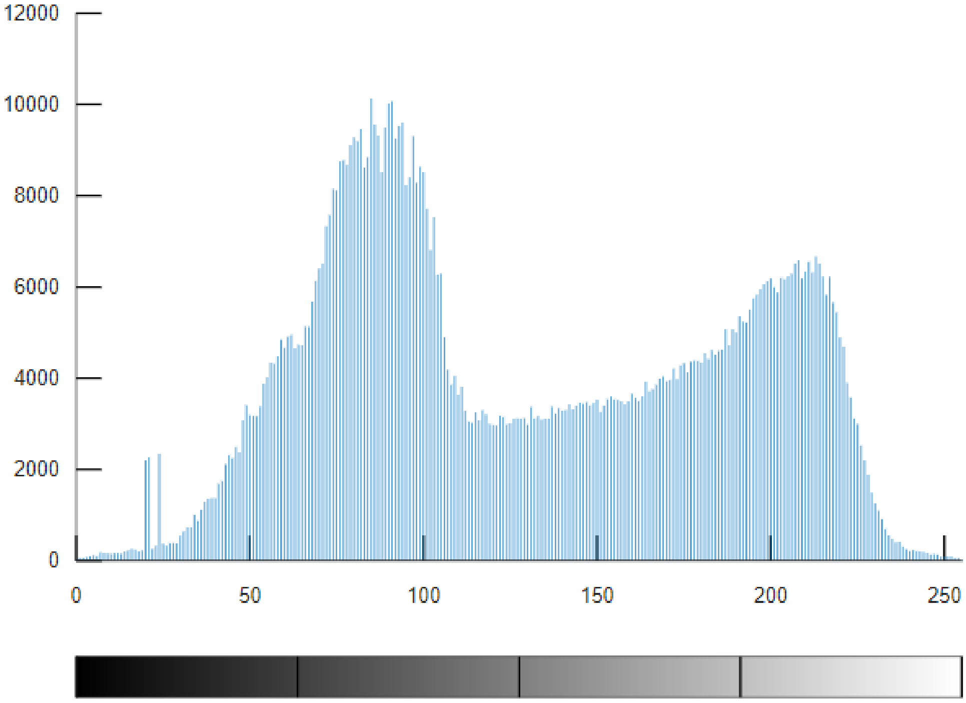
**Output**>



(4) Obtaining of Negative of the Grayscale Image:

**Input**>I=imread('cotton.png');
imshow(I);
**Output**>

**Input**>imhist(I);
**Output**>

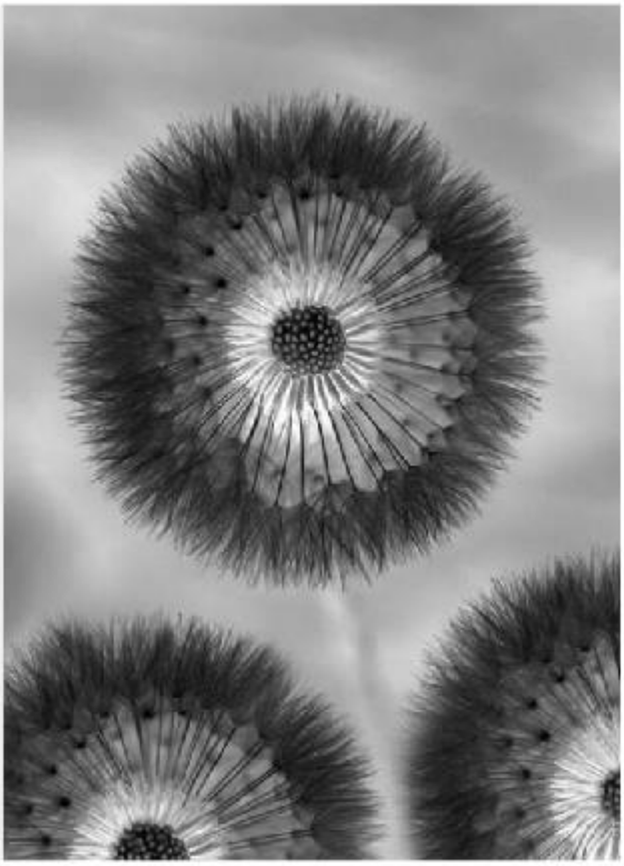


**Input**>A=imfinfo('cotton.png');
display(A);

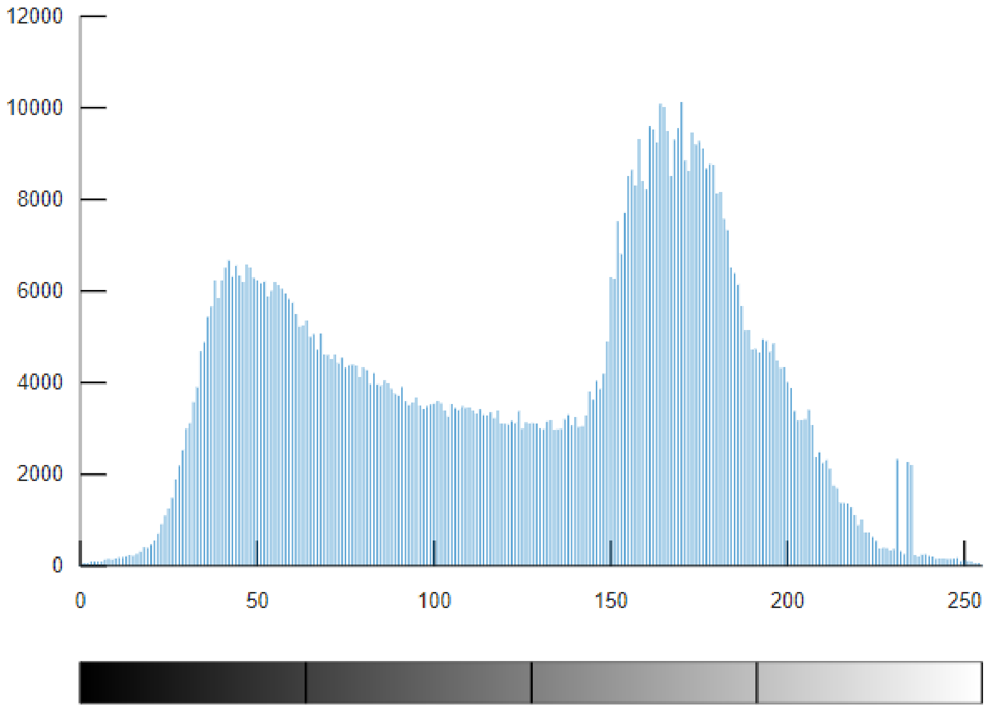
**Output>**

```
A =

  scalar structure containing the fields:

    Filename = /home/oo/cotton.png
    FileModDate = 25-Sep-2020 13:59:10
    FileSize = 273891
    Format = PNG
    FormatVersion =
    Width = 483
    Height = 667
    BitDepth = 8
    ColorType = truecolor
    DelayTime = 0
    DisposalMethod =
    LoopCount = 0
    ByteOrder = undefined
    Gamma = 0.4545
    Chromaticities =

     Columns 1 through 6:

       0.312700   0.329000   0.640000   0.330000   0.300000   0.600000

     Columns 7 and 8:

       0.150000   0.060000

    Comment =
    Quality = 75
    Compression = undefined
    Colormap = [](0x0)
    Orientation = 1
    ResolutionUnit = Centimeter
    XResolution = 47.240
    YResolution = 47.240
    Software =
    Make =
    Model =
    DateTime =
    ImageDescription =
    Artist =
    Copyright =
    DigitalCamera =

      scalar structure containing the fields:


    GPSInfo =

      scalar structure containing the fields:
```

```
Input>B=2^8;
b=B-1;
neg=b-I;
imshow(neg);
Output>
```



```
Input>imhist(neg);
Output>
```



**Results and Conclusion:** The brightening, darkening and obtaining the negative of an image has been done successfully.