**Assignment NO #03**

****

Spring 2023

CSE-423 Digital Image Processing

Submitted by: **Ashfaq Ahmad**

Registration No: **19PWCSE1795**

Class Section: **B**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Submitted to:

**Doctor. Abeer Irfan**

**June** 14, 2023

**Department of Computer Systems Engineering**

**University of Engineering and Technology, Peshawar**

**Task01:**

Prove that the images in Fig. 6.1(a) and Fig. 6.2(a) have the same histogram.

**MATLAB Code(a):**

% Specify the size of the image

imageSize = 255;

% Calculate the number of pixels for each portion (white and black)

numPixels = floor(imageSize^2 / 2);

% Create a binary image with equal portions of white and black

image = ones(imageSize);

% Set the pixels for the white portion

image(1:numPixels) = 0;

% Convert the image to grayscale

grayImage = uint8(image \* 255);

% Display the grayscale image

imshow(grayImage);

% Compute the histogram of gray-level values

histValues = imhist(grayImage);

% Plot the histogram

figure;

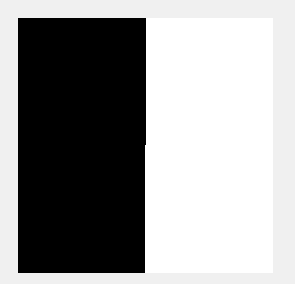
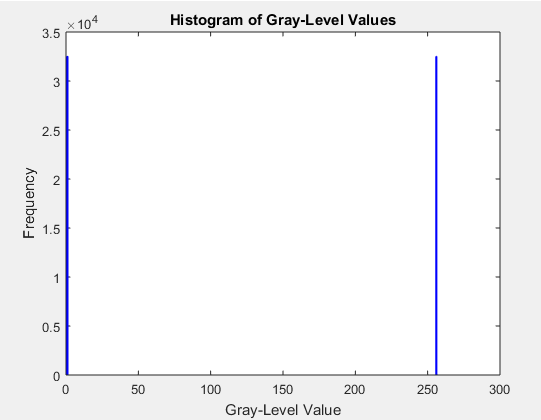
bar(histValues, 'FaceColor', 'b', 'EdgeColor', 'none');

title('Histogram of Gray-Level Values');

xlabel('Gray-Level Value');

ylabel('Frequency');

**Result:**

**MATLAB Code (b):**

% Specify the size of the image

imageSize = 255;

% Create a binary image with the desired pattern

image = zeros(imageSize);

% Set the pixels for the upper left corner (white)

image(1:imageSize/2, 1:imageSize/2) = 1;

% Set the pixels for the upper right corner (black)

image(1:imageSize/2, imageSize/2+1:end) = 0;

% Set the pixels for the lower left corner (black)

image(imageSize/2+1:end, 1:imageSize/2) = 0;

% Set the pixels for the lower right corner (white)

image(imageSize/2+1:end, imageSize/2+1:end) = 1;

% Convert the image to grayscale

grayImage = uint8(image \* 255);

% Display the grayscale image

imshow(grayImage);

% Compute the histogram of gray-level values

histValues = imhist(grayImage);

% Plot the histogram

figure;

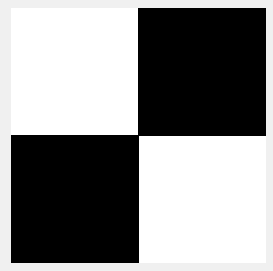
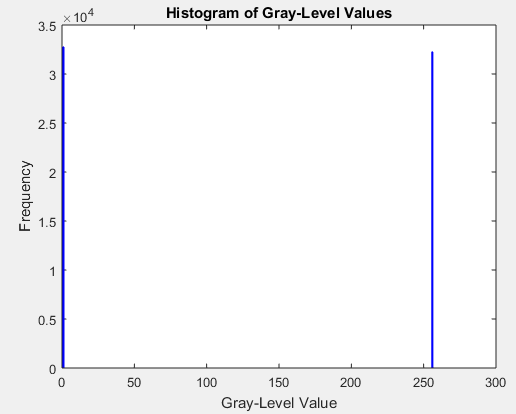
bar(histValues, 'FaceColor', 'b', 'EdgeColor', 'none');

title('Histogram of Gray-Level Values');

xlabel('Gray-Level Value');

ylabel('Frequency');

**Result:**

**Task02:**

Prove that dark, bright, low contrast, and high contrast images have histograms similar to the

histograms in Fig. 6.4 (a), Fig. 6.4 (b), Fig. 6.4 (c), and Fig. 6.4(d),

respectively.

**MATLAB Code:**

% Load the image

grayImage = imread('lena.png');

% Create the modified versions of the image

darkImage = imadjust(grayImage, [], [], 2);

brightImage = imadjust(grayImage, [], [], 0.088);

lowContrastImage = imadjust(grayImage, [], [], 1);

highContrastImage = imadjust(grayImage, [], [], 2.5);

% Calculate the histograms of the modified images

darkHistogram = imhist(darkImage);

brightHistogram = imhist(brightImage);

lowContrastHistogram = imhist(lowContrastImage);

highContrastHistogram = imhist(highContrastImage);

% Plot the histograms

figure;

subplot(2, 2, 1);

bar(darkHistogram);

title('Dark Image Histogram');

xlabel('Intensity');

ylabel('Frequency');

subplot(2, 2, 2);

bar(brightHistogram);

title('Bright Image Histogram');

xlabel('Intensity');

ylabel('Frequency');

subplot(2, 2, 3);

bar(lowContrastHistogram);

title('Low Contrast Image Histogram');

xlabel('Intensity');

ylabel('Frequency');

subplot(2, 2, 4);

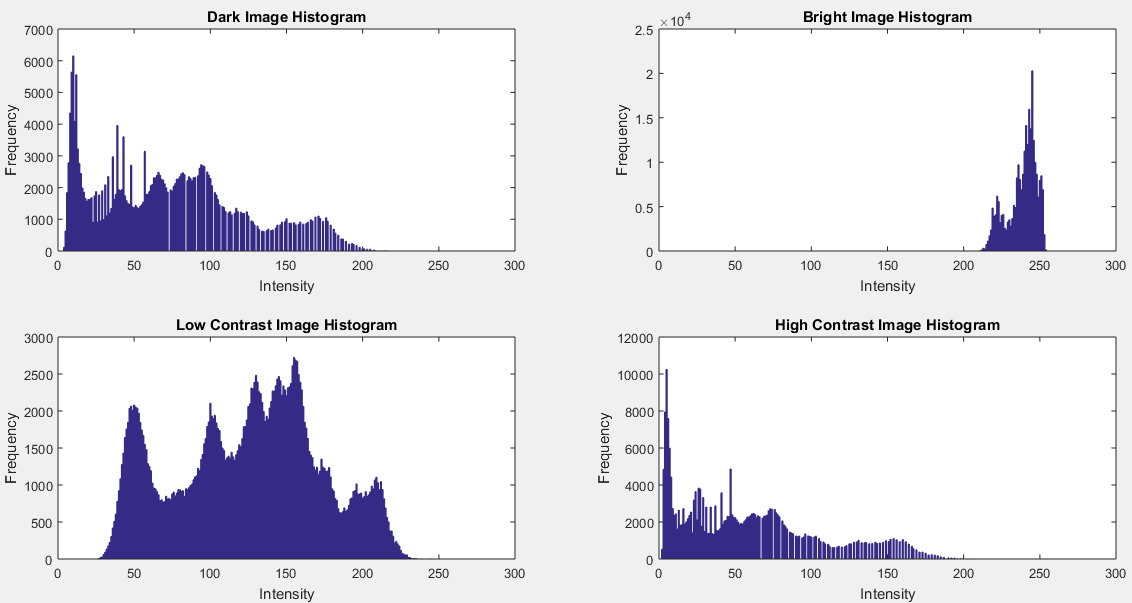
bar(highContrastHistogram);

title('High Contrast Image Histogram');

xlabel('Intensity');

ylabel('Frequency');

Result:



**Task03:**

Apply histogram specification on a low-contrast image. Sketch and compare the histogram of the input

and output images.

**MATLAB Code:**

% Load the low-contrast image

lowContrastImage = imread('house.png');

% Compute the histogram of the input image

inputHistogram = imhist(lowContrastImage);

% Define the desired histogram (specification)

desiredHistogram = linspace(0, 255, 256);

% Perform histogram specification

outputImage = histeq(lowContrastImage, desiredHistogram);

% Compute the histogram of the output image

outputHistogram = imhist(outputImage);

% Plot the histograms

figure;

subplot(2, 1, 1);

bar(inputHistogram);

title('Input Image Histogram');

xlabel('Intensity');

ylabel('Frequency');

subplot(2, 1, 2);

bar(outputHistogram);

title('Output Image Histogram');

xlabel('Intensity');

ylabel('Frequency');

**Result:**

