**University of the Punjab**

**Gujranwala Campus**

**Department of Information Technology**



**Assignment: Computer Vision**

**Prepared by:**

Hafiza Amber Ijaz

**Roll no:**

BIT21001

**Submitted to:**

Miss Fouqia Zafeer

**EXERCISE 1:**

**Write a program which can read an image as an input and do the following automatically. Show the results of all steps.1.**Find the type of image: binary, gray or RGB.

% Load or use the input image stored in variable A

A = imread('pic1.png');

% Display the input image

figure('Name', 'Original Image');

imshow(A);

title('Original Image');

% Step 1: Identify image type

if islogical(A)

img\_type = 'Binary';

elseif ndims(A) == 3

img\_type = 'RGB';

else

img\_type = 'Grayscale';

end

% Display the image type

disp(['Image type: ', img\_type]);

% Convert RGB to grayscale for further processing

if strcmp(img\_type, 'RGB')

img\_gray = rgb2gray(A);

elseif strcmp(img\_type, 'Grayscale')

img\_gray = A;

else

img\_gray = double(A) \* 255; % Convert binary to grayscale

End

2.Find the issue in image, over dark, over bright, low contrast, or normal. (Hint: can use histogram).

% Step 2: Analyze histogram

figure('Name', 'Histogram');

histogram(img\_gray, 256);

title('Image Histogram');

% Determine brightness and contrast issues

mean\_intensity = mean(img\_gray(:));

contrast = max(img\_gray(:)) - min(img\_gray(:));

if mean\_intensity < 50

issue = 'Over Dark';

elseif mean\_intensity > 200

issue = 'Over Bright';

elseif contrast < 50

issue = 'Low Contrast';

else

issue = 'Normal';

end

% Display the identified issue

disp(['Image issue: ', issue]);

3.Resolve the issue if any and show the final image after enhancement.

% Step 3: Resolve issues

enhanced\_img = img\_gray; % Start with the grayscale image

switch issue

case 'Over Dark'

enhanced\_img = imadjust(img\_gray, stretchlim(img\_gray), []);

case 'Over Bright'

enhanced\_img = imadjust(img\_gray, [0.2, 1], []);

case 'Low Contrast'

enhanced\_img = histeq(img\_gray);

end

% Display results

figure('Name', 'Enhanced Image');

imshow(enhanced\_img);

title('Enhanced Image');

4.Test your program on following images

% Step 4: Compare original and enhanced images

if ~strcmp(issue, 'Normal')

disp(['Enhancement applied: ', issue]);

figure('Name', 'Comparison');

subplot(1, 2, 1);

imshow(img\_gray);

title('Before Enhancement');

subplot(1, 2, 2);

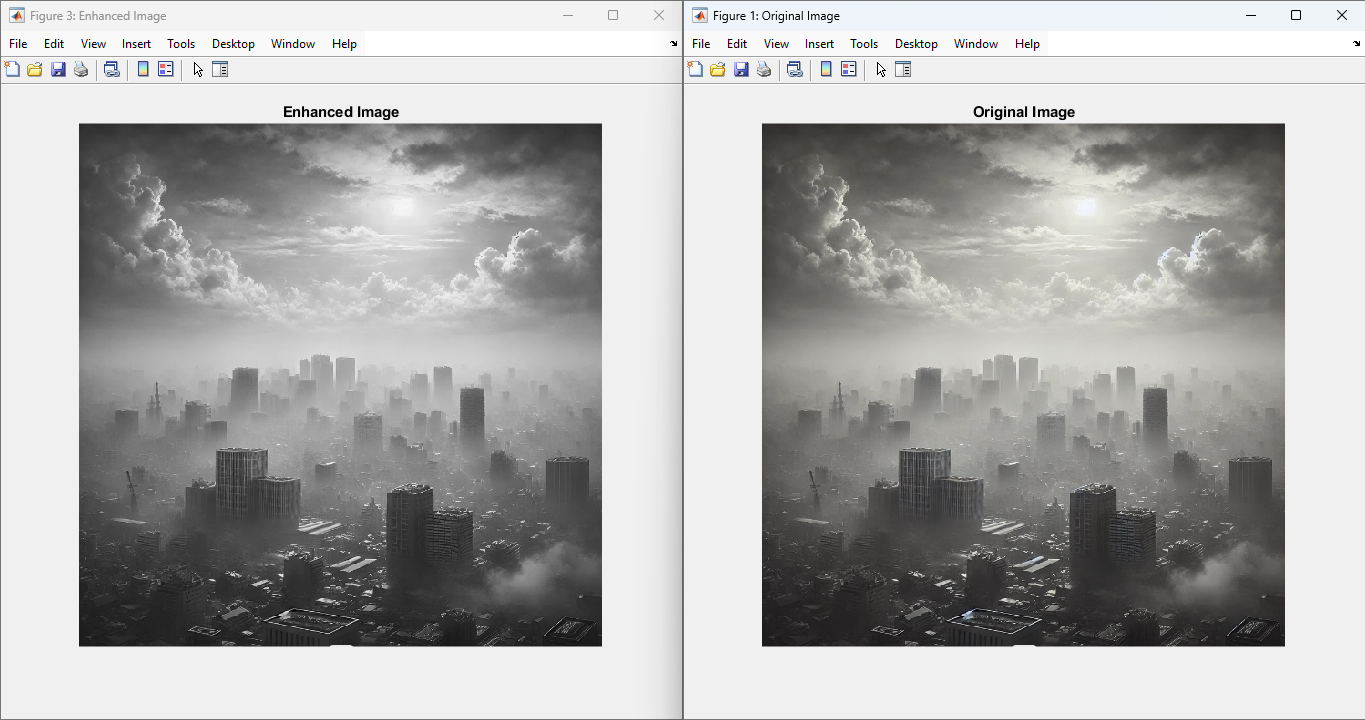
imshow(enhanced\_img);

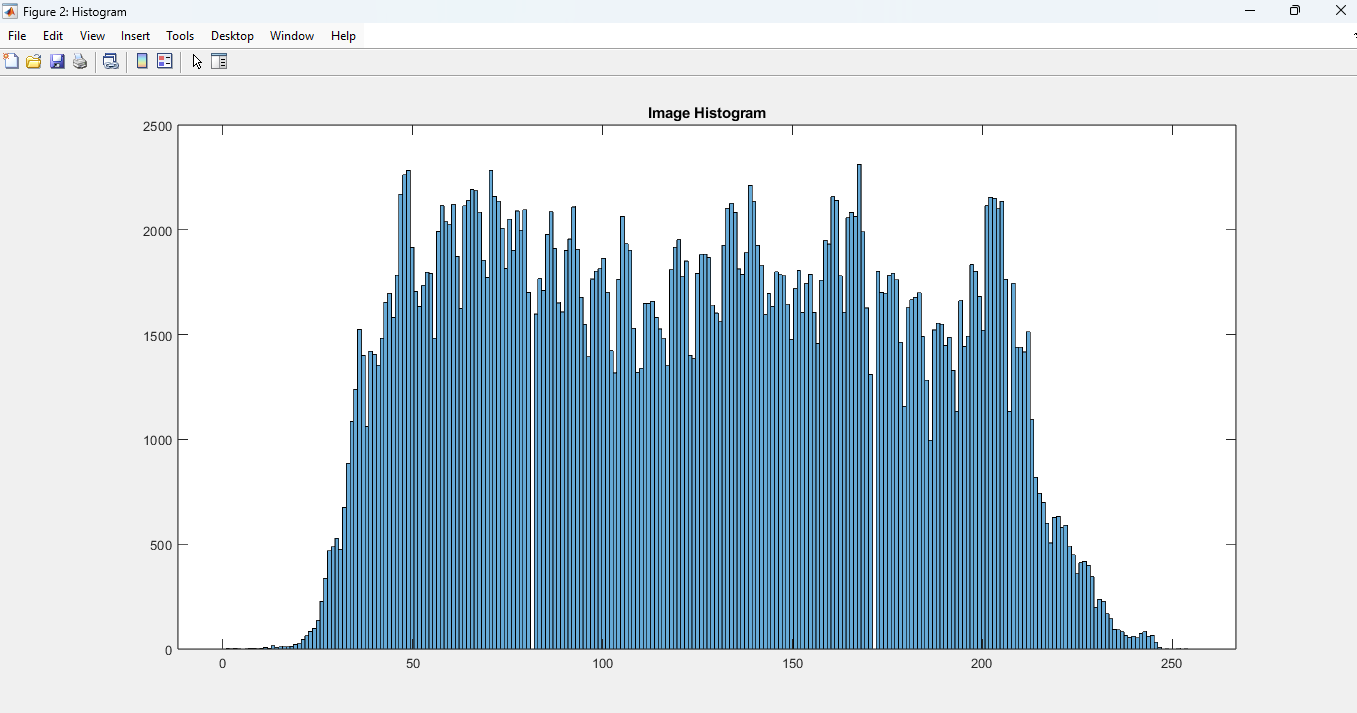
title('After Enhancement');

else

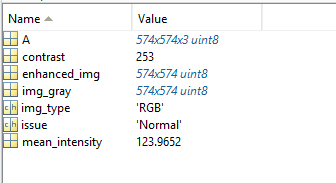
disp('No enhancement needed.');

end









**EXERCISE 2:**

**Identify which intensity transformation was used on liftingbody.png to create each of the four results below. Write a script to reproduce the results using the intensity transformation functions.**

**CODE:**

function intensityTransformations()

% Read the original image

originalImage = imread('tree.jpeg');

% Ensure the image is grayscale

if ndims(originalImage) == 3

originalImage = rgb2gray(originalImage);

end

% Apply transformations

% 1. Darkened image (Result 1)

result1 = imadjust(originalImage, [], [], 0.5); % Gamma correction with gamma < 1

% 2. Brightened image (Result 2)

result2 = imadjust(originalImage, [], [], 1.5); % Gamma correction with gamma > 1

% 3. High contrast image (Result 3)

result3 = histeq(originalImage); % Histogram equalization

% 4. Low contrast image (Result 4)

result4 = imadjust(originalImage, [0.3 0.7], [0.4 0.6]); % Adjust intensity range

% Display results

figure;

subplot(2, 3, 1);

imshow(originalImage);

title('Original Image');

subplot(2, 3, 2);

imshow(result1);

title('Result 1: Darkened');

subplot(2, 3, 3);

imshow(result2);

title('Result 2: Brightened');

subplot(2, 3, 4);

imshow(result3);

title('Result 3: High Contrast');

subplot(2, 3, 5);

imshow(result4);

title('Result 4: Low Contrast');

End

