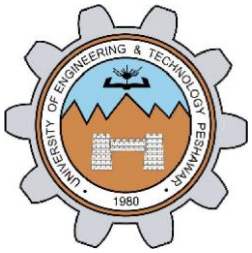
Assignment No 1



Spring 2025

CSE-408 Digital Image Processing

Submitted by: MUHAMMAD SADEEQ

Registration No.: 21PWCSE2028

Section: C

"On my honor, as a student of the University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work"

Submitted to:

Engr. Mehran Ahmad

(20 May 2025)

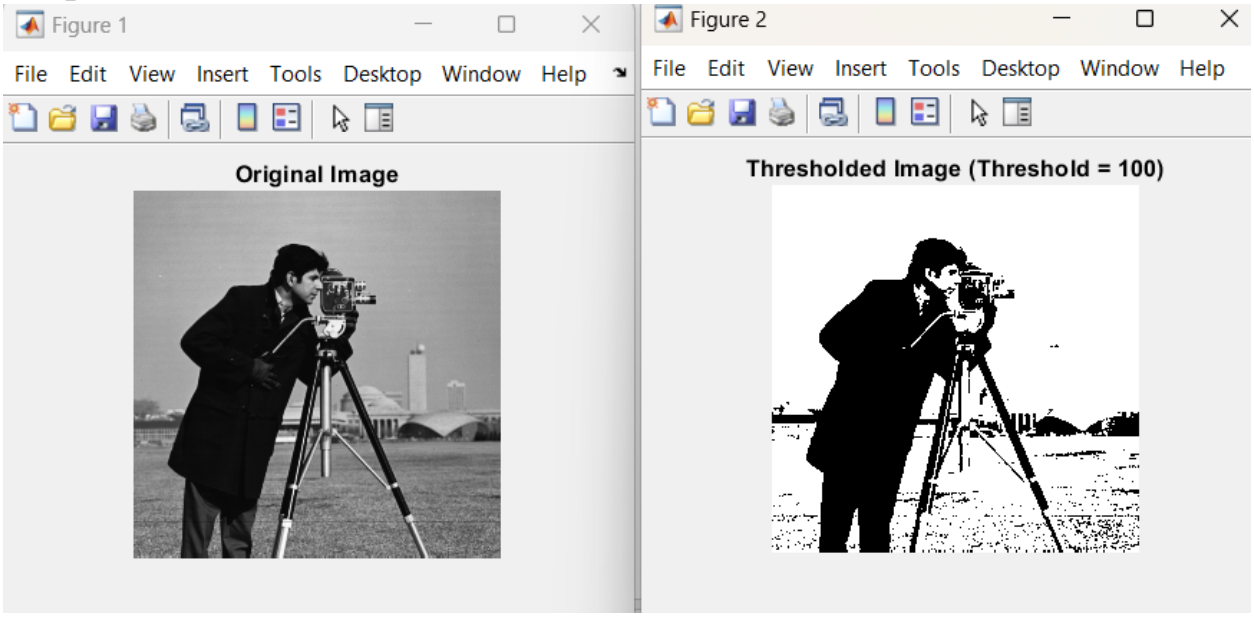
Department of Computer systems engineering
University of Engineering and Technology,
Peshawar

Activity 1

Code:

```
% Read the original image
originalImage = imread('cameraman.tif');
figure;
imshow(originalImage);
title('Original Image');

% Convert to grayscale if the image is RGB
if size(originalImage, 3) == 3
    grayImage = rgb2gray(originalImage);
else
    grayImage = originalImage;
end

% Define a threshold value (0 to 255 for uint8 images)
thresholdValue = 100;

% Apply thresholding
binaryImage = grayImage > thresholdValue;

% Display the thresholded image
figure;
imshow(binaryImage);
title(['Thresholded Image (Threshold = ', num2str(thresholdValue), ')']);
```

Output:



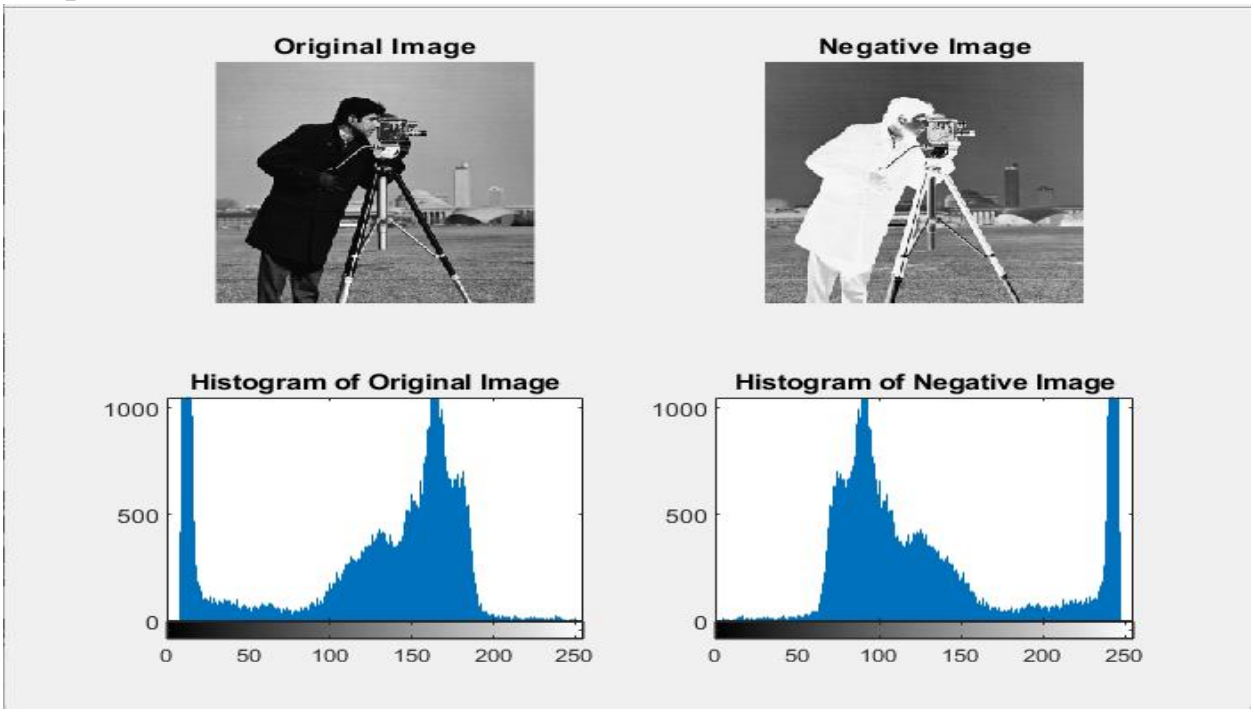
Activity 2

Code:

```
% Read the original image
```

```
originalImage = imread('cameraman.tif');

% Convert to grayscale if it's RGB
if size(originalImage, 3) == 3
    grayImage = rgb2gray(originalImage);
else
    grayImage = originalImage;
end

% Negative transformation
negativeImage = 255 - grayImage;

% Display original and negative images
figure;
subplot(2, 2, 1);
imshow(grayImage);
title('Original Image');

subplot(2, 2, 2);
imshow(negativeImage);
title('Negative Image');

% Plot histograms
subplot(2, 2, 3);
imhist(grayImage);
title('Histogram of Original Image');

subplot(2, 2, 4);
imhist(negativeImage);
title('Histogram of Negative Image');
```

Output:



Activity 3

Code:

```
% Read the original image
originalImage = imread('cameraman.tif');

% Convert to grayscale if it's RGB
if size(originalImage, 3) == 3
    grayImage = rgb2gray(originalImage);
else
    grayImage = originalImage;
end

% Convert image to double for log transformation
doubleImage = im2double(grayImage);

% Apply log transformation
c = 1; % Scaling constant (can adjust to enhance effect)
logImage = c * log(1 + doubleImage);

% Normalize to 0-1 for display
logImage = mat2gray(logImage);

% Display original and log-transformed image
figure;

subplot(1, 2, 1);
imshow(grayImage);
title('Original Image');

subplot(1, 2, 2);
imshow(logImage);
title('Log-Transformed Image');
```

Output:

Activity 4

Code:

```
% Read the original image
originalImage = imread('cameraman.tif');

% Convert to grayscale if it's RGB
if size(originalImage, 3) == 3
    grayImage = rgb2gray(originalImage);
else
    grayImage = originalImage;
end

% Convert to double and normalize (0 to 1)
doubleImage = im2double(grayImage);

% Define gamma values to test
gammaValues = [0.4, 0.7, 1.0, 1.5, 2.5];

% Create figure for original and transformed images
figure;

% Show original image
subplot(2, ceil((length(gammaValues) + 1)/2), 1);
imshow(grayImage);
title('Original Image');

% Apply power-law transformation for each gamma
for i = 1:length(gammaValues)
    gamma = gammaValues(i);
    transformedImage = doubleImage .^ gamma;

    % Normalize and convert to image format
    transformedImage = mat2gray(transformedImage);

    % Display the transformed image
    subplot(2, ceil((length(gammaValues) + 1)/2), i + 1);
    imshow(transformedImage);
    title(['\gamma = ', num2str(gamma)]);
end
```

Output: