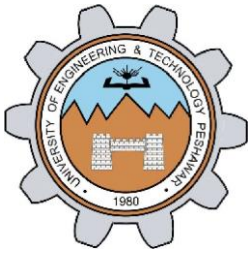
Assignment No 2



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');

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

% Convert to double and normalize
I = im2double(grayImage);

% Define control points for piecewise linear transformation
% These points define how intensity values are mapped
r1 = 0.2; s1 = 0.1;
r2 = 0.7; s2 = 0.9;

% Apply piecewise linear transformation
J = zeros(size(I));
for i = 1:size(I,1)
    for j = 1:size(I,2)
        r = I(i,j);
        if r < r1
            J(i,j) = (s1/r1) * r;
        elseif r <= r2
            J(i,j) = ((s2 - s1)/(r2 - r1)) * (r - r1) + s1;
        else
            J(i,j) = ((1 - s2)/(1 - r2)) * (r - r2) + s2;
        end
    end
end

% Display results
figure;

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

subplot(1, 2, 2);
imshow(J);
title('Contrast Stretched Image');
```

Output:

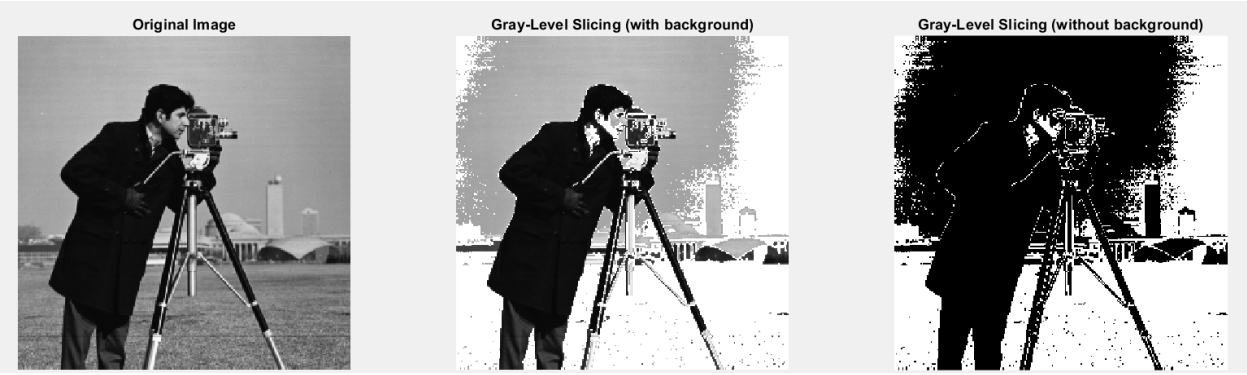
Activity 2

Code:

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

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

% Convert to double for calculations
grayImageDouble = double(grayImage);

% Define slicing range (intensity values between 0–255)
low = 80;
high = 160;

% Method 1: Highlight range, keep background
slice1 = grayImage;  % Copy of original
slice1(grayImage >= low & grayImage <= high) = 255;

% Method 2: Highlight range, suppress background
slice2 = zeros(size(grayImage));
slice2(grayImage >= low & grayImage <= high) = 255;

% Display results
figure;

subplot(1, 3, 1);
imshow(grayImage);
title('Original Image');

subplot(1, 3, 2);
imshow(slice1);
```

```
title('Gray-Level Slicing (with background)');

subplot(1, 3, 3);
imshow(slice2);
title('Gray-Level Slicing (without background)');
```

Output:

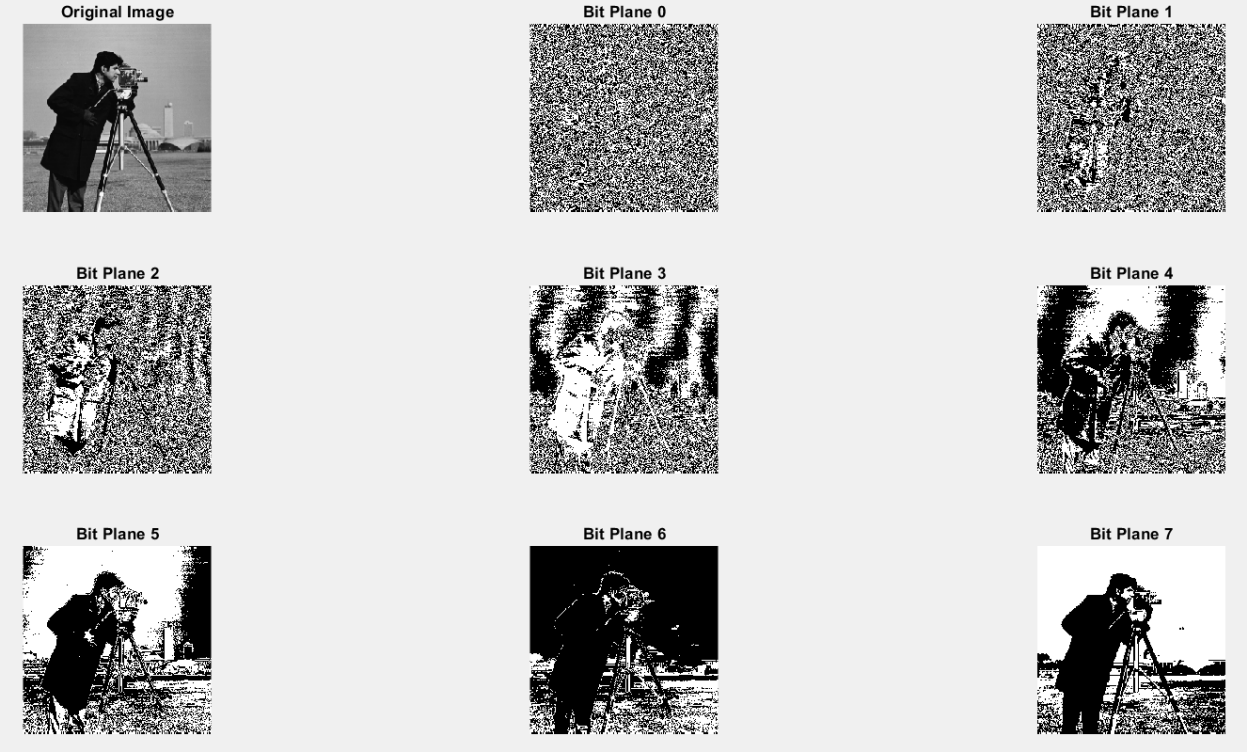


Activity 3

Code:

```
% Read the grayscale image
originalImage = imread('cameraman.tif');
% Convert to grayscale if RGB
if size(originalImage, 3) == 3
    grayImage = rgb2gray(originalImage);
else
    grayImage = originalImage;
end

% Display original image
figure;
subplot(3, 3, 1);
imshow(grayImage);
title('Original Image');

% Extract and display all 8 bit planes
for bit = 1:8
    % Extract the bit plane
    bitPlane = bitget(grayImage, bit);  % bit = 1 is LSB, bit = 8 is MSB

    % Display the bit plane
    subplot(3, 3, bit + 1);
    imshow(logical(bitPlane));
    title(['Bit Plane ', num2str(bit - 1)]);
end
```

Output:



Activity 4
(Part A)

Code:

```
% Read an image (grayscale or convert it)
originalImage = imread('cameraman.tif');
% Convert to grayscale if RGB
if size(originalImage, 3) == 3
    grayImage = rgb2gray(originalImage);
else
    grayImage = originalImage;
end

% Display original image
figure;
subplot(2, 3, 1);
imshow(grayImage);
title('Original Image');

% ---- 1. Moving Average Filter (3x3) ----
averageKernel = fspecial('average', [3 3]);
```

```
avgFiltered = imfilter(grayImage, averageKernel, 'replicate');
subplot(2, 3, 2);
imshow(avgFiltered);
title('3×3 Moving Average Filter');

% ---- 2. Median Filter (3x3) ----
medianFiltered = medfilt2(grayImage, [3 3]);
subplot(2, 3, 3);
imshow(medianFiltered);
title('Median Filter (3×3)');

% ---- 3. Min Filter (3x3) ----
minFiltered = ordfilt2(grayImage, 1, true(3));
subplot(2, 3, 4);
imshow(minFiltered);
title('Min Filter (3×3)');

% ---- 4. Max Filter (3x3) ----
maxFiltered = ordfilt2(grayImage, 9, true(3));
subplot(2, 3, 5);
imshow(maxFiltered);
title('Max Filter (3×3)');

% Optional: Difference image (background enhancement)
backgroundEnhanced = imsubtract(maxFiltered, minFiltered);
subplot(2, 3, 6);
imshow(backgroundEnhanced, []);
title('Max - Min (Background Features)');
```

Output:

(Part B)

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

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

% Apply Laplacian filter
laplacianKernel = fspecial('laplacian', 0.2);  % Default alpha = 0.2
laplacianFiltered = imfilter(double(grayImage), laplacianKernel, 'replicate');

% Enhance image by adding Laplacian (sharpening)
sharpenedImage = double(grayImage) - laplacianFiltered;

% Normalize to display
sharpenedImage = uint8(mat2gray(sharpenedImage) * 255);
laplacianFiltered = mat2gray(laplacianFiltered);

% Display results
figure;

subplot(1, 3, 1);
imshow(grayImage);
title('Original Image');

subplot(1, 3, 2);
imshow(laplacianFiltered, []);
title('Laplacian Filtered (Edges)');

subplot(1, 3, 3);
imshow(sharpenedImage);
title('Sharpened Image');
```

Output: