Lab Report 01

Submitted by:

Name: Shaikat Hazra Pranto

ID: 2020-1-60-072

Course: CSE438

Section: 2

Submitted To:

Md Ashraful Haider Chowdhury

Lecturer

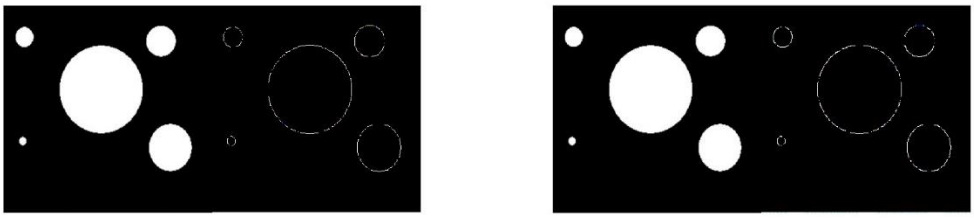
Department of Computer Science and Engineering

Submission date:

4th March, 2025

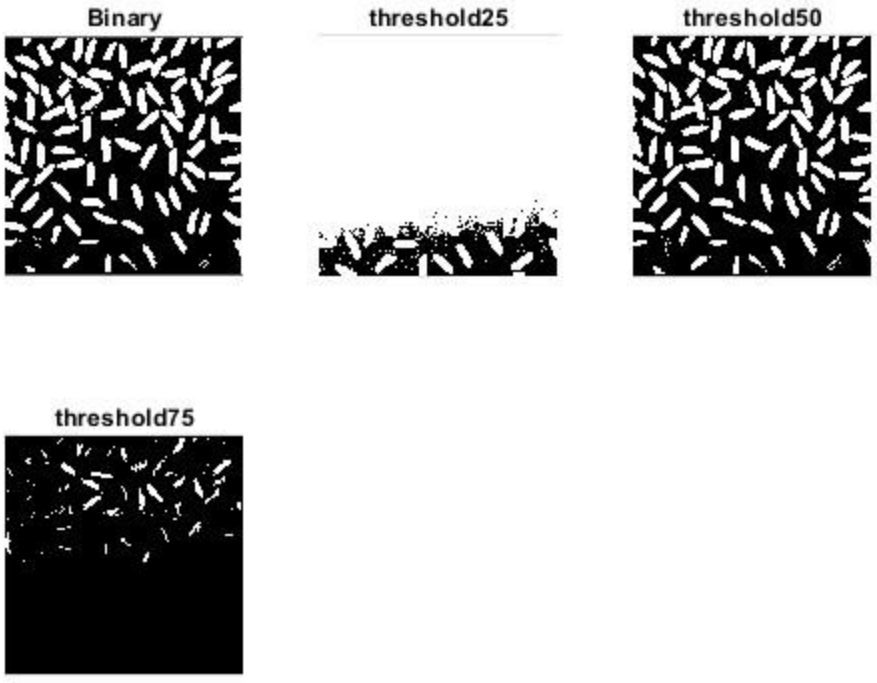
Problem 01:

```
clear all;
close all;

% Read the image
I = imread('Picture1.png');

n4 = bwperim(I, 4);
n8 = bwperim(I, 8);

figure();
subplot(1,2,1);
imshowpair(I,n4,'montage');
subplot(1,2,2);
imshowpair(I,n8,'montage');
%2020-1-60-072
```



Problem 02:

```
clear all;
close all;

% Read the image
I2 = imread('Picture2.png');

binaey_image = im2bw(I2);
threshold_025 = im2bw(I2 ,0.25);
threshold_50 = im2bw(I2 ,0.50);
threshold_75 = im2bw(I2 ,0.75);

% Display images
figure();
subplot(2,3,1), imshow(binaey_image ), title('Binary');
subplot(2,3,2), imshow(threshold_025), title('threshold25');
subplot(2,3,3), imshow(threshold_50), title('threshold50');
subplot(2,3,4), imshow(threshold_75), title('threshold75');
%2020-1-60-072
```



Problem 03:

```
clear all;
close all;

% Read the image
I2 = imread('Picture2.png');

binaey_image = im2bw(I2);
threshold_025 = im2bw(I2 ,0.25);
threshold_50 = im2bw(I2 ,0.50);
threshold_75 = im2bw(I2 ,0.75);
```

```
% Find the connected components using bwconncomp
cc_binary = bwconncomp(binaey_image);
cc_threshold_025 = bwconncomp(threshold_025);
cc_threshold_50 = bwconncomp(threshold_50);
cc_threshold_75 = bwconncomp(threshold_75);

% Number of objects (connected components)
numObjects_binary = cc_binary.NumObjects;
numObjects_threshold_025 = cc_threshold_025.NumObjects;
numObjects_threshold_50 = cc_threshold_50.NumObjects;
numObjects_threshold_75 = cc_threshold_75.NumObjects;

% Display the results
disp(['Number of objects in the binary image: ', num2str(numObjects_binary)]);
disp(['Number of objects with threshold 0.25: ', num2str(numObjects_threshold_025)]);
disp(['Number of objects with threshold 0.50: ', num2str(numObjects_threshold_50)]);
disp(['Number of objects with threshold 0.75: ', num2str(numObjects_threshold_75)]);
%2020-1-60-072
```

output:

Number of objects in the binary image: 256

Number of objects with threshold 0.25: 475

Number of objects with threshold 0.50: 256

Number of objects with threshold 0.75: 283

Problem 04:

```
clear all;
close all;

% Read the image
I2 = imread('Picture2.png');

% Convert to grayscale if needed
if size(I2, 3) == 3
   I2 = rgb2gray(I2);
end

% Display the image and select two points using the mouse
imshow(I2);
title('Click two points in the image');

% Get the coordinates of the two points selected by the user
% Use ginput to select two points from the image
[x, y] = ginput(2);  % Select 2 points

% Coordinates of the two points
x1 = x(1);
y1 = y(1);
x2 = x(2);
y2 = y(2);

% Calculate the Euclidean distance between the two points
```

```
distance = sqrt((x2 - x1)^2 + (y2 - y1)^2);

% Display the Euclidean distance
disp(['Euclidean Distance between the two points: ', num2str(distance)]);
%2020-1-60-072
```

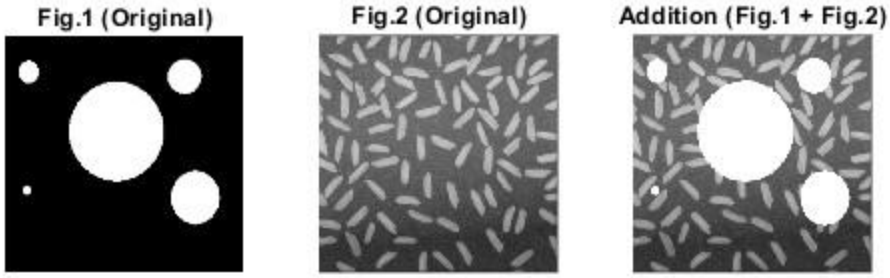
output:

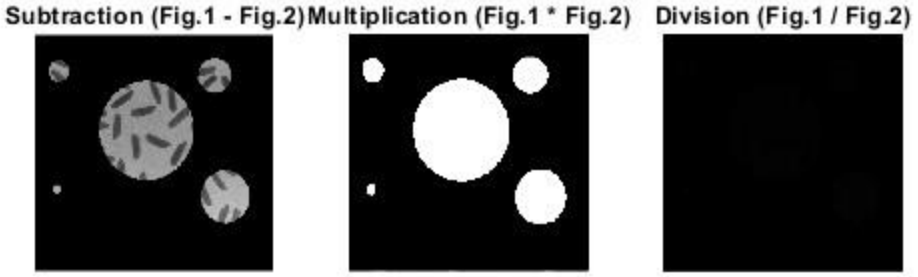
Euclidean Distance between the two points: 67.2086

Problem 05:

```
clear all
close all;

% Read the two images
I1 = imread('Picture1.png');
I2 = imread('Picture2.png');

% Convert to grayscale if needed
if size(I1, 3) == 3
   I1 = rgb2gray(I1);
end

if size(I2, 3) == 3
   I2 = rgb2gray(I2);
end
% Resize I2 to match I1 dimensions
I2 = imresize(I2, size(I1));

% a. Addition (Element-wise addition)
```

```
I_add = I1 + I2;

% b. Subtraction (Element-wise subtraction)
I_sub = I1 - I2;

% c. Multiplication (Element-wise multiplication)
I_mul = I1 .* I2;

% d. Division (Element-wise division)
% Note: We need to handle division carefully to avoid division by zero.
% Add a small epsilon value to avoid division by zero.
epsilon = 1e-10;  % Small value to prevent division by zero
I_div = I1 ./ (I2 + epsilon);

% Display images
figure();
subplot(2,3,1), imshow(I1), title('Fig.1 (Original)');
subplot(2,3,2), imshow(I2), title('Fig.2 (Original)');
subplot(2,3,3), imshow(I_add), title('Addition (Fig.1 + Fig.2)');
subplot(2,3,4), imshow(I_sub), title('Subtraction (Fig.1 - Fig.2)');
subplot(2,3,5), imshow(I_mul), title('Multiplication (Fig.1 * Fig.2)');
subplot(2,3,6), imshow(I_div), title('Division (Fig.1 / Fig.2)');
%2020-1-60-072
```

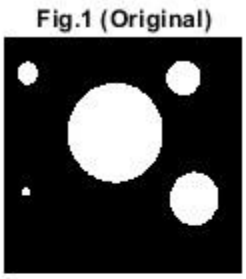


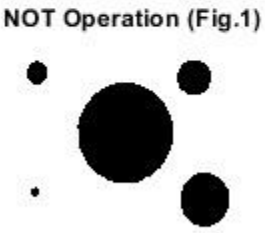


Problem 06:

```
clear all
close all;

% Read the two images
I1 = imread('Picture1.png');
I2 = imread('Picture2.png');

% Convert to grayscale if needed
if size(I1, 3) == 3
   I1 = rgb2gray(I1);
end
```

```
if size(I2, 3) == 3
    I2 = rgb2gray(I2);
end
% Resize I2 to match I1 dimensions
I2 = imresize(I2, size(I1));

% Convert to binary (if not already binary)
I1 = im2bw(I1);
I2 = im2bw(I2);

% AND operation
I_AND = I1 & I2;

% OR operation
I_OR = I1 | I2;

% NOT operation (only on Fig.1)
I_NOT = ~I1;

% Display images
figure();
subplot(2,3,1), imshow(I1), title('Fig.1 (Original)');
subplot(2,3,2), imshow(I2), title('Fig.2 (Original)');
subplot(2,3,3), imshow(I_AND), title('AND Operation');
subplot(2,3,4), imshow(I_OR), title('OR Operation');
subplot(2,3,5), imshow(I_NOT), title('NOT Operation (Fig.1)');
%2020-1-60-072
```

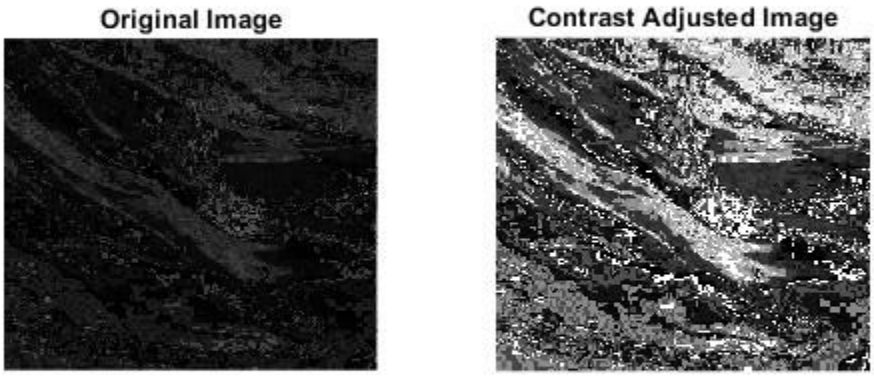




Problem 07:

```
clear all
close all;

% Read the image
I = imread('Picture4.png');

% Adjust contrast using imadjust
I_contrast = imadjust(I, stretchlim(I, [0.02 0.9]), []);
```

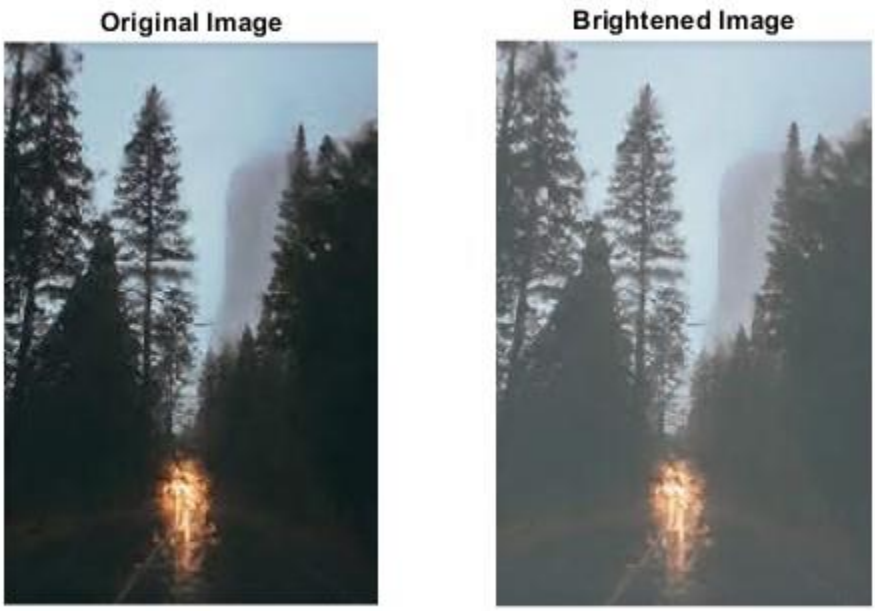
```
% Display the images
subplot(1,2,1), imshow(I), title('Original Image');
subplot(1,2,2), imshow(I_contrast), title('Contrast Adjusted Image');
%2020-1-60-072
```



Problem 08:

```
clear all
close all;

% Read the image
I = imread('Picture5.jpg');

% Brighten the image using imadjust
I_bright = imadjust(I, [], [0.25 1]); % Increase intensity
```

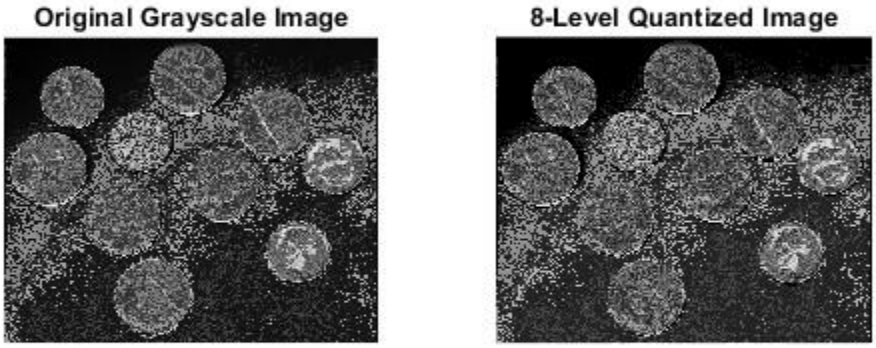
```
% Display the images
subplot(1,2,1), imshow(I), title('Original Image');
subplot(1,2,2), imshow(I_bright), title('Brightened Image');
%2020-1-60-072
```



Problem 09:

```
clear all
close all;

% Read the image
I = imread('Picture6.png');
```

```
% Quantize using built-in function
[I_quantized, map] = gray2ind(I, 8); % 8-level quantization

% Convert back to grayscale format for display
I_quantized = uint8(I_quantized * (255 / (8 - 1)));

% Display the images
subplot(1,2,1), imshow(I), title('Original Grayscale Image');
subplot(1,2,2), imshow(I_quantized), title('8-Level Quantized Image');
%2020-1-60-072
```



Problem 10:

```
clear all
close all;

% Read the image
I = imread('Picture7.png');

% Compute the negative image
I_negative =imcomplement(I);

% Display the images
subplot(1,2,1), imshow(I), title('Original Image');
subplot(1,2,2), imshow(I_negative), title('Negative Image');
%2020-1-60-072
```