CSC566

LAB 5

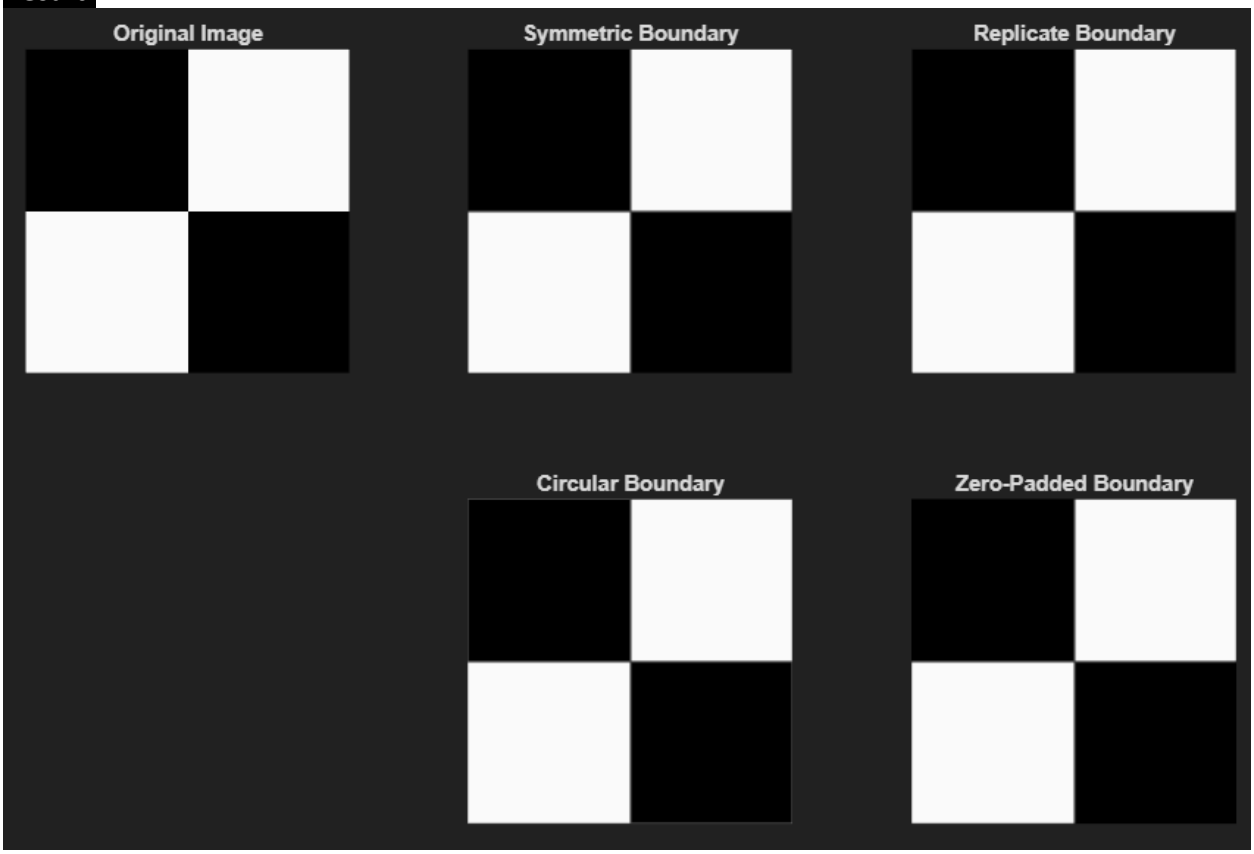
Name: Amir Hafizi Bin Musa
Student ID: 2024745815
Group: A4CDCS2306A

Assignment1:
Describe and explain the results of filtering 'pattern.tif' with average filter using different boundary options.

Code:

```
% Load the image
I_orig = imread('pattern.tif');

% convert it to grayscale for filtering
if size(I_orig, 3) == 3
    I_orig = rgb2gray(I_orig);
end

% Create an average filter
% A 5x5 filter will make the boundary effects more visible
h = fspecial('average', [5 5]);

% Apply the filter with different boundary options
I_symmetric = imfilter(I_orig, h, 'symmetric');
I_replicate = imfilter(I_orig, h, 'replicate');
I_circular = imfilter(I_orig, h, 'circular');
I_zero = imfilter(I_orig, h, 0); % Pad with zeros

% Display the results for comparison
figure('Name', 'Comparison of Boundary Options with Original');

% Display the original image first
subplot(2, 3, 1);
imshow(I_orig);
title('Original Image');

% Display the filtered results
subplot(2, 3, 2);
imshow(I_symmetric);
title('Symmetric Boundary');

subplot(2, 3, 3);
imshow(I_replicate);
title('Replicate Boundary');

subplot(2, 3, 5);
imshow(I_circular);
title('Circular Boundary');

subplot(2, 3, 6);
imshow(I_zero);
title('Zero-Padded Boundary');
```

Result:



Symmetric Boundary The edges of the filtered image will look smooth and natural. The transition from the original content to the "padded" area is seamless because the padding is a mirror reflection. This is generally the preferred method.
Replicate Boundary The edges will appear slightly blurred or "streaky" as the edge pixel values are pulled outwards and averaged. It's less smooth than the symmetric option.
Circular Boundary This will look strange unless the original image has a repeating pattern. You will see the content from the left edge of the image appear on the right side of the filtered border, and the content from the top appear at the bottom.
Zero-Padded Boundary This will produce a distinct dark border around the entire image. The average filter calculates the mean of the neighborhood, and since the out-of-bounds pixels are 0, they pull down the average, making the edges darker.

Assignment2:
Remove noises in 'Noisy1.tif' and 'Noisy2.tif' images.

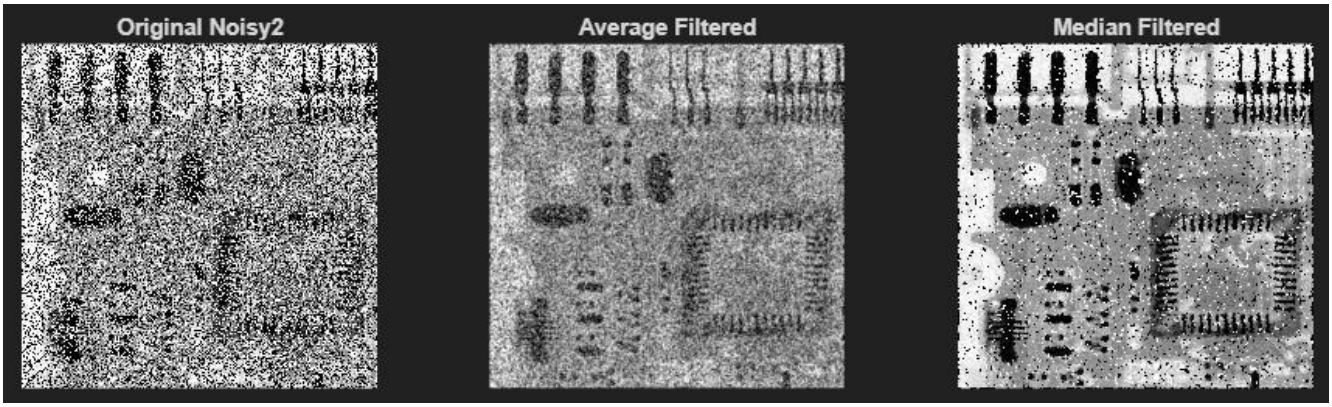
Code:

```
% Load the first noisy image
I1 = imread('Noisy1.tif');
if size(I1, 3) == 3, I1 = rgb2gray(I1); end

% Apply Average Filter
h_avg = fspecial('average', [3 3]);
I1_avg = imfilter(I1, h_avg);

% Apply Median Filter
I1_med = medfilt2(I1, [3 3]);

% Display results for Noisy1.tif
figure('Name', 'Filtering Noisy1.tif');
subplot(1, 3, 1); imshow(I1); title('Original Noisy1');
subplot(1, 3, 2); imshow(I1_avg); title('Average Filtered');
subplot(1, 3, 3); imshow(I1_med); title('Median Filtered');

% Load the second noisy image
I2 = imread('Noisy2.tif');
if size(I2, 3) == 3, I2 = rgb2gray(I2); end

% Apply Average Filter
I2_avg = imfilter(I2, h_avg); % Re-use the same filter

% Apply Median Filter
I2_med = medfilt2(I2, [3 3]);

% Display results for Noisy2.tif
figure('Name', 'Filtering Noisy2.tif');
subplot(1, 3, 1); imshow(I2); title('Original Noisy2');
subplot(1, 3, 2); imshow(I2_avg); title('Average Filtered');
subplot(1, 3, 3); imshow(I2_med); title('Median Filtered');
```

Result:

Noisy1.tif image:

Noisy2.tif image:



Assignment3:
Remove the small blobs in 'Blobs.jpg'.

Code:

```
% Load the original image
I_orig = imread('Blobs.jpg');

% Convert to grayscale for binarization
if size(I_orig, 3) == 3
    I_gray = rgb2gray(I_orig);
else
    I_gray = I_orig;
end

% Convert the grayscale image to a binary image
% This creates a black and white version where blobs are white (1) and background is black
(0).
BW = imbinarize(I_gray);

% Remove blobs smaller than a 50px
BW_clean = bwareaopen(BW, 50);

% 5. Display the results for a full comparison
figure('Name', 'Removing Small Blobs - Full Comparison');

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

% Display the intermediate binary image
subplot(1, 3, 2);
imshow(BW);
title('Binary Image');

% Display the final cleaned image
subplot(1, 3, 3);
imshow(BW_clean);
title('Cleaned Image (Min Size = 50 pixels)');
```

Result: