*Student: Melis Kilic*

**Linear filtering**

**LF-1. Linear low-pass filters**

**Task 1a Filtering with the averaging filter**

%% Task 1a

clearvars;

close all;

clc;

image = imread('board.png');

filter\_size = 3;

averaging\_filter = fspecial('average', [filter\_size filter\_size]);

convolved\_image = filter2(averaging\_filter, image, 'same');

convolved\_image = uint8(convolved\_image);

absolute\_difference = imabsdiff(image, convolved\_image);

% Display

figure('Position', [100, 100, 1000, 300]);

subplot(1, 3, 1);

imshow(image, []);

title('Original Image');

subplot(1, 3, 2);

imshow(convolved\_image, []);

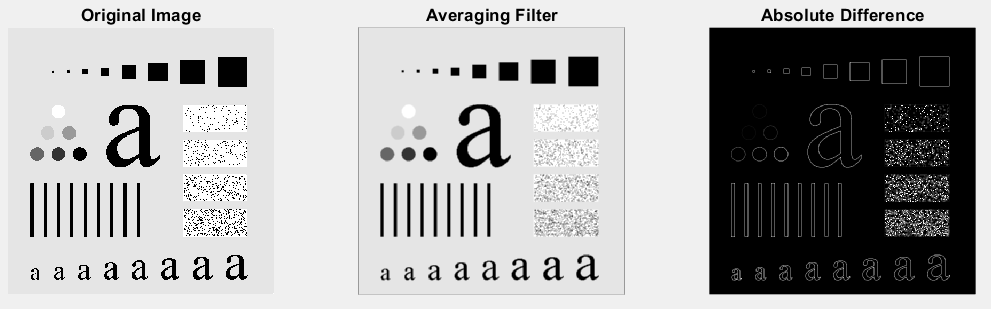
title('Averaging Filter');

subplot(1, 3, 3);

imshow(absolute\_difference, []);

title('Absolute Difference');

**Result of the code:**



% Analyse the impact of the kernel size on the result.

filter\_size2 = 35;

averaging\_filter2 = fspecial('average', [filter\_size2 filter\_size2]);

convolved\_image2 = filter2(averaging\_filter2, image, 'same');

convolved\_image2 = uint8(convolved\_image2);

absolute\_difference2 = imabsdiff(image, convolved\_image2);

% Display

figure('Position', [100, 100, 1000, 600]);

subplot(2, 3, 1);

imshow(image, []);

title('Original Image');

subplot(2, 3, 2);

imshow(convolved\_image, []);

title('Averaging Filter 3x3');

subplot(2, 3, 3);

imshow(absolute\_difference, []);

title('Absolute Difference');

subplot(2, 3, 4);

imshow(image, []);

title('Original Image');

subplot(2, 3, 5);

imshow(convolved\_image2, []);

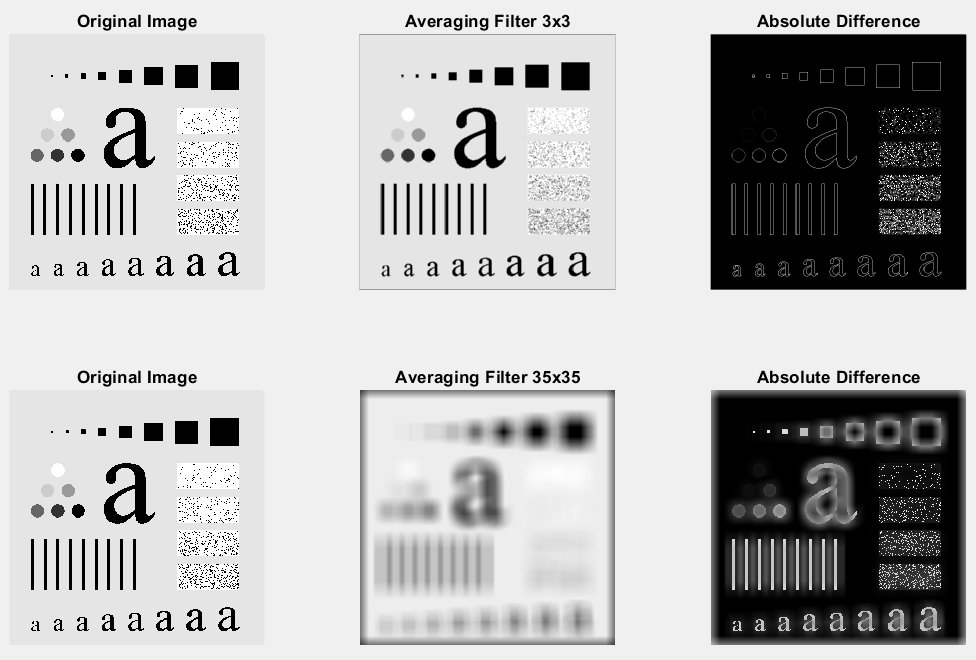
title('Averaging Filter 35x35');

subplot(2, 3, 6);

imshow(absolute\_difference2, []);

title('Absolute Difference');

**Result of the code:**

****

**Task 1b User-defined mask**

%% Task 1b

clearvars;

close all;

clc;

image = imread('board.png');

% Kernel (mask)

M = [1 2 1; 2 4 2; 1 2 1];

M = M/sum(sum(M));

convolved\_image = filter2(M, image, 'same');

convolved\_image = uint8(convolved\_image);

absolute\_difference = imabsdiff(image, convolved\_image);

% Display

figure('Position', [100, 100, 1000, 400]);

subplot(1, 3, 1);

imshow(image, []);

title('Original Image');

subplot(1, 3, 2);

imshow(convolved\_image, []);

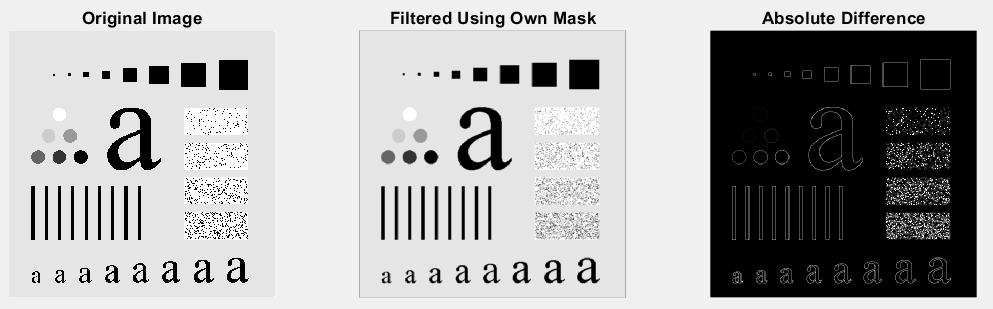
title('Filtered Using Own Mask');

subplot(1, 3, 3);

imshow(absolute\_difference, []);

title('Absolute Difference');

**Result of the code:**

****

**Task 1c Gaussian filter**

image = imread('jet.bmp');

filter1 = fspecial('gaussian', [15, 15], 3);

filter2 = fspecial('gaussian', [30, 30], 6);

filteredImage1 = imgaussfilt(image, 3);

filteredImage2 = imgaussfilt(image, 6);

% Display

figure;

subplot(2, 3, 1);

imshow(image);

title('Original Image');

subplot(2, 3, 2);

imshow(filteredImage1);

title('Filtrated for std=3');

subplot(2, 3, 3);

imshow(filteredImage2);

title('Filtrated for std=6');

subplot(2, 3, 5);

mesh(filter1);

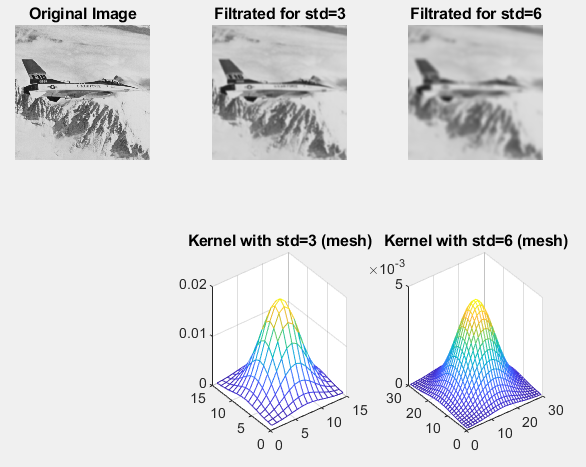
title('Kernel with std=3 (mesh)');

subplot(2, 3, 6);

mesh(filter2);

title('Kernel with std=6 (mesh)');

**Result of the code:**



**LF-2. Filters based on gradient (first derivative) approximation**

clearvars;

close all;

clc;

originalImage = imread('squares.bmp');

originalImage = double(originalImage);

% Sobel filters

sobelHorizontal = fspecial('sobel');

sobelVertical = sobelHorizontal';

% Convolution

sobelHorizontalResult = imfilter(originalImage, sobelHorizontal);

sobelVerticalResult = imfilter(originalImage, sobelVertical);

combinedFilter = sqrt(sobelHorizontalResult.^2 + sobelVerticalResult.^2);

% Display

figure;

subplot(2, 2, 1);

imshow(uint8(originalImage));

title('Original Image squares.bmp');

subplot(2, 2, 2);

imshow(sobelHorizontalResult, []);

title('Result of Sobel Horizontal');

subplot(2, 2, 3);

imshow(sobelVerticalResult, []);

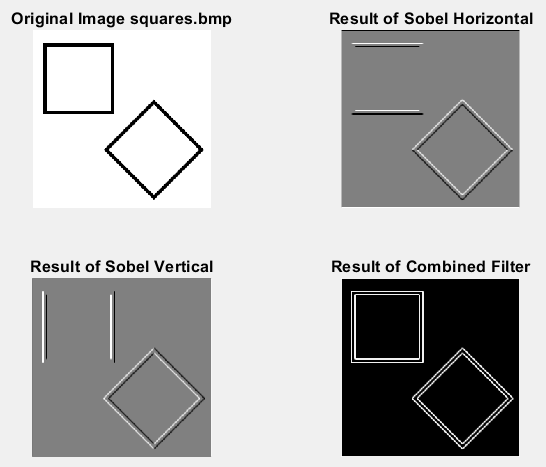
title('Result of Sobel Vertical');

subplot(2, 2, 4);

imshow(combinedFilter, []);

title('Result of Combined Filter');

**Result of the code:**



%%

% Change filename to moon.png and present the results:

clearvars;

close all;

clc;

originalImage = imread('moon.png');

originalImage = double(originalImage);

% Sobel filters

sobelHorizontal = fspecial('sobel');

sobelVertical = sobelHorizontal';

% Convolution

sobelHorizontalResult = imfilter(originalImage, sobelHorizontal);

sobelVerticalResult = imfilter(originalImage, sobelVertical);

combinedFilter = sqrt(sobelHorizontalResult.^2 + sobelVerticalResult.^2);

% Display

figure;

subplot(2, 2, 1);

imshow(uint8(originalImage));

title('Original Image moon.png');

subplot(2, 2, 2);

imshow(sobelHorizontalResult, []);

title('Result of Sobel Horizontal');

subplot(2, 2, 3);

imshow(sobelVerticalResult, []);

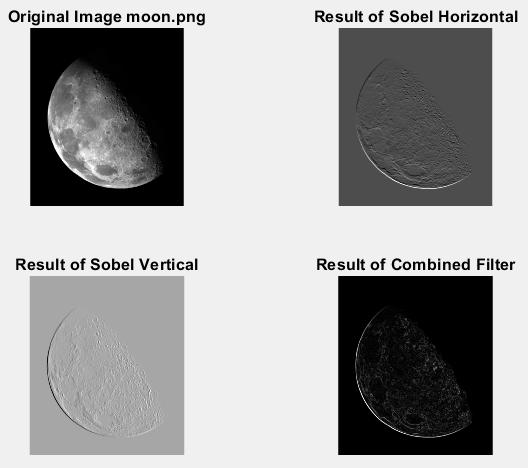
title('Result of Sobel Vertical');

subplot(2, 2, 4);

imshow(combinedFilter, []);

title('Result of Combined Filter');

**Result of the code:**



**LF-3. Canny filter**

clearvars;

close all;

clc;

originalImage = imread('moon.png');

edgesDefault = edge(originalImage, 'canny');

sigmaAdjusted = 1.5;

edgesSigmaAdjusted = edge(originalImage, 'canny', [], sigmaAdjusted);

thresholdAdjusted = 0.2;

edgesThresholdAdjusted = edge(originalImage, 'canny', thresholdAdjusted);

% Display

figure;

imshow(originalImage);

title('Fig. 1. Original Image moon.png');

figure;

imshow(edgesDefault);

title('Fig. 2. Canny filter with default parameters');

figure;

imshow(edgesSigmaAdjusted);

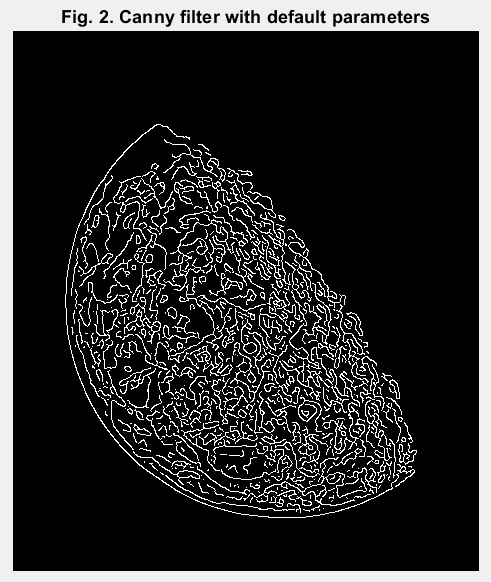
title('Fig. 3. Canny filter with adjusted standard deviation');

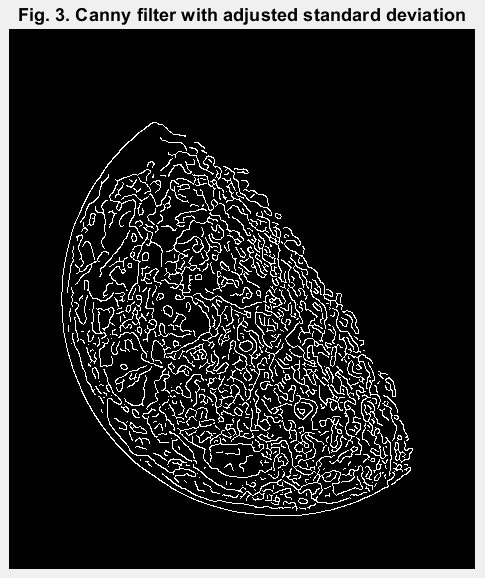
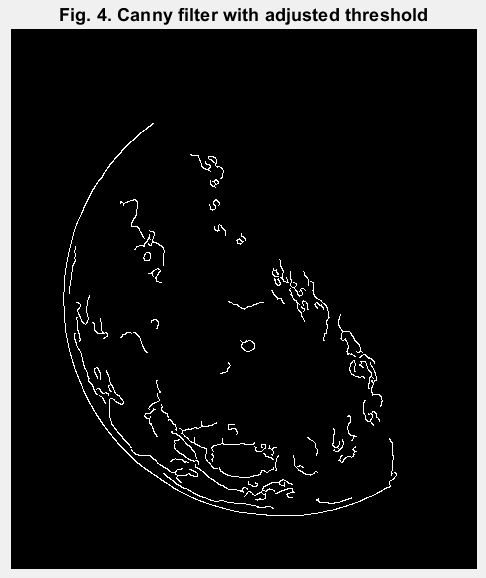
figure;

imshow(edgesThresholdAdjusted);

title('Fig. 4. Canny filter with adjusted threshold');

**Result of the code:**

** **

** **

**LF-4. Laplacian (second derivative of the image)**

clearvars;

close all;

clc;

originalImage = imread('moon.png');

% Laplacian mask

laplacianMask = fspecial('laplacian', 0);

% Convolution

laplacianFiltered = filter2(laplacianMask, double(originalImage), 'same');

% Normalizing

laplacianFilteredAbs = abs(laplacianFiltered);

maxValue = max(laplacianFilteredAbs(:));

laplacianFilteredNormalized = (laplacianFilteredAbs / maxValue) \* 255;

% Display

figure;

subplot(1, 2, 1);

imshow(uint8(originalImage));

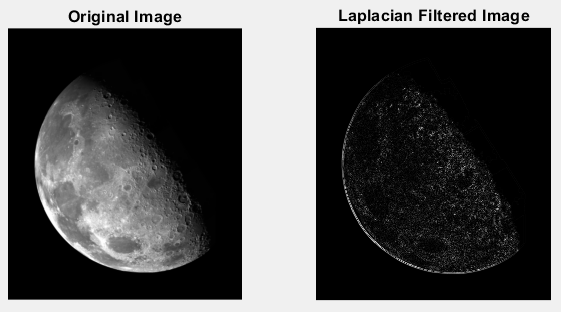
title('Original Image);

subplot(1, 2, 2);

imshow(uint8(laplacianFilteredNormalized));

title('Laplacian Filtered Image');

**Result of the code:**

****