**Assignment 3\_Turnitin Report**

**Sucheta Ravikanti 160040100**

**Swadha Sanghvi 16D070037**

**Neharika Jali 160040101**

[output, input] = myMeanShiftSegmentation('../data/baboonColor.png', 0.5, 0.1, 10, 30, 250);  
subplot(1, 2, 1), imshow((input)); title('Input Image');  
subplot(1, 2, 2), imshow(mat2gray(output)); title('Segmented Image');

% [output1, input] = myMeanShiftSegmentation('../data/baboonColor.png', 0.5, 0.1, 3, 30, 50);  
% [output2, input] = myMeanShiftSegmentation('../data/baboonColor.png', 0.5, 0.1, 6, 30, 100);  
% [output3, input] = myMeanShiftSegmentation('../data/baboonColor.png', 0.5, 0.1, 10, 30, 150);  
% [output4, input] = myMeanShiftSegmentation('../data/baboonColor.png', 0.5, 0.1, 15, 30, 250);  
% subplot(2, 2, 1), imshow(mat2gray(output1)); title('Sigma space = 3');  
% subplot(2, 2, 2), imshow(mat2gray(output2)); title('Sigma space = 6');  
% subplot(2, 2, 3), imshow(mat2gray(output3)); title('Sigma space = 10');  
% subplot(2, 2, 4), imshow(mat2gray(output4)); title('Sigma space = 15');  
  
function [segmented\_image, input\_image] = myMeanShiftSegmentation(path\_input, resizing\_factor, sigma\_color, sigma\_space, no\_of\_iter, no\_of\_nbs)  
 input\_image = im2double(imread(path\_input));  
 sigma = 0.5;  
 smoothened\_image = imfilter(input\_image, fspecial('gaussian', 6\*sigma, sigma));  
 resized\_image = imresize(smoothened\_image, resizing\_factor);  
 [height, width, channels] = size(resized\_image);  
 intensities\_image = reshape(resized\_image, [height\*width, 3]);  
 width\_vector = reshape(repmat([1:width], height, 1), width\*height, 1);  
 height\_vector = repmat(transpose([1:height]), width, 1);  
 vector = [intensities\_image/sigma\_color height\_vector/sigma\_space width\_vector/sigma\_space];  
  
 reference\_vector = vector;  
 Z = vector;  
 for i = 1:no\_of\_iter  
 disp(i);  
 [Idx, D] = knnsearch(reference\_vector, reference\_vector, 'k', no\_of\_nbs);  
 for j = 1:height\*width  
 weights = exp(-(D(j, :).^2));  
 weights = transpose(weights);  
 weights\_multiply = repmat(weights, 1, 3);  
 denominator = sum(weights);  
 numerator = sum(weights\_multiply.\*reference\_vector(uint16(Idx(j, :)), 1:3));  
 Z(j, 1:3) = numerator/denominator;  
 end  
 reference\_vector = Z;  
 end  
  
 segmented\_image = zeros(height, width, channels);  
 for k = 1:height\*width  
 i = uint16(reference\_vector(k, 4)\*sigma\_space);  
 j = uint16(reference\_vector(k, 5)\*sigma\_space);  
 segmented\_image(i, j, :) = reference\_vector(k, 1:3);  
 end  
 segmented\_image = imresize(segmented\_image, 2);  
end

**Mean Shift Segmentation**

The given image has the given input image and the mean shift segmented image. It is clearly seen that the pixel values have converged to a mean intensity value and the segments can be seen clearly in the segmented image.

The parameters used to attain this image are given as below :

**● Bandwidth for color or intensity (sigma\_color) = 0.1**

**● Bandwidth for space (sigma\_space) = 10**

**● Number of iterations = 30**

**● Number of neighbours in knnsearch = 200**

On tinkering with the parameters, it is observed that

● Segments formed decrease on increasing the bandwidth parameter of color intensity

● The image becomes smooth in segments on increasing the spatial bandwidth parameter

● Increasing the number of iterations shows better convergence

The above observations can be proved using the results of the simulations below : Note that the other parameters are those from the best segmented image. The changed parameters are written for each image.

It is clearly seen that the segments are decreasing on increasing the sigma colour value. The first image has fine segments due to a smaller value of sigma space which allows a finer window for the intensity convergence and better colour mixing.

**Increasing the bandwidth parameter of color intensity :**

It is observed that the number of segments increase and are very fine in spatial domain as the bandwidth of space increases.

**Variation of the sigma space of bandwidth of space :**

As clearly seen, on increasing the number of iterations, the segments are formed better and there is a larger convergence seen due to the increased number of steps in gradient ascent and better convergence to the mean.

**Increasing the number of iterations :**

load('boat.mat')

input = im2double(imageOrig);

input = input./255.0;

%imtool(input)

EigenImage\_1 = zeros(size(input));

EigenImage\_2 = zeros(size(input));

min(min(input))

[dy, dx] = meshgrid(-1:1, -1:1);

Size = size(input);

vector = zeros(Size(1,1), Size(1,2), 15);

% parameters

sig = 0.01;

sigma = 3;

k = 0.03;

threshold = 0.015;

x = 3;

%%%% derivative matrix Ix and Iy

input = imgaussfilt(input, sig);

%imtool(input)

I\_x = conv2(input, dx, 'same');

I\_y = conv2(input, dy, 'same');

I\_x2 = I\_x.\*I\_x;

I\_y2 = I\_y.\*I\_y;

I\_xy = I\_x.\*I\_y;

%%%%% weight convolution matrix

dim = max(1, 5);

m = dim; n = dim;

[h1, h2] = meshgrid(-(m-1)/2 :(m-1)/2, -(n-1)/2 :(n-1)/2 );

v = exp(-(1.0)\*(h1.^2 + h2.^2)/(2\*sigma^2));

sum\_weight = sum(sum(v));

v = v./sum\_weight;

% structure tensor

I\_X2 = conv2(I\_x2, v, 'same');

I\_Y2 = conv2(I\_y2, v, 'same');

I\_XY = conv2(I\_xy, v, 'same');

% harris corner measure

for i = 1:Size(1,1)

for j = 1:Size(1,2)

C = [1, I\_X2(i,j)+ I\_Y2(i,j), (I\_X2(i,j))\*(I\_Y2(i,j)) - (I\_XY(i,j).^2)];

Roots = roots(C);

EigenImage\_1(i,j) = min(Roots);

EigenImage\_2(i,j) = max(Roots);

end

end

R = I\_X2.\*I\_Y2 - I\_XY.\*I\_XY - k\*(I\_X2.^2 + I\_Y2.^2 + 2\*I\_X2.\*I\_Y2);

output\_Image = ordfilt2(R, x.^2, true(x));

final\_image = (R == output\_Image) & (R > threshold);

[row,col] = find(final\_image);

figure, colormap(jet(10)), imshow(mat2gray(input)), hold on,

plot(col, row, 'r^', 'MarkerSize', 3.5)

imtool(mat2gray(final\_image))

imtool((I\_x))

imtool(I\_y)

imtool(mat2gray(EigenImage\_1))

imtool(mat2gray(EigenImage\_2))

histogram(R)

**Harris Corner Detection**

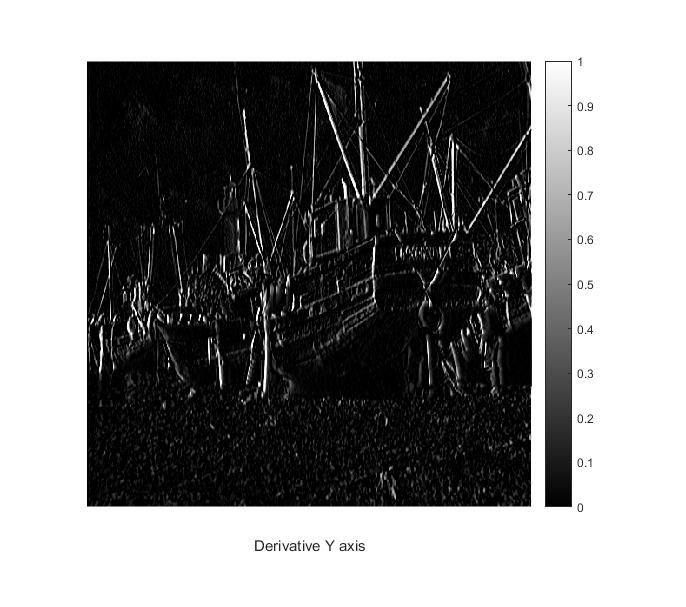
Sucheta Ravikanti (160040100)

Swadha Sanghvi (16D070037)

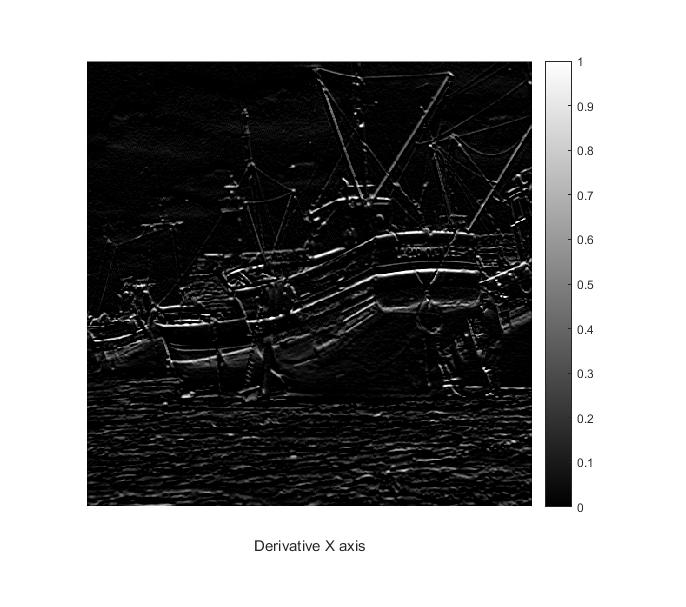
Neharika Jali (160040101)

The following two image represents the derivative along the y axis or the vertical derivative and the horizontal or the derivative along the x-axis of the given image respectively.

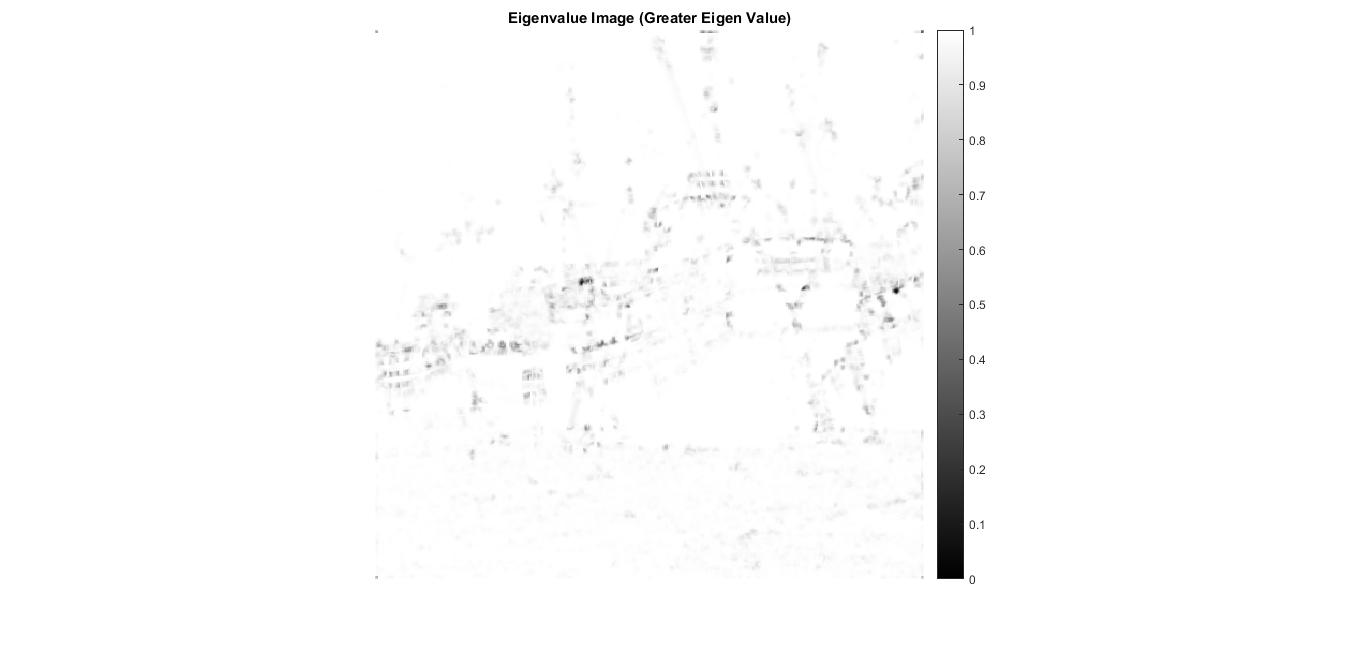
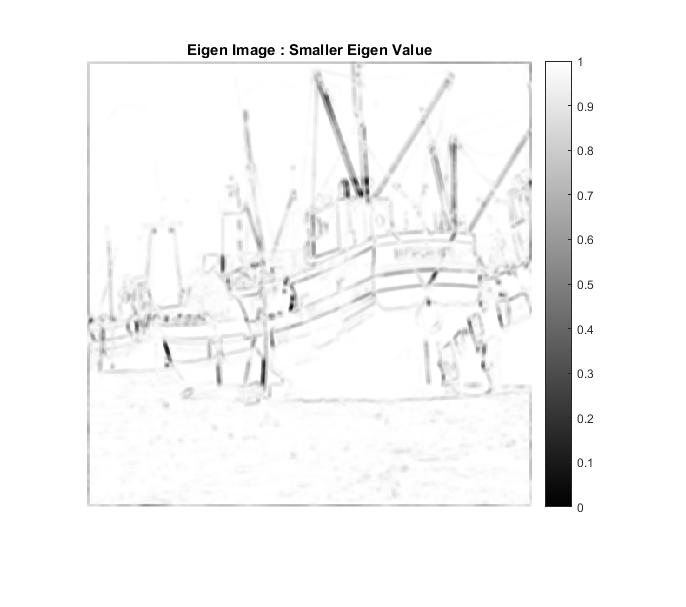
**Vertical or derivative along y-axis :**



**Horizontal or derivative along x-axis :**

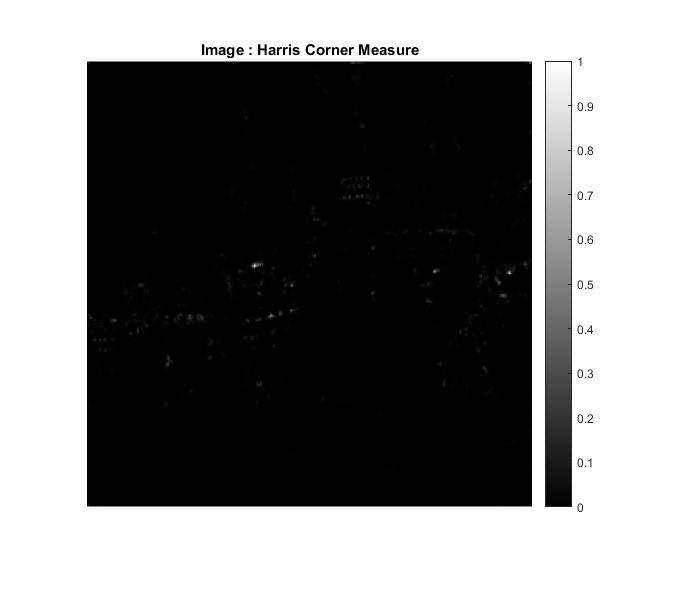


**Image of the smaller and greater eigenvalue of the structure tensor evaluated at each pixel respectively :**

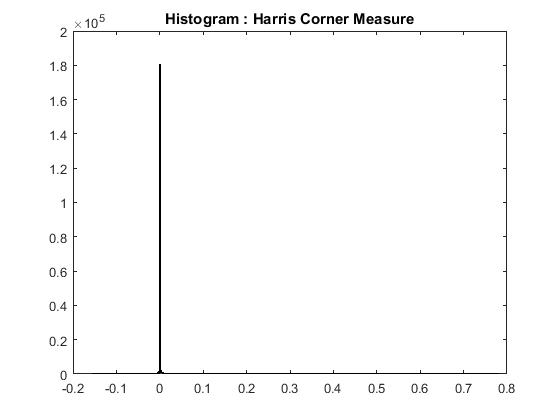


**Note: Please check the folder corresponding to the output Images for clearer images**

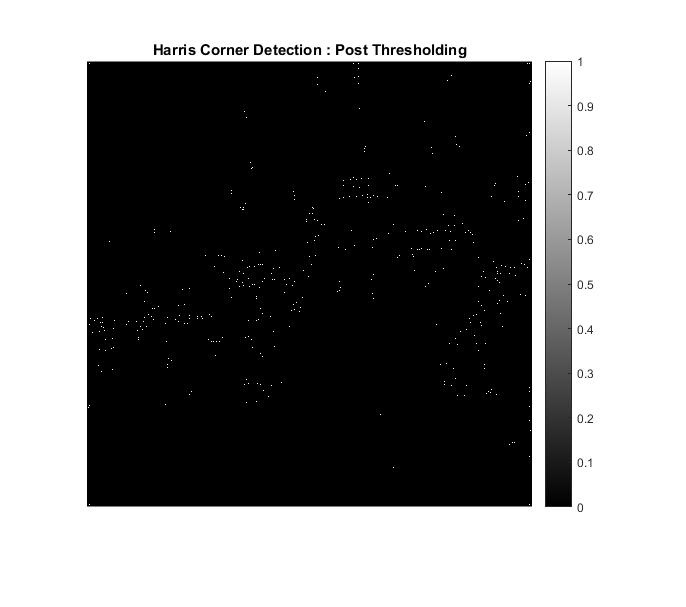
**Harris Cornerness Measure :**



**Histogram of the Harris Cornerness measure :**



**Post thresholding Image of the harris corner detection output :**



**Final Image with corners marked**

Parameters used :

* **Window size = 5X5 (Window deciding the weights of the neighbours)**
* **K = 0.03 (used in calculating the harris corner measure)**
* **Sigma\_1 = 3 (Variance for the gaussian filter for smoothing the derivatives)**
* **Sigma\_2 = 0.01 (Variance for the gaussian filter for smoothing the input image)**

