Name: MEHTA NIHAR NIKHIL

Roll No: 13D100011

Q1 Harris Corner Detection

Code:

```
function [corner_coeff]=myHarrisCornerDetector(orig_image,k,sigma_x,sigma_y)


min_intensity=min(min(orig_image));
max_intensity=max(max(orig_image));
orig_image=(orig_image-min_intensity)/(max_intensity-min_intensity);
[row,col]=size(orig_image);

orig_image=double(orig_image);



partialX=Ix(orig_image);
partialY=Iy(orig_image);

A=partialX.^2;
B=partialY.^2;
C=partialX.*partialY;

kernelX = [[-1, 0, 1];
            [-1, 0, 1];
            [-1, 0, 1]];

kernelY = [[-1, -1, -1];
            [0, 0, 0];
            [1, 1, 1]];

kernel=exp(-0.5*((kernelX.^2)/(2*sigma_x^2) + kernelY.^2/(2*sigma_y^2)));

A=apply_kernel(A,kernel);
B=apply_kernel(B,kernel);
C=apply_kernel(C,kernel);

eigen1=zeros(row,col);
eigen2=zeros(row,col);

for i=1:row
    for j=1:col
        temp_matrix=[A(i,j) C(i,j); C(i,j) B(i,j)];
        eig_output=eig(temp_matrix);
```

```
        eigen1(i,j)=eig_output(1);
        eigen2(i,j)=eig_output(2);
    end
end


trace=(A+B);
Det=A.*B-C.*C;
corner_coeff=Det-k*trace.^2;



figure, imshow(partialX), title('X derivative');
figure, imshow(partialY),title('Y derivative');
figure, imshow(eigen1),colorbar, title('1st eigen value');
figure, imshow(eigen2),colorbar, title('2nd eigen value');
figure, imshow(max(corner_coeff,0)),colorbar,title('Harris Cornerness
measure');

radius=30;
threshold=0.1;
% perform non-maximal suppression using ordfilt2
n = ordfilt2(corner_coeff, radius^2, ones([radius radius]));

% display corner pixels on the original image
corners = (corner_coeff==n)&(n>threshold);

list=find(corners==1);

figure,imshow(orig_image),title('Corners on original image');
hold on
plot(ceil(list/row),mod(list,row),'r+', 'MarkerSize', 5, 'LineWidth', 3)
end

function [xderiv]=Ix(image)

kernel=[-1,0,1];
[row,col]=size(image);
xderiv=zeros(row,col);

for i=1:row
    for j=2:col-1
        xderiv(i,j)=sum(kernel.*[image(i,j-1) image(i,j) image(i,j+1)]);
    end
end

end




function [yderiv]=Iy(image)
```

```
kernel=[-1,0,1];
[row,col]=size(image);
yderiv=zeros(row,col);
for i=2:row-1
    for j=1:col
        yderiv(i,j)=sum(kernel.*[image(i-1,j) image(i,j) image(i+1,j)]);
    end
end

end

function [new_image]=apply_kernel(image,kernel)

[row,col]=size(image);
[krow,kcol]=size(kernel);

new_image=zeros(row,col);

midrow=floor((krow-1)/2);
midcol=floor((kcol-1)/2);

for i=1+midrow:row-midrow
    for j=1+midcol:col-midcol
        new_image(i,j)=sum(sum(kernel.*image(i-midrow:i+midrow,j-
midcol:j+midcol)));
    end
end

end
```

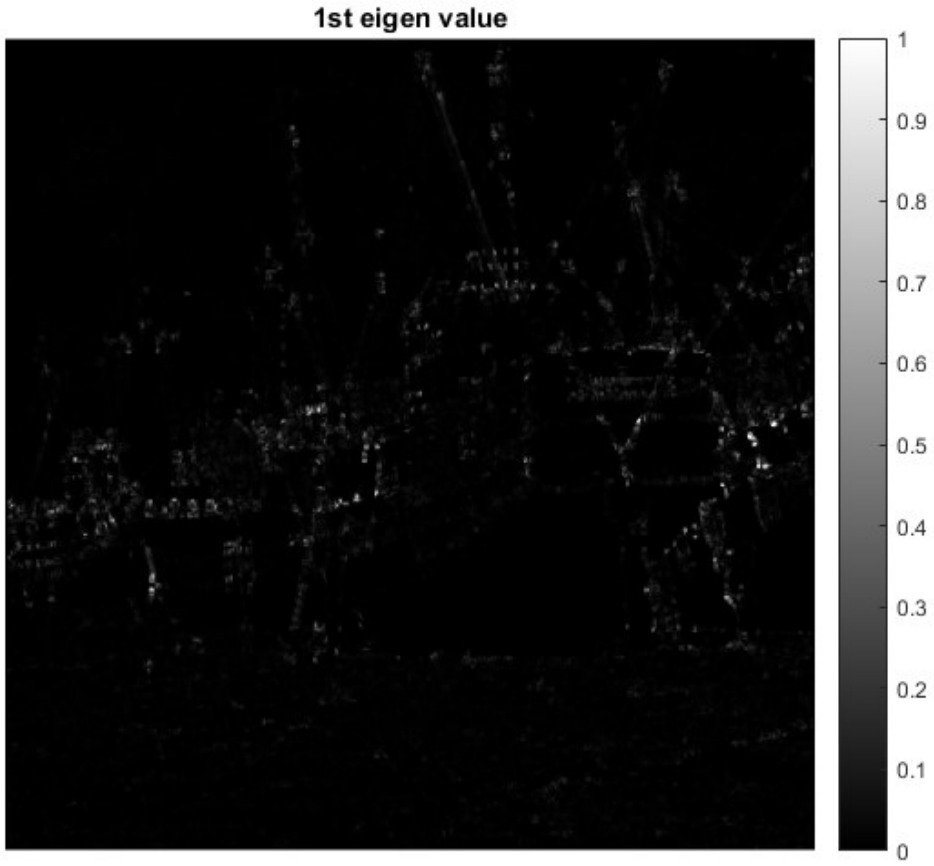
Original image

Derivative Images:

X derivative

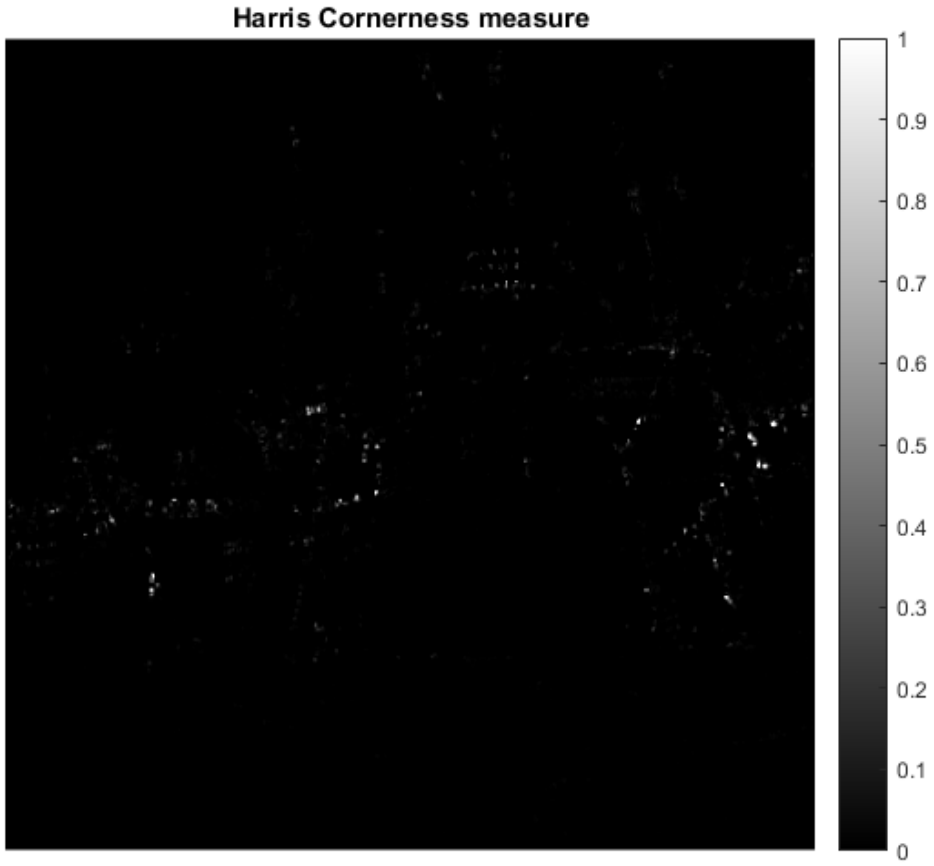
Y derivative

Eigen value 1:

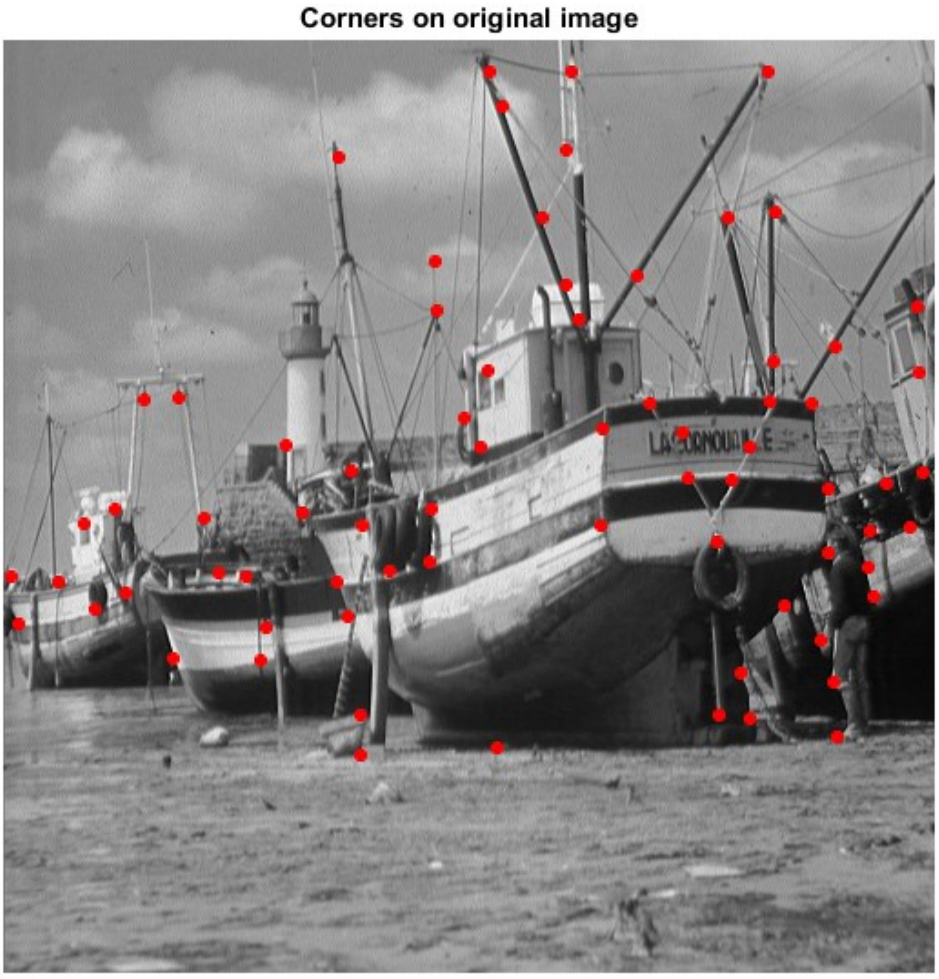


Eigen value 2:

Harris Corner measure:



Final result:



Parameters used:

K=0.05

Sigma_x=10

Sigma_y=10

Effects of parameters:

If K is decreased, more corners are captured. Generally the K value must be around 0.04-0.06. A higher value of K results in missing out of the corners.

If sigma_x and sigma_y are not used, then some corners are missed out because of the low gradient changes with the background.

If radius of thresholding is increased, less corners are captured. This must be an optimum so that neither specious corners appear, nor true corners are skipped.

If threshold parameter is increased, less corners of captured.