Table of Contents

Devon Quaternik

Elen 644 HW5

```
clear;
close all;

im1 = checkerboard(20,4,4);
im2 = rgb2gray(imread('4.1.05.tiff'));
```

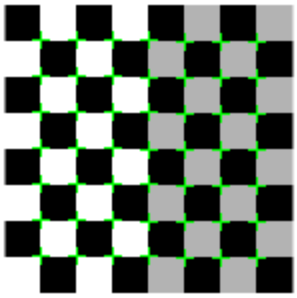
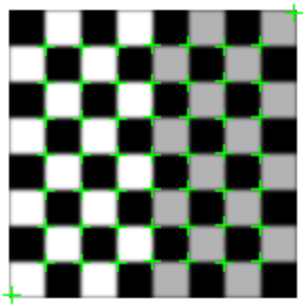
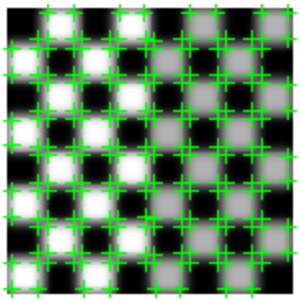
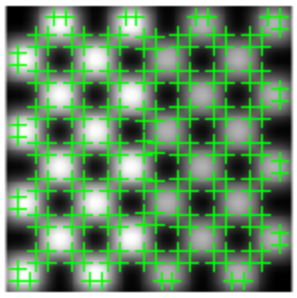
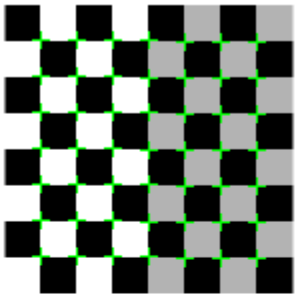
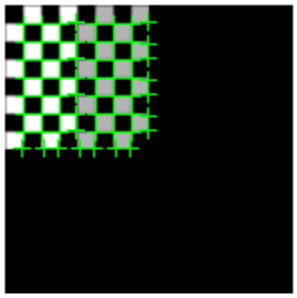
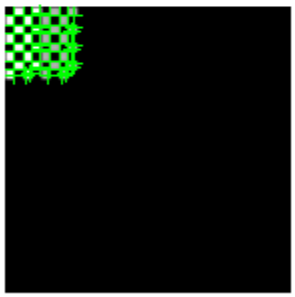
Problem 1

Part a

```
ogpoints = detectHarrisFeatures(im1);
figure;
subplot(2,2,1),imshow(im1); hold on;
plot(ogpoints);

numpoints = length(ogpoints)

disp('The corners are where I expected to see them. At the edge of an
 image a corner turns into an edge and should not be detected.');
disp('The strongest corners are those in the middle of the image.
 Theyu return the highest metric.');

% Part b
h1 = fspecial('Gaussian',5,1);
h2 = fspecial('Gaussian',15,3);
h3 = fspecial('Gaussian',25,5);
smim1 = filter2(h1,im1);
smim2 = filter2(h2,im1);
smim3 = filter2(h3,im1);

smpts1 = detectHarrisFeatures(smim1);
smptsl1 = length(smpts1)
smpts2 = detectHarrisFeatures(smim2);
smptsl2 = length(smpts2)
smpts3 = detectHarrisFeatures(smim3);
smptsl3 = length(smpts3)

subplot(2,2,2),imshow(smim1,[]); hold on;
```

```
plot(smpts1);

subplot(2,2,3),imshow(smim2,[]); hold on;
plot(smpts2);

subplot(2,2,4),imshow(smim3,[]); hold on;
plot(smpts3);

disp('For the first, the corners are mostly where I expect, with a
 couple extras on the image corners. The second and third are much
 less accurate, with a significant number of extras detected');
disp('The extra corners show up in places where the corners are
 blurred. This has to do with the shape of a gaussian, and makes some
 sense as to why they are there');

% Part c
h = (1/16)*[1 4 6 4 1];
dim = dscale2(im1,h,3);
dpts1 = detectHarrisFeatures(dim(:,:,1));
dptsl1 = length(dpts1)
dpts2 = detectHarrisFeatures(dim(:,:,2));
dptsl2 = length(dpts2)
dpts3 = detectHarrisFeatures(dim(:,:,3));
dptsl3 = length(dpts3)

figure;
subplot(2,2,1),imshow(im1); hold on;
plot(ogpoints);
subplot(2,2,2),imshow(dim(:,:,1),[]); hold on;
plot(dpts1);
subplot(2,2,3),imshow(dim(:,:,2),[]); hold on;
plot(dpts2);
subplot(2,2,4),imshow(dim(:,:,3),[]); hold on;
plot(dpts3);

disp('The corners of the downsampled images show up more where I
 am expecting them to. They line up pretty well with the actual
 corners.');
disp('The first couple levels gain a few extra points, but the lower
 levels lose them quickly. None are perfectly accurate compared to
 original');


numpoints =

    49

The corners are where I expected to see them. At the edge of an image
 a corner turns into an edge and should not be detected.
The strongest corners are those in the middle of the image. Theyu
 return the highest metric.

smptsl1 =
```

```
    51


smptsl2 =

   226


smptsl3 =

   226

For the first, the corners are mostly where I expect, with a couple
 extras on the image corners. The second and third are much less
 accurate, with a significant number of extras detected
The extra corners show up in places where the corners are blurred.
 This has to do with the shape of a gaussian, and makes some sense as
 to why they are there

dptsl1 =

    63


dptsl2 =

    62


dptsl3 =

    13

The corners of the downsampled images show up more where I am
 expecting them to. They line up pretty well with the actual corners.
The first couple levels gain a few extra points, but the lower levels
 lose them quickly. None are perfectly accurate compared to original
```


Problem 2

Part a

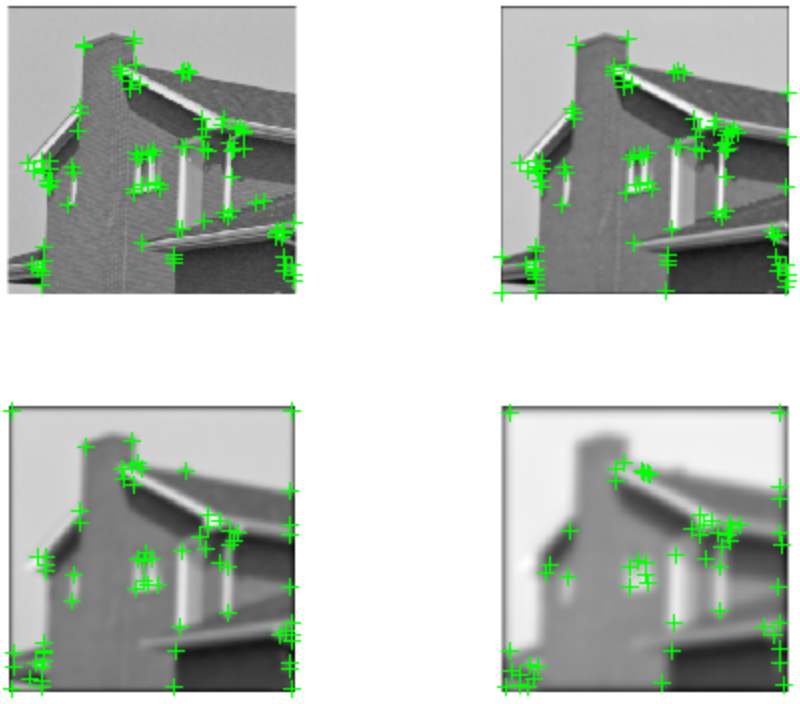
```
ogpoints = detectHarrisFeatures(im2);
numpoints = length(ogpoints);

figure;
subplot(2,2,1),imshow(im2); hold on;
plot(ogpoints);

disp('The original Image shows a good number of points, although not
 in ever location I would expect, and a few in places I would not.
 Overall works well');

% Part b
h1 = fspecial('Gaussian',5,1);
h2 = fspecial('Gaussian',15,3);
h3 = fspecial('Gaussian',25,5);
smim1 = filter2(h1,im2);
smim2 = filter2(h2,im2);
smim3 = filter2(h3,im2);

smpts1 = detectHarrisFeatures(smim1);
smptsl1 = length(smpts1)
smpts2 = detectHarrisFeatures(smim2);
smptsl2 = length(smpts2)
smpts3 = detectHarrisFeatures(smim3);
smptsl3 = length(smpts3)

subplot(2,2,2),imshow(smim1,[]); hold on;
plot(smpts1);
subplot(2,2,3),imshow(smim2,[]); hold on;
plot(smpts2);
subplot(2,2,4),imshow(smim3,[]); hold on;
plot(smpts3);

disp('Blurring the images has similar results as before, generating
 a lot of extra points where they should not be, getting worse with
 bigger filters');

% Part c
h = (1/16)*[1 4 6 4 1];
dim = dscale2(im2,h,3);
dpts1 = detectHarrisFeatures(dim(:,:,1));
dptsl1 = length(dpts1)
dpts2 = detectHarrisFeatures(dim(:,:,2));
dptsl2 = length(dpts2)
dpts3 = detectHarrisFeatures(dim(:,:,3));
dptsl3 = length(dpts3)

figure;
subplot(2,2,1),imshow(im2); hold on;
```

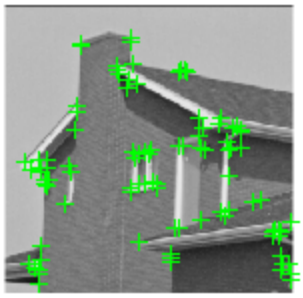
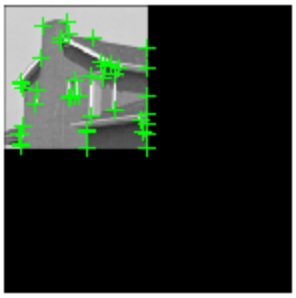
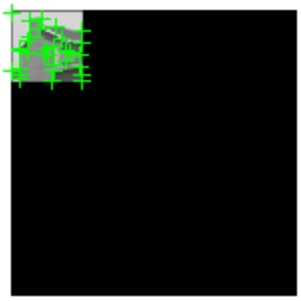
```
plot(ogpoints);
subplot(2,2,2),imshow(dim(:,:,1),[]); hold on;
plot(dpts1);
subplot(2,2,3),imshow(dim(:,:,2),[]); hold on;
plot(dpts2);
subplot(2,2,4),imshow(dim(:,:,3),[]); hold on;
plot(dpts3);

disp('Downscaling again helps eliminate some of the extra points
 brought about by blurring. As you go down in levels you quickly lose
 some extra points, and those that remain seem to be the strongest.
 These could be used for identification of points in space');

% Part d

disp('Compared to the last image, the corner detection is very
 similar. The downsampled images seemed like the best way to execute
 corner detection and finding which stick out best.');
```

```
The original Image shows a good number of points, although not in ever
 location I would expect, and a few in places I would not. Overall
 works well

smptsl1 =

    90


smptsl2 =

    67


smptsl3 =

    58

Blurring the images has similar results as before, generating a lot
 of extra points where they should not be, getting worse with bigger
 filters

dptsl1 =

    44


dptsl2 =

    31


dptsl3 =

    19
```

Downscaling again helps eliminate some of the extra points brought about by blurring. As you go down in levels you quickly lose some extra points, and those that remain seem to be the strongest. These could be used for identification of points in space

Compared to the last image, the corner detection is very similar. The downsampled images seemed like the best way to execute corner detection and finding which stick out best.



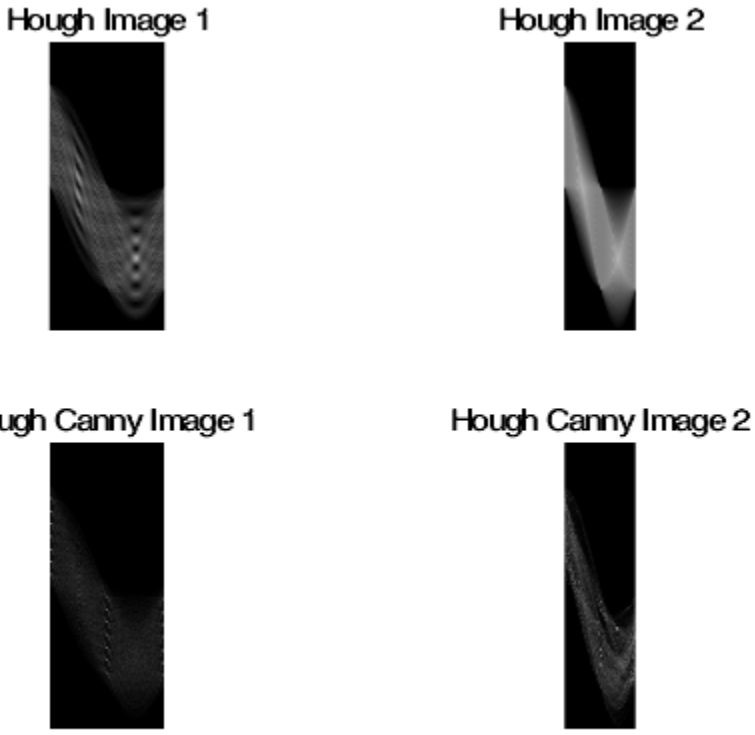

Problem 3

Part a

```
cim1 = edge(im1,'Canny');
cim2 = edge(im2,'Canny');
him1 = hough(im1);
him2 = hough(im2);
him3 = hough(cim1);
him4 = hough(cim2);

figure;
subplot(2,2,1),imshow(him1,[]);
title('Hough Image 1');
subplot(2,2,2),imshow(him2,[]);
title('Hough Image 2');
subplot(2,2,3),imshow(him3,[]);
title('Hough Canny Image 1');
subplot(2,2,4),imshow(him4,[]);
title('Hough Canny Image 2');

% Part b
disp('The origin is assumed to be in the top left corner');

% Part c
```

```
disp('Lines aree at a 45 degree angle because about half the edges run
 vertically and the other half horizontally.');
disp('The edge images cut out a lot of the noise that the extra info
 brings. You are left with strong points on the edges, evenly spaced
 as the checkerboard is.');
disp('For the house image, the edge image is still not super clear,
 but this is due to the large number of small edges picked up in the
 bricks. The strongest show up along the angle of the roof edge');
```

```
The origin is assumed to be in the top left corner
Lines aree at a 45 degree angle because about half the edges run
 vertically and the other half horizontally.
The edge images cut out a lot of the noise that the extra info brings.
 You are left with strong points on the edges, evenly spaced as the
 checkerboard is.
For the house image, the edge image is still not super clear, but this
 is due to the large number of small edges picked up in the bricks.
 The strongest show up along the angle of the roof edge
```

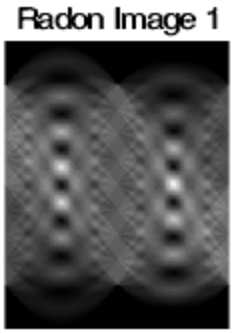


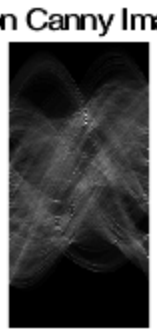
Problem 4

```
rim1 = radon(im1);
rim2 = radon(im2);
rim3 = radon(cim1);
rim4 = radon(cim2);
```

```
figure;
subplot(2,2,1),imshow(rim1,[]);
title('Radon Image 1');
subplot(2,2,2),imshow(rim2,[]);
title('Radon Image 2');
subplot(2,2,3),imshow(rim3,[]);
title('Radon Canny Image 1');
subplot(2,2,4),imshow(rim4,[]);
title('Radon Canny Image 2');

disp('The radon image gives an evenly spaced circle for the
 checkerboard pattern. Its bright and dark appear to align with the
 checkerboard edges.');
disp('Again the edge image cuts a lot of the noise the extra info
 brings. You are left with strong lines along the angles of the
 checkerboard.');
disp('The house image radon transform has high spots in similar places
 as the hough transform');
```

```
The radon image gives an evenly spaced circle for the checkerboard
 pattern. Its bright and dark appear to align with the checkerboard
 edges.
Again the edge image cuts a lot of the noise the extra info brings.
 You are left with strong lines along the angles of the checkerboard.
The house image radon transform has high spots in similar places as
 the hough transform
```





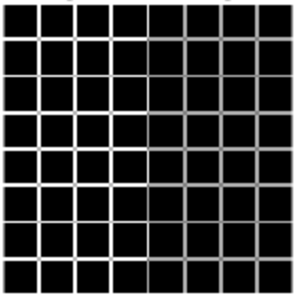
Problem 5

```
sb1 = [1 2 1; 0 0 0; -1 -2 -1];
sb2 = [1 0 -1; 2 0 -2; 1 0 -1];
x1 = conv2(im1,sb1,'valid'); y1 = conv2(im1, sb2,'valid');
x2 = conv2(im2,sb1,'valid'); y2 = conv2(im2, sb2,'valid');

mag1 = sqrt((x1).^2 + (y1).^2);
ang1 = atan2(y1,x1);
mag2 = sqrt((x2).^2 + (y2).^2);
ang2 = atan2(y2,x2);

%normalize magnitudes angles between 0 and 1 for hsv image;
nang1 = ang1 - min(ang1(:));
nang1 = nang1 ./ max(nang1(:));
nang2 = ang2 - min(ang2(:));
nang2 = nang2 ./ max(nang2(:));

nmag1 = mag1 - min(mag1(:));
nmag1 = nmag1 ./ max(nmag1(:));
nmag2 = mag2 - min(mag2(:));
nmag2 = nmag2 ./ max(nmag2(:));

% Part a
figure;
subplot(2,2,1),imshow(nmag1,[]);
title('Magnitude Image 1');
subplot(2,2,2),imshow(nang1,[]);
title('Angle Image 1');
subplot(2,2,3),imshow(nmag2,[]);
title('Magnitude Image 2');
subplot(2,2,4),imshow(nang2,[]);
title('Angle Image 2');

disp('An edge is 2 pixels thick');
disp('The response at the corners is slightly weaker than the rest
 of the edges. This is due to the smoothing nature of the sobel
 filter.');

% Part b
[m,n] = size(nmag1);
im1hsv = ones(m,n,3);
im1hsv(:,:,1) = nang1;
im1hsv(:,:,3) = nmag1;
im1rgb = hsv2rgb(im1hsv);

figure;
subplot(2,3,1),imshow(im1rgb,[]);
title('Color representation');
subplot(2,3,2),imshow(x1,[]);
title('Horizontal Gradient Image');
subplot(2,3,3),imshow(y1,[]);
title('Vertical Gradient Image');
```

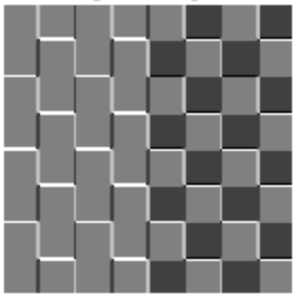
```
[m,n] = size(nmag2);
im2hsv = ones(m,n,3);
im2hsv(:,:,1) = nang2;
im2hsv(:,:,3) = nmag2;
im2rgb = hsv2rgb(im2hsv);

subplot(2,3,4),imshow(im2rgb,[]);
title('Color Representation');
subplot(2,3,5),imshow(x2,[]);
title('Horizontal Gradient Image');
subplot(2,3,6),imshow(y2,[]);
title('Vertical Gradient Image');

disp('The blue and red appear to be related to horizontal angles.
 Blue for white square, red for black square. Purple and green are
 related to vertical lines. Purple corresponds to white squre, green
 to black.');
disp('The color change in the curvature of the pipe has to do with the
 change of the angle as you move through the curve. Because the angles
 tangent is used to create a hue, it will change with the curves
 angle');
```

```
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
An edge is 2 pixels thick
The response at the corners is slightly weaker than the rest of the
 edges. This is due to the smoothing nature of the sobel filter.
The blue and red appear to be related to horizontal angles. Blue for
 white square, red for black square. Purple and green are related to
 vertical lines. Purple corresponds to white squre, green to black.
The color change in the curvature of the pipe has to do with the
 change of the angle as you move through the curve. Because the angles
 tangent is used to create a hue, it will change with the curves angle
```

Magnitude Image 1



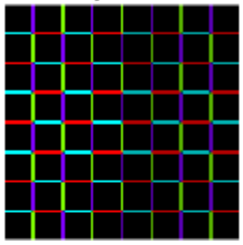
Angle Image 1

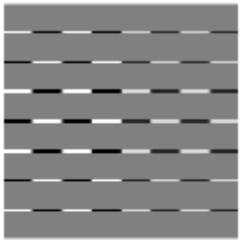


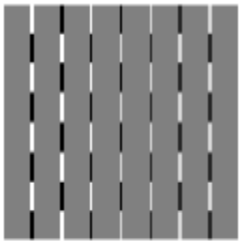
Magnitude Image 2

Angle Image 2

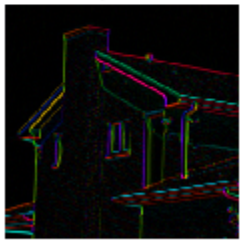
Color representation Horizontal Gradient Image Vertical Gradient Image

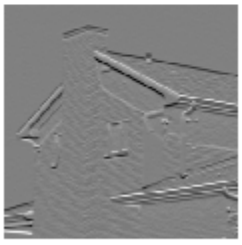


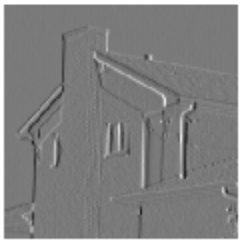




Color Representation Horizontal Gradient Image Vertical Gradient Image







Published with MATLAB® R2015a