ECE 280L – Fall 2020

Image Processing

Contents

1 Exercise 1

The images show 3 circles with different colors (red, green, blue) close to the center. As the circles expand, the color fades away. The greater the p value the faster the color fades away.

2 Exercise 2

The images generated show randomly colored dots. When the code is recompiled. The dots appear as if they were scattered in a different order.

3 Exercise 3

The 5 point moving average results in a smoother plot than the 2 point moving average. This is due to the 5 point moving average averaging larger samples of data. The 5 point moving average also has a lower peak than the 2 point moving average

4 Exercise 4

The derivative approximation using 11 points has a lower peak and a lower minimum than the derivative approximation using 51 points. Furthermore, the separation between points is higher using 11 points than when using 51 points. Lastly, the approximations using 51 points is more intricate and has more curves than when using 11 points.

5 Exercise 5

The 10x10 blur applies a scaled 10 row, 10 column blur to the coins image. The resulting image appears to be blurred equally horizontally and vertically. The 2x50 blur applies a a scaled 2 row, 50 column blur to the coins image. The resulting image appears to have a wider blur horizontally than vertically.The 50x2 blur applies a a scaled 50 row, 2 column blur to the coins image. The resulting image appears to have a wider blur vertically than horizontally.

6 Exercise 6

According to the MATLAB documentation of the conv2 function, the valid option returns the convolution parts computed with unpadded edges, whereas conv2 with the same option returns the central part of the convolution such that the returned element has the same size as the first parameter fed to conv2. The resulting images using the valid option are slightly cropped and smaller when compared to the images using the same option.

7 Exercise 7

1. The first image shows the result of a 5x5 Gaussian Blur. The Guassian blur resulting image is less blurry than the 10x10 blur.
2. The second image shows the vertical edges of the coins using a Prewitt operator that detects changes in gray level from left to right.
3. The third image shows the horizontal edges of the coins using a Prewitt operator that detects changes in gray level from top to bottom.
4. The fourth image displays the normalized square root of the sum of squares of the previous 2 Prewitt operators. The resulting image shows the edges of the coins in bold white, the background in black, and the edges of drawings on the coin in faint white.
5. The fifth image displays a 3x3 third edge detection kernel. The resulting image displays the edges of the coins in white, the background in dark gray, and the edges of drawings on the coin in faint black.

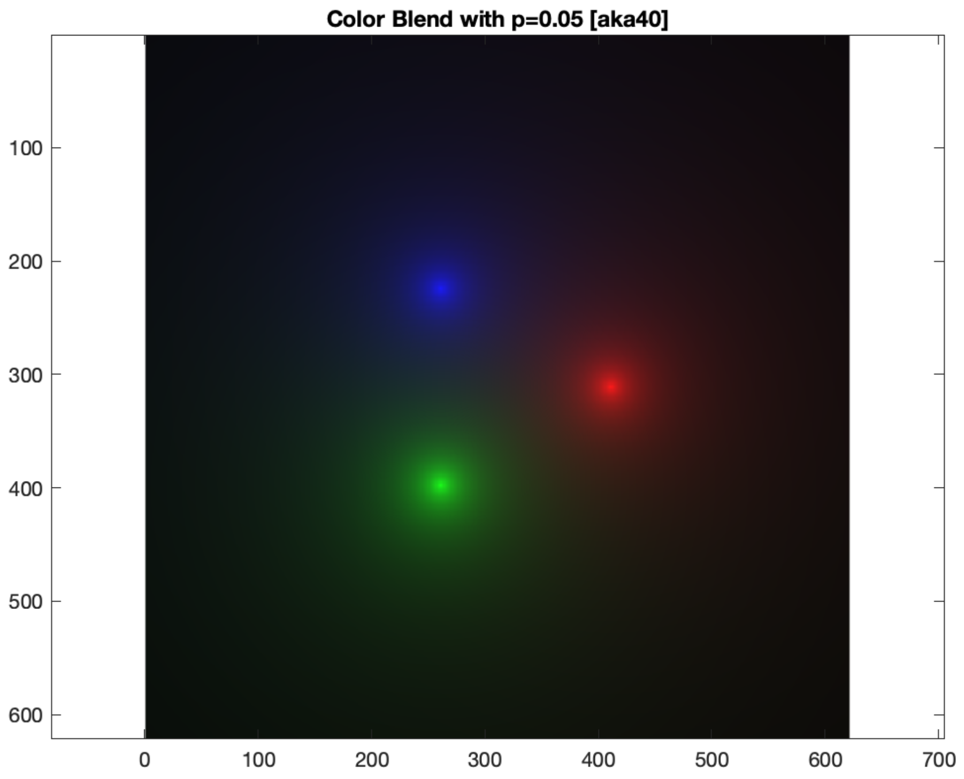
The fourth image displays better edge detection than the fifth due to higher contrast in the colors of the edges and the background. The edges are also more clearly outlined in the fourth image than in the fifth.

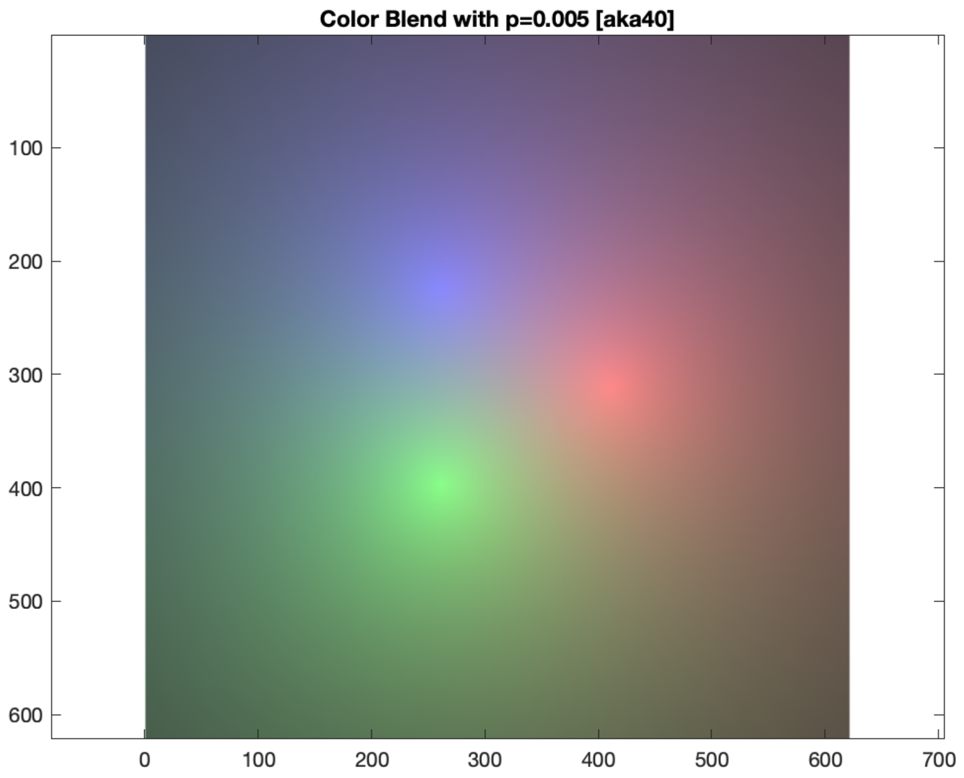
8 Exercise 8

1. The first image shows the original image with no blurs.
2. The second image shows the result of applying a 21x21 box blur to the original image. This blur looks a standard intense blur.
3. The third image shows the result of applying a 5x5 Gaussian Blur. The Guassian blur resulting image is very subtle and less blurry than the 21x21 box blur.
4. The fourth image shows the result of applying a 3x3 Sobel operator blur that detects change in color horizontally (going from left to right). The resulting image shows colored vertical edges of the original image.
5. The fifth image shows the result of applying a 3x3 Sobel operator blur that detects change in color vertically (going from top to bottom). The resulting image shows colored horizontal edges of the original image.
6. The last image displays the square root of the sum of squares of the previous 2 Sobel operators. The resulting image displays the colored horizontal and vertical edges in the original image.

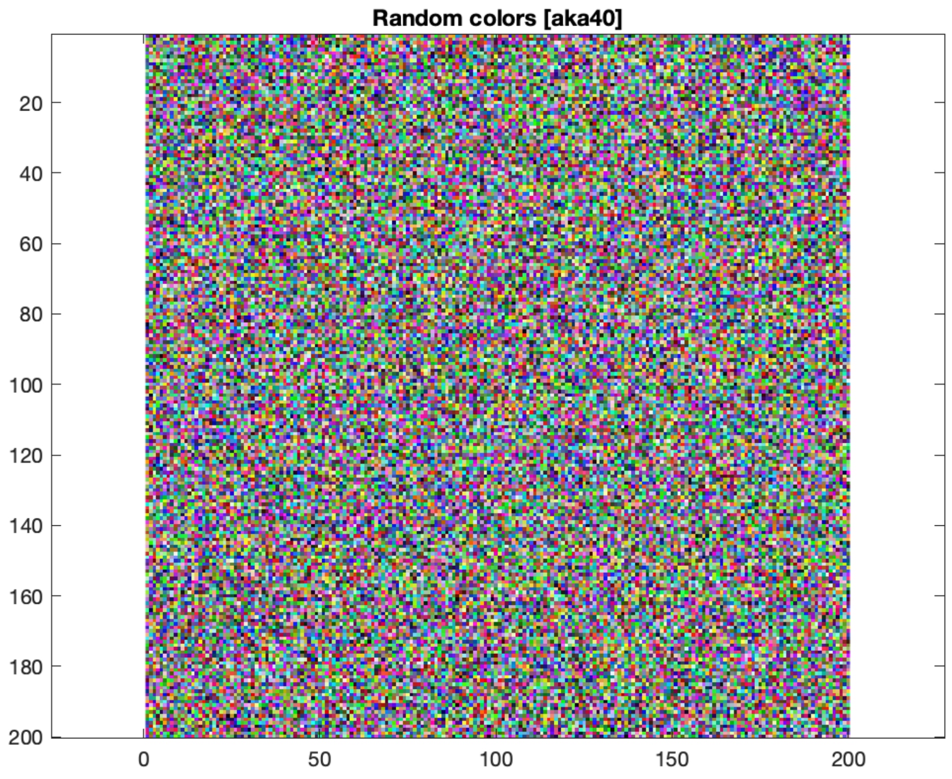
9 Graphs

9.1 Exercise 1

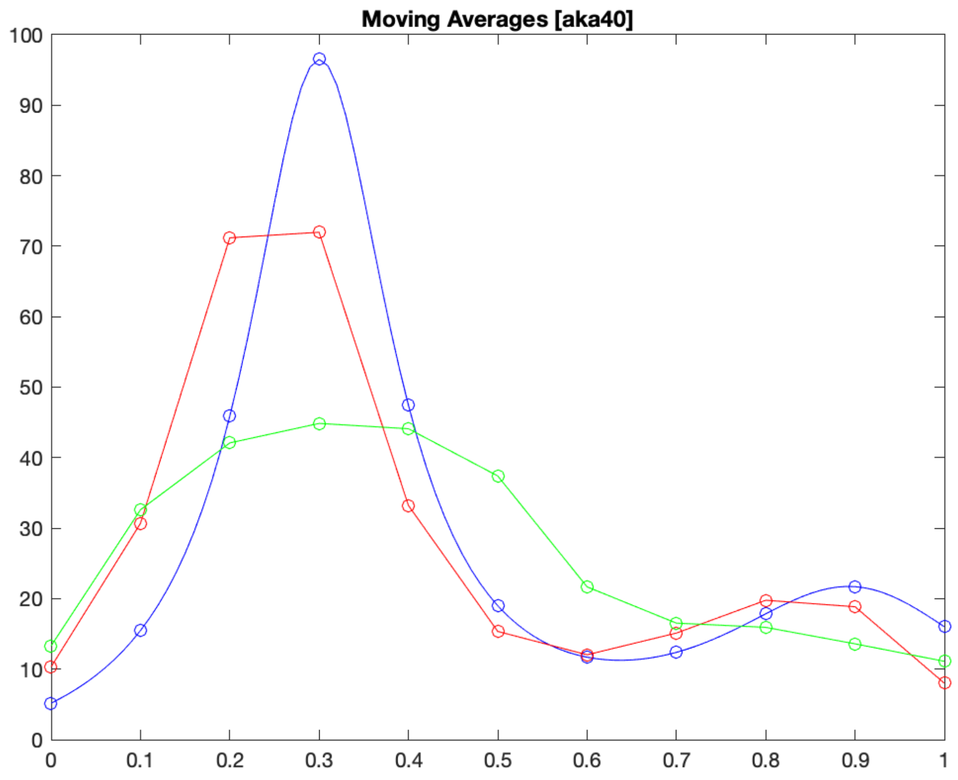




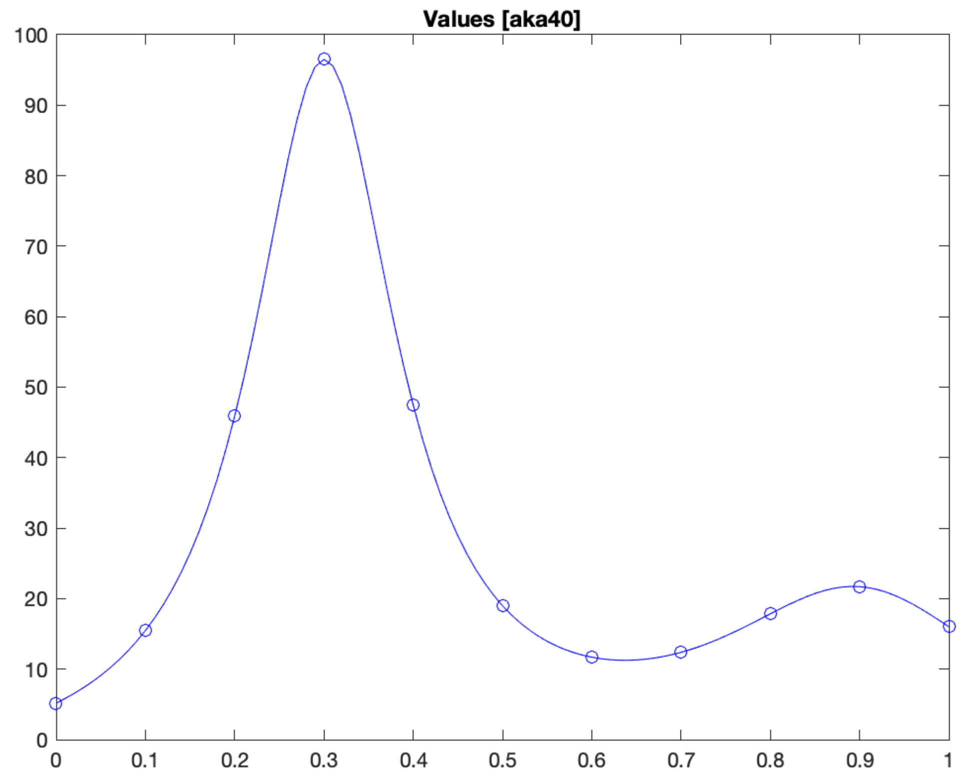
9.2 Exercise 2

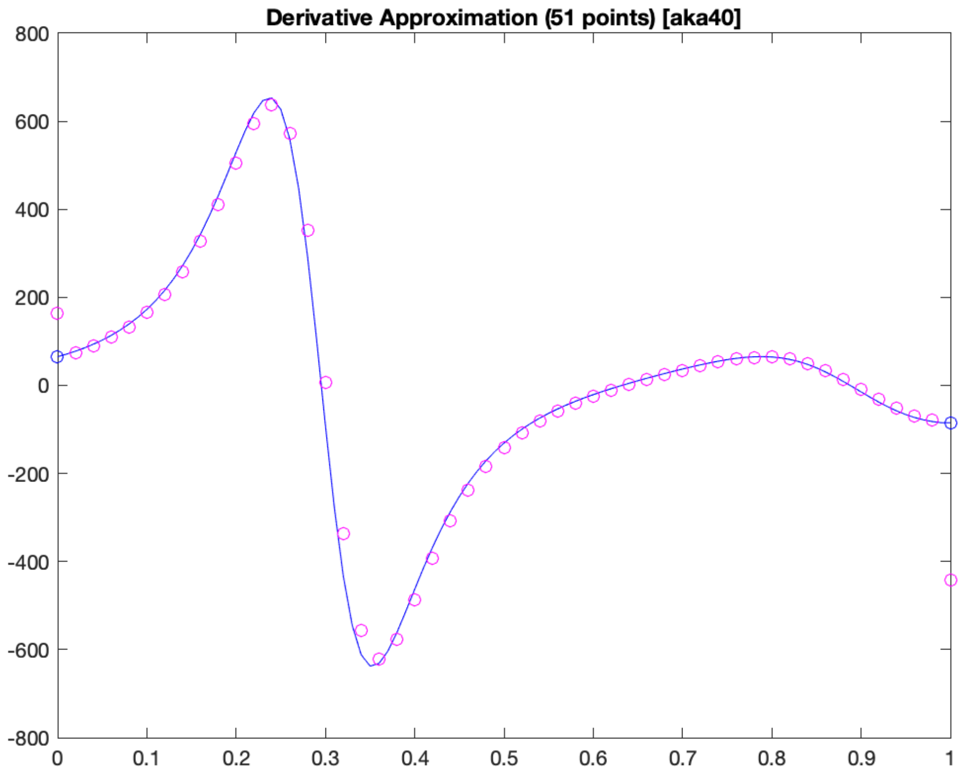


9.3 Exercise 3

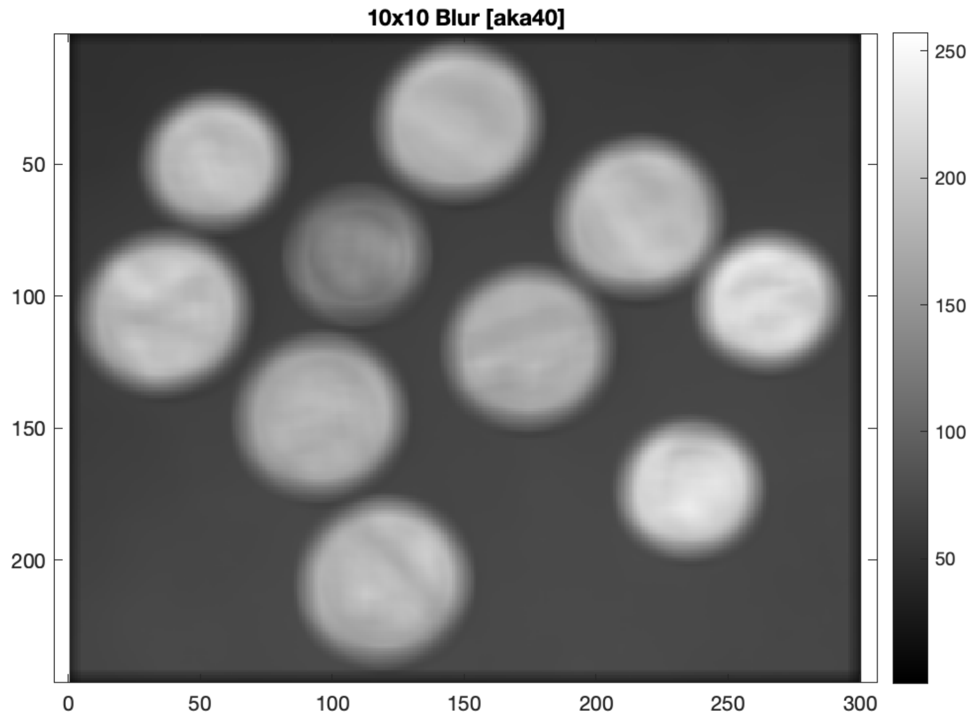


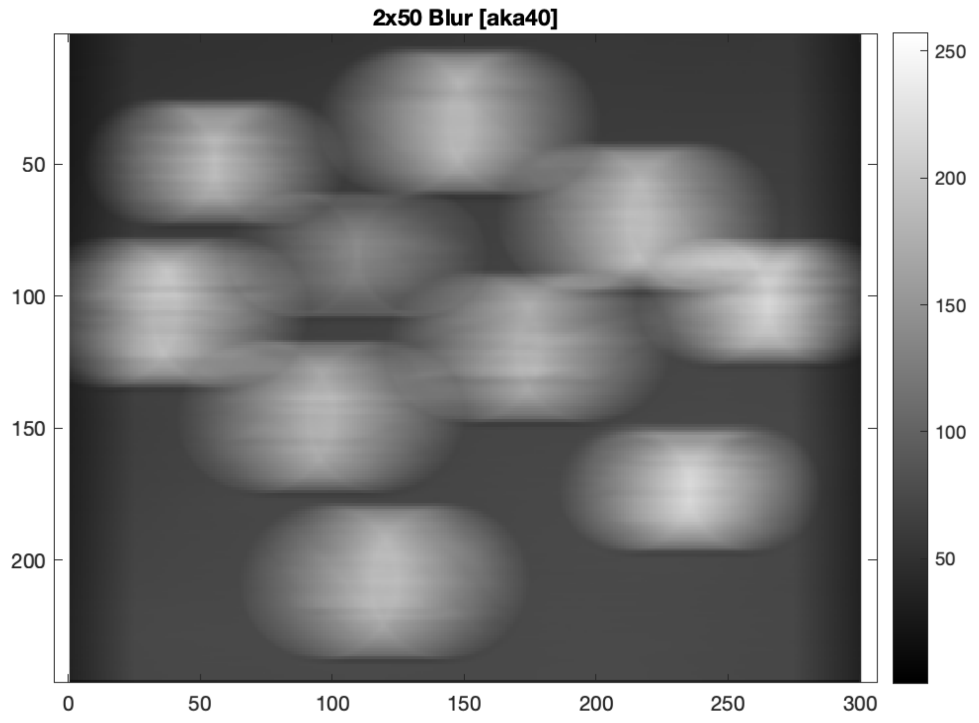
9.4 Exercise 4

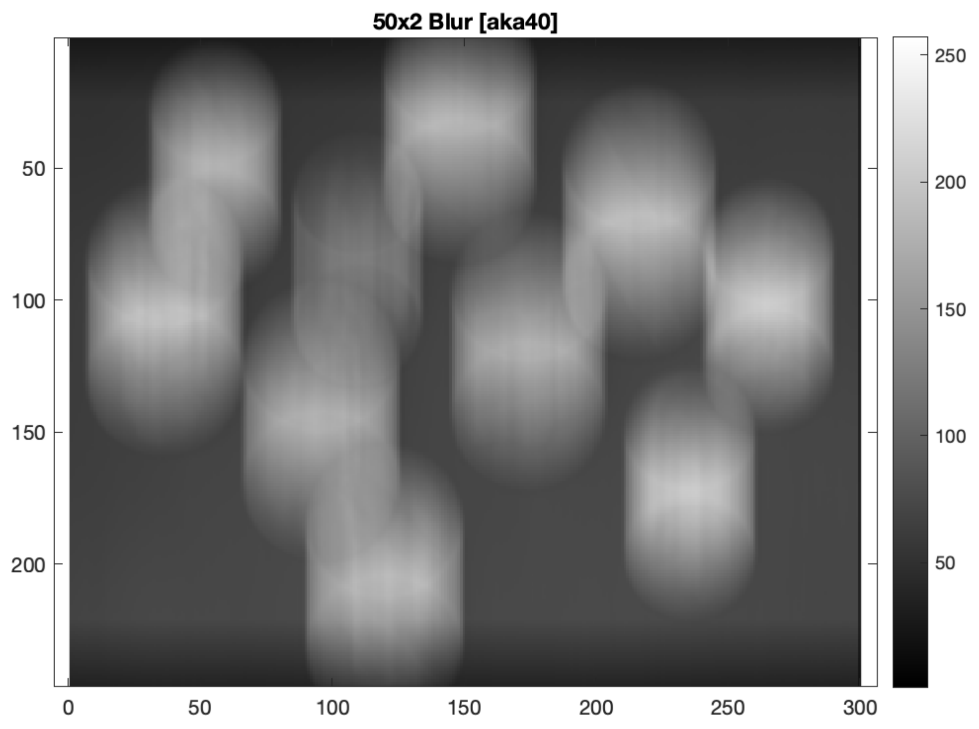




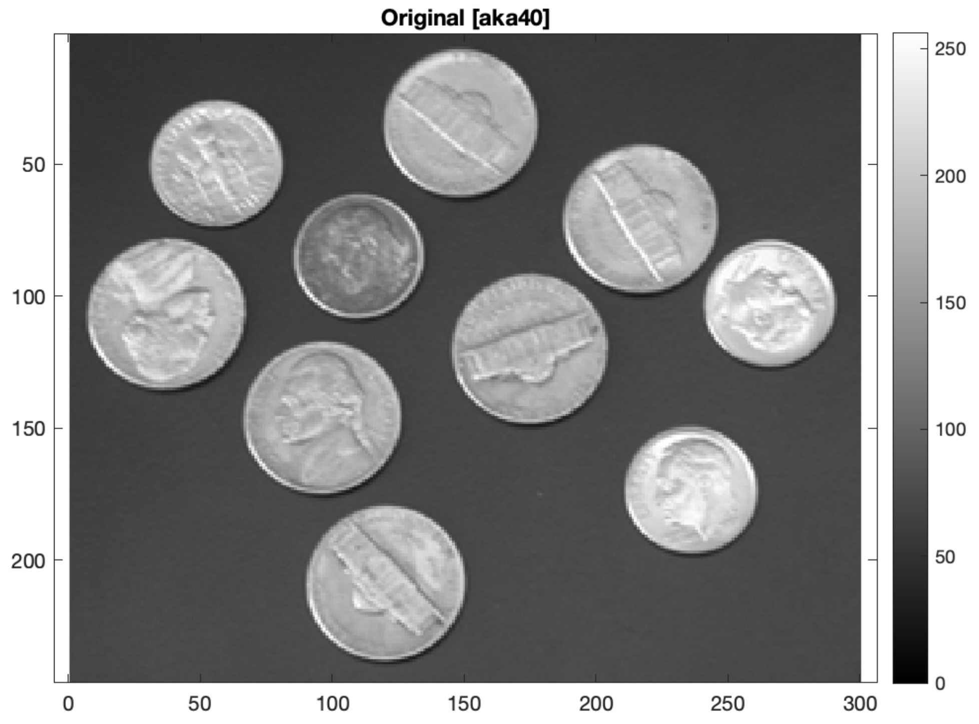
9.5 Exercise 5

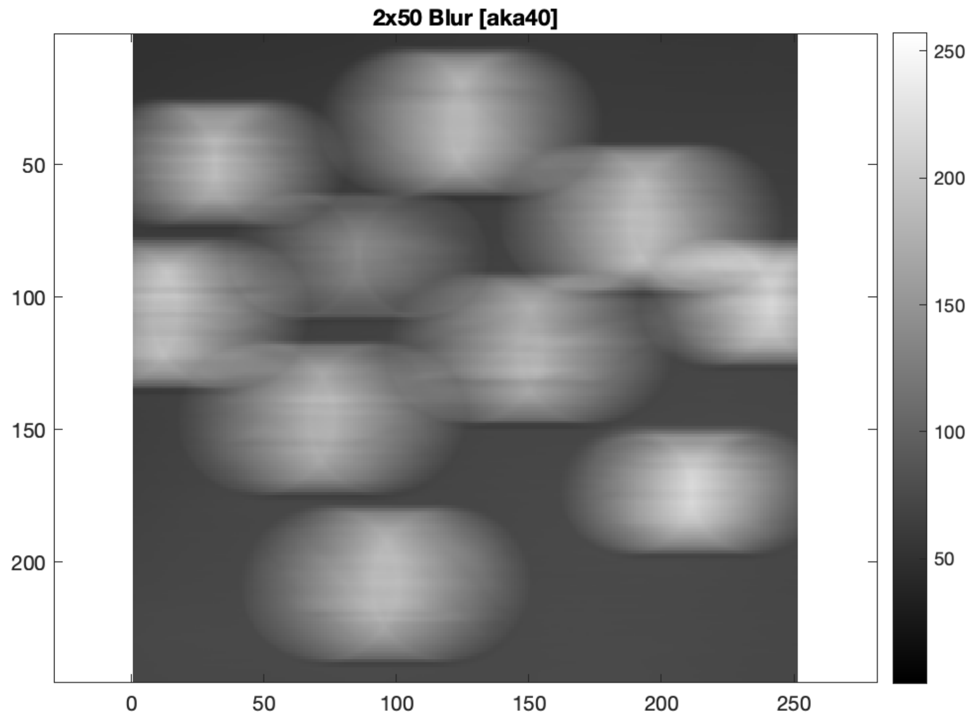


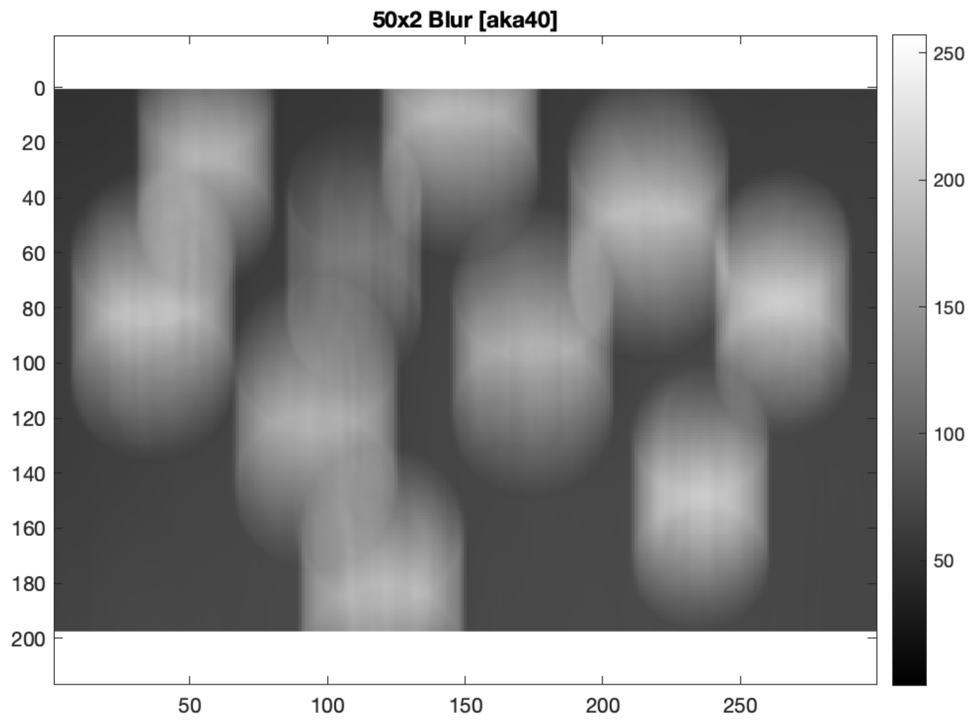



50x2 Blur [aka40]
50
100
150
200
0
50
100
150
200
250
300
250
200
150
100
50

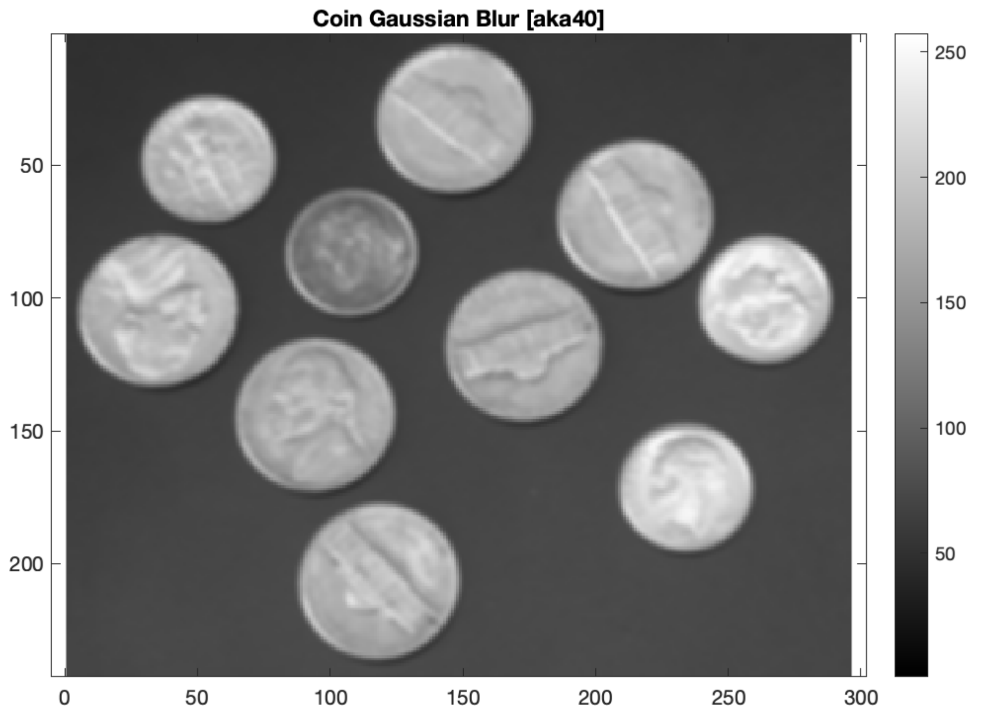
9.6 Exercise 6

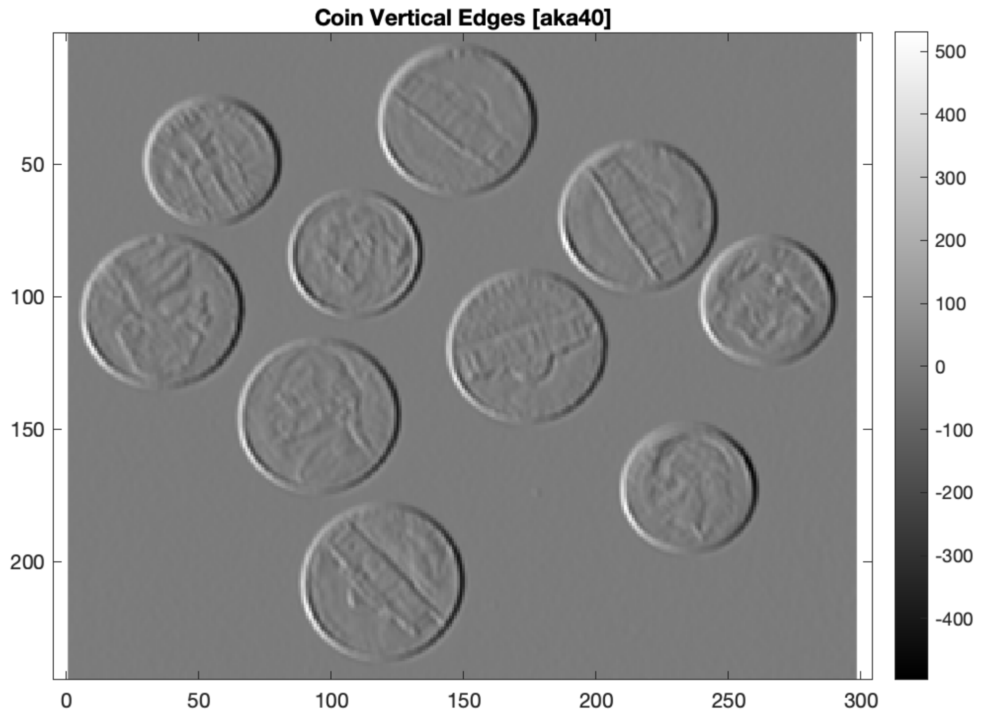


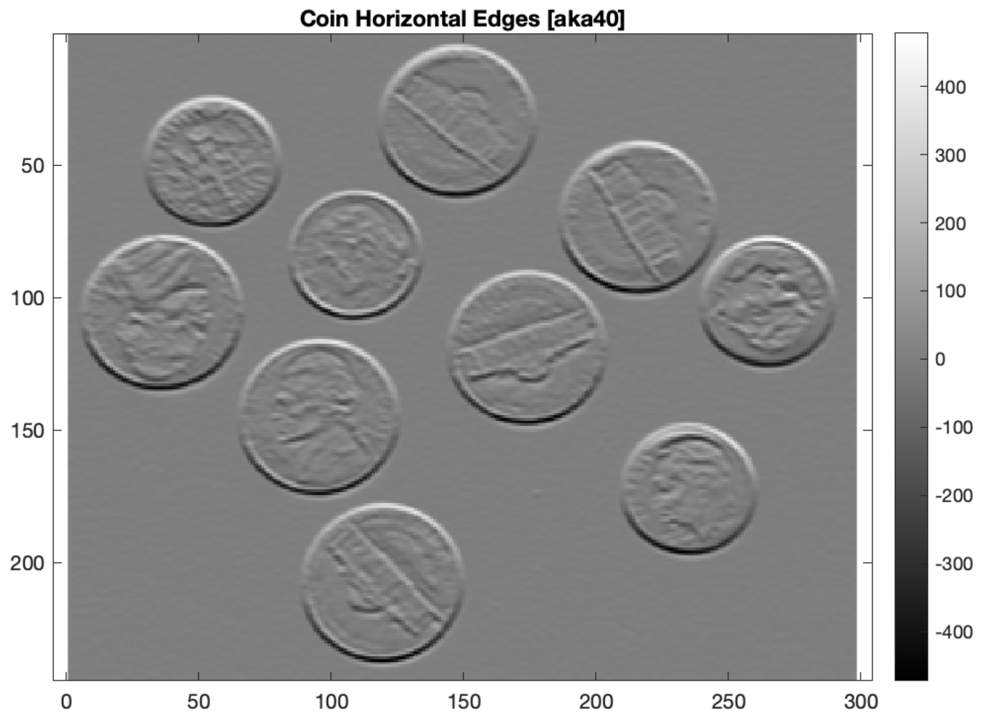


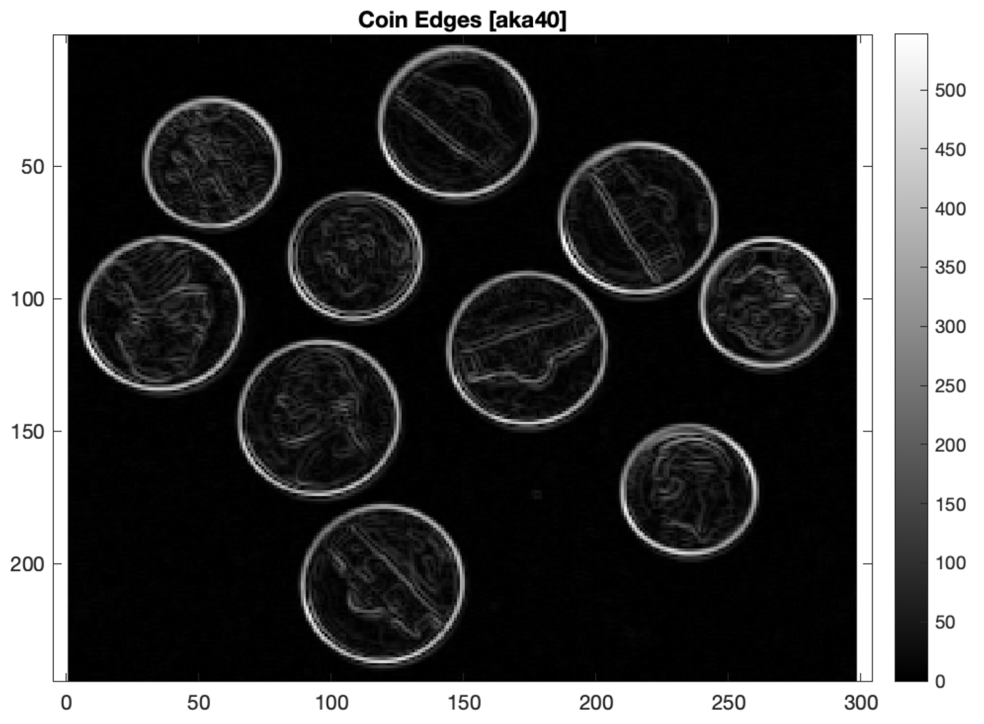

50x2 Blur [aka40]
0
20
40
60
80
100
120
140
160
180
200
50
100
150
200
250
250
200
150
100
50

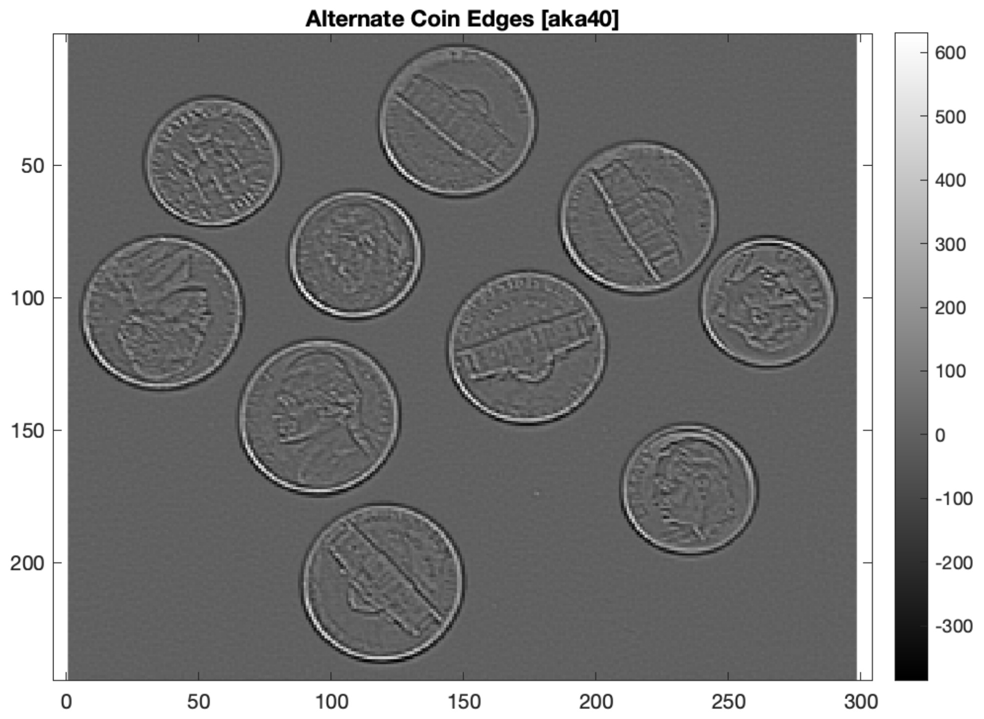
9.7 Exercise 7



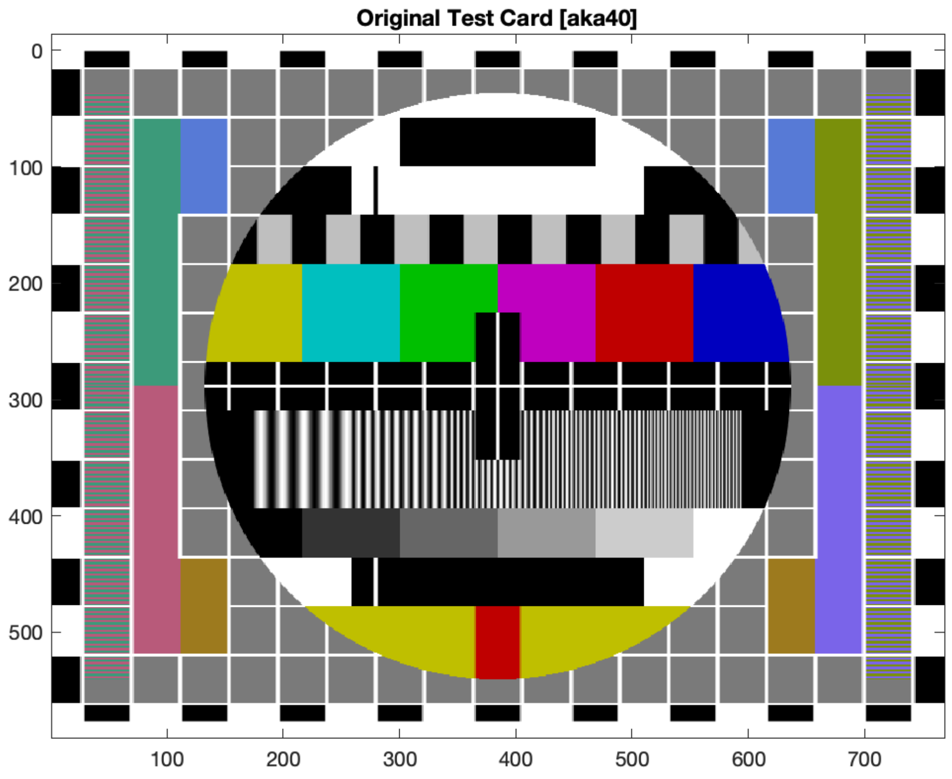


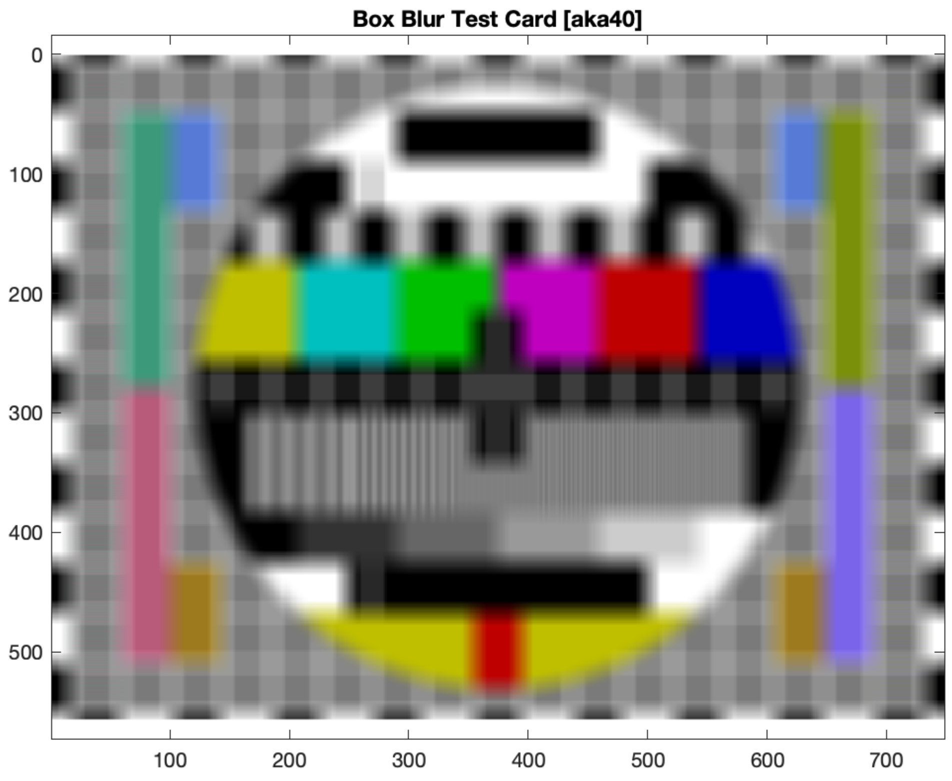

Coin Horizontal Edges [aka40]
50
100
150
200
0
50
100
150
200
250
300
400
300
200
100
0
-100
-200
-300
-400


Coin Edges [aka40]
50
100
150
200
0
50
100
150
200
250
300
500
450
400
350
300
250
200
150
100
50
0

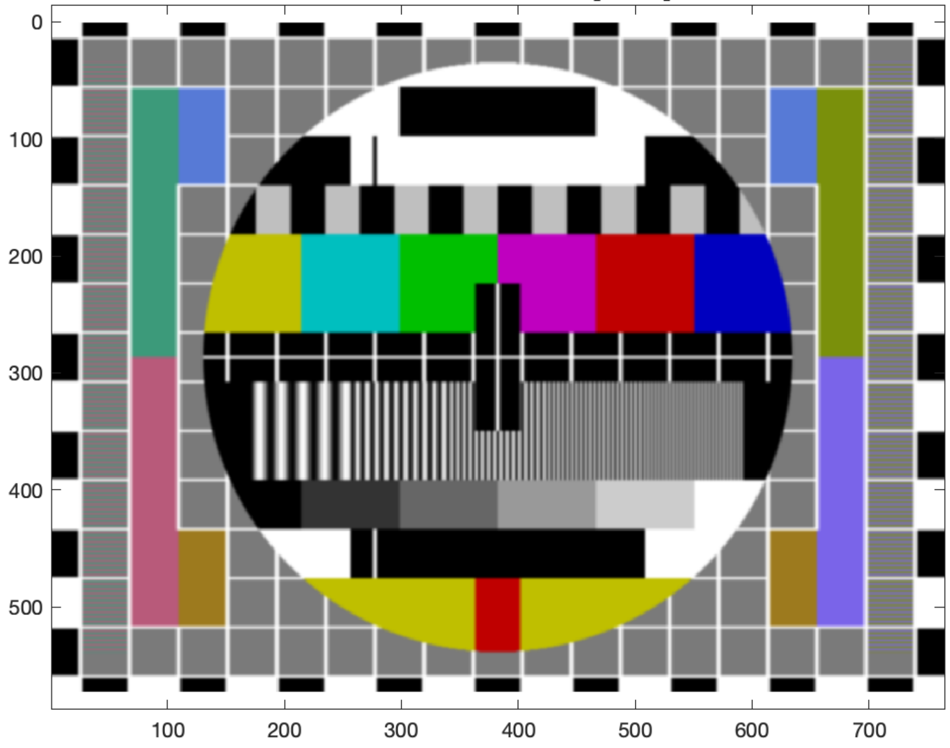

Alternate Coin Edges [aka40]
50
100
150
200
0
50
100
150
200
250
300
600
500
400
300
200
100
0
-100
-200
-300

9.8 Exercise 8

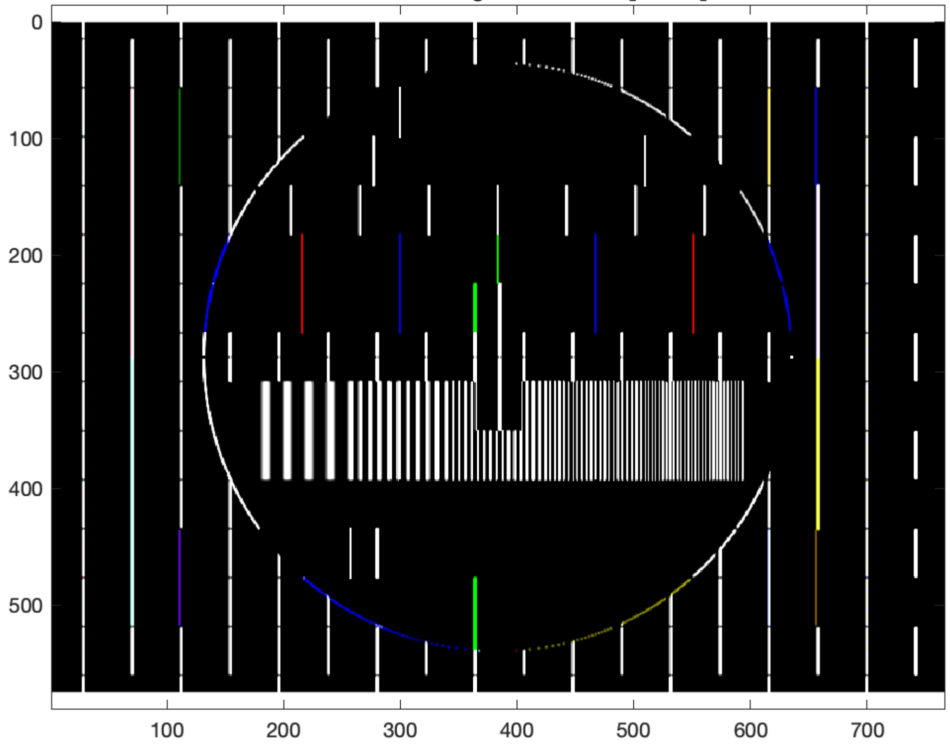


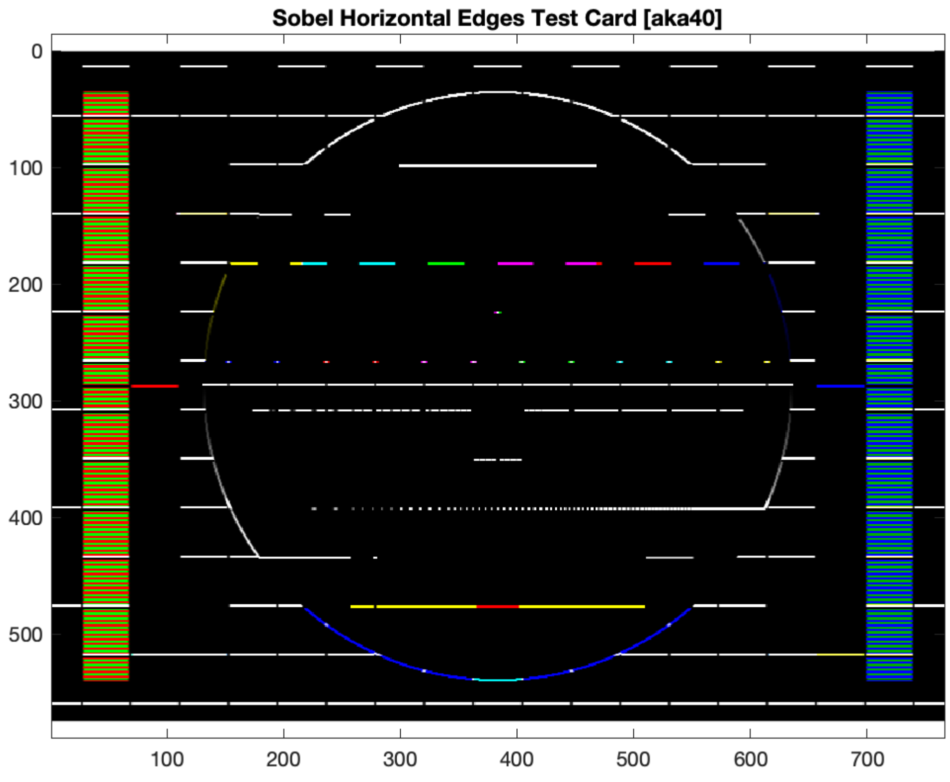


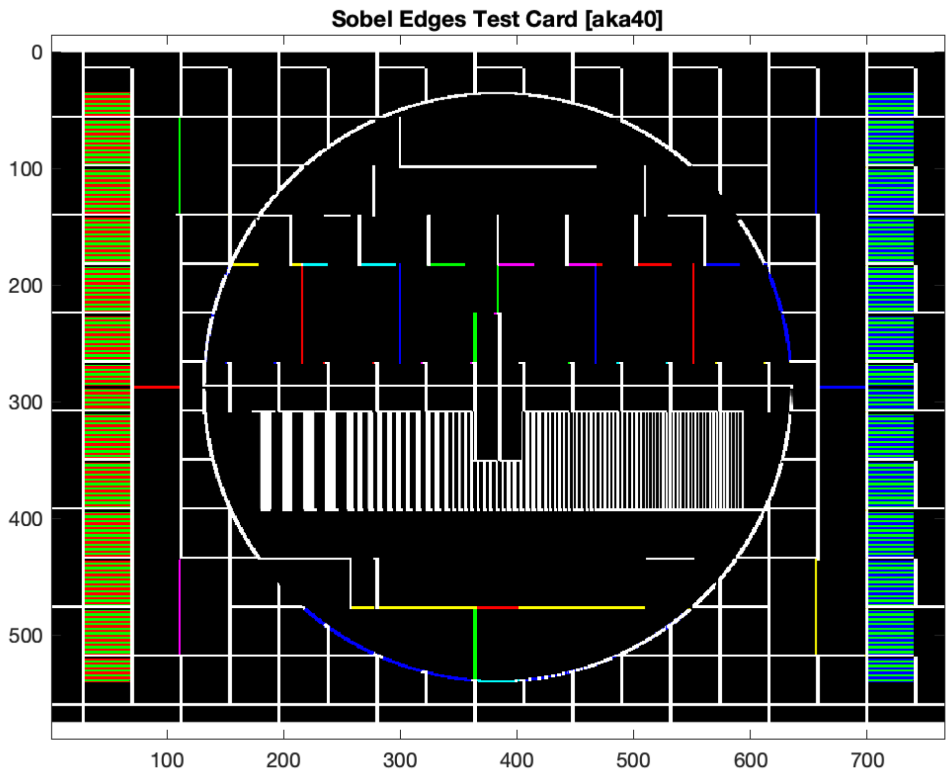
Gaussian Blur Test Card [aka40]



Sobel Vertical Edges Test Card [aka40]




Sobel Horizontal Edges Test Card [aka40]
0
100
200
300
400
500
100
200
300
400
500
600
700


Sobel Edges Test Card [aka40]
0
100
200
300
400
500
100
200
300
400
500
600
700

10 Codes

10.1 Exercise 1

```
clear
rad = 100;
del = 10;
[x, y] = meshgrid((-3*rad-del):(3*rad+del));
[rows, cols] = size(x);
p=0.05;
dist = @(x, y, xc, yc) sqrt((x-xc).^2+(y-yc).^2);
venn_img = zeros(rows, cols, 3);
venn_img(:,:,1) = 1./(1+p*sqrt((x-rad.*cos(0)).^2+(y-rad.*sin(0)).^2));
venn_img(:,:,2) = 1./(1+p*sqrt((x-rad.*cos(2*pi/3)).^2+(y-rad.*sin(2*pi/3)).^2));
venn_img(:,:,3) = 1./(1+p*sqrt((x-rad.*cos(4*pi/3)).^2+(y-rad.*sin(4*pi/3)).^2));
figure(1); clf
image(venn_img)
axis equal
title("Color Blend with p=0.05 [aka40]")

print -dpng IP1_EX1_Plot1.png
p=0.005;
dist = @(x, y, xc, yc) sqrt((x-xc).^2+(y-yc).^2);
venn_img = zeros(rows, cols, 3);
venn_img(:,:,1) = 1./(1+p*sqrt((x-rad.*cos(0)).^2+(y-rad.*sin(0)).^2));
venn_img(:,:,2) = 1./(1+p*sqrt((x-rad.*cos(2*pi/3)).^2+(y-rad.*sin(2*pi/3)).^2));
venn_img(:,:,3) = 1./(1+p*sqrt((x-rad.*cos(4*pi/3)).^2+(y-rad.*sin(4*pi/3)).^2));
figure(2); clf
image(venn_img)
axis equal
title("Color Blend with p=0.005 [aka40]")

print -dpng IP1_EX1_Plot2.png

% The images show 3 circles with different colors at the center (red, green, blue). As th
% circles expand, the color fades away. The greater the p value the faster
% the color fades away.
%
```

10.2 Exercise 2

```
clear
array = zeros(200,200,3);
array(:,:,1)= rand(200,200);
array(:,:,2)= rand(200,200);
array(:,:,3)= rand(200,200);
figure(1);clf
image(array)
axis equal
title("Random colors [aka40]")

print -dpng IP1_EX2_Plot1.png

% The images randomly colored dots. When the code is recompiled. The dots
% appear as if they were scattered in a different order.
```

10.3 Exercise 3

```
clear
tc = linspace(0,1, 101);
xc = humps(tc);
td = linspace(0,1,11);
xd = humps(td);
figure(1),clf
plot (tc, xc, 'b-')
hold on
plot (td, xd, 'bo')

movingAverage2 = conv(xd, ones(1,2)/2, 'same')
plot (td, movingAverage2, 'ro-')

movingAverage5 = conv(xd, ones(1,5)/5, 'same')
plot (td, movingAverage5, 'go-')
title("Moving Averages [aka40]")
hold off

print -dpng IP1_EX3_Plot1.png

% The 5 point moving average results in a smoother plot than the 2 point
% moving average. This is due to the 5 point moving average average larger
% samples of data this is. The 5 point moving average also has a lower peak than
% the 2 point moving average
```

10.4 Exercise 4

```
clear
tc = linspace(0,1, 101);
xc = humps(tc);
deltatc = tc(2) - tc(1)
td = linspace(0,1,11);
xd = humps(td);
deltatd = td(2) - td(1)

figure(1),clf
plot (tc, xc, 'b-')
hold on
plot (td, xd, 'bo')
hold off

title("Values [aka40]")

figure(2); clf
twopointdiff = diff(xc)/deltatc;
twopointdiff(end+1) = twopointdiff(end);
plot(tc,twopointdiff, 'b-');

hold on
plot(tc(1:10:end), twopointdiff(1:10:end), 'bo');

con1 = conv(xd, [1,0,-1]/2/deltatd, 'same')
plot (td, con1, 'mo')
title("Derivative Approximation (11 points) [aka40]")
hold off
print -dpng IP1_EX4_Plot11.png
```

```

figure(3); clf
tc = linspace(0,1, 101);
xc = humps(tc);
deltatc = tc(2) - tc(1)
td = linspace(0,1,51);
xd = humps(td);
deltatd = td(2) - td(1)
twopointdiff = diff(xc)/deltatc;
twopointdiff(end+1) = twopointdiff(end);
plot(tc,twopointdiff, 'b-');

hold on
plot(tc(1:100:end), twopointdiff(1:100:end), 'bo');

con1 = conv(xd, [1,0,-1]/2/deltatd, 'same')
plot (td, con1, 'mo')
title("Derivative Approximation (51 points) [aka40]")

%

print -dpng IP1_EX4_Plot51.png
```

10.5 Exercise 5

```
clear
x = imread('coins.png');
h = ones(10,10)/10^2;
y = conv2(x,h, 'same');

figure(1),clf
image(x);
axis equal;
colormap gray; colorbar
title('Original [aka40]');

figure(2); clf
image(y)
axis equal; colormap gray; colorbar;
title("10x10 Blur [aka40]")
print -dpng IP1_EX5_Plot1.png


figure(4)
h = ones(2,50)/10^2;
y = conv2(x,h, 'same');
image(y)
axis equal; colormap gray; colorbar;
title("2x50 Blur [aka40]")

print -dpng IP1_EX5_Plot2.png
figure(3)
h = ones(50,2)/10^2;
y = conv2(x,h, 'same');
image(y)
axis equal; colormap gray; colorbar;
title("50x2 Blur [aka40]")

print -dpng IP1_EX5_Plot3.png
```

10.6 Exercise 6

```
clear
x = imread('coins.png');
h = ones(10,10)/10^2;
y = conv2(x,h, 'valid');

tc = linspace(0,1, 101);
xc = humps(tc);
deltatc = tc(2) - tc(1)
td = linspace(0,1,11);
xd = humps(td);
deltatd = td(2) - td(1)

figure(1),clf
image(x);
axis equal;
colormap gray; colorbar
title('Original [aka40]');

figure(2); clf
image(y)
axis equal; colormap gray; colorbar;
title("10x10 Blur [aka40]")
print -dpng IP1_EX6_Plot1.png


figure(4)
h = ones(2,50)/10^2;
y = conv2(x,h, 'valid');
image(y)
axis equal; colormap gray; colorbar;
title("2x50 Blur [aka40]")

print -dpng IP1_EX6_Plot2.png
figure(3)
h = ones(50,2)/10^2;
y = conv2(x,h, 'valid');
image(y)
axis equal; colormap gray; colorbar;
title("50x2 Blur [aka40]")

print -dpng IP1_EX6_Plot3.png
```

10.7 Exercise 7

```
clear
x = imread('coins.png');
gb5= [1 4 6 4 1; 4 16 24 16 4; 6 24 36 24 6;  4 16 24 16 4; 1 4 6 4 1]/256;

y1 = conv2(x,gb5, 'valid');
CoinEdgeX=[1 0 -1;1 0 -1; 1 0 -1];
CoinEdgeY=[1 1 1;0 0 0; -1 -1 -1];
figure(1),clf
image(y1);
axis equal;
colormap gray; colorbar
title('Coin Gaussian Blur [aka40]');
print -dpng IP1_EX7_Plot1.png


y2 = conv2(x, CoinEdgeX, 'valid');
figure(2),clf
imagesc(y2);
axis equal;
colormap gray; colorbar
title('Coin Vertical Edges [aka40]');
print -dpng IP1_EX7_Plot2.png

y3 = conv2(x, CoinEdgeY, 'valid');
figure(3),clf
imagesc(y3);
axis equal;
colormap gray; colorbar
title('Coin Horizontal Edges [aka40]');
print -dpng IP1_EX7_Plot3.png

y4= sqrt(y2.^2 + y3.^2);
figure(4),clf
imagesc(y4);
axis equal;
colormap gray; colorbar
title('Coin Edges [aka40]');
print -dpng IP1_EX7_Plot4.png


h = [-1 -1 -1; -1 8 -1; -1 -1 -1];
y5 = conv2(x,h, 'valid');
figure(5),clf
imagesc(y5);
axis equal;
colormap gray; colorbar
title('Alternate Coin Edges [aka40]');
print -dpng IP1_EX7_Plot5.png
```

10.8 Exercise 8

```
clear
img = imread('PM5544_with_non-PAL_signals.png');
figure(1); clf
image(img); axis equal
title('Original Test Card [aka40]')

print -dpng IP1_EX8_Plot0.png


BoxBlur = ones(21,21)/(21^2);
for k=1:3
    y1(:,:,k) = conv2(img(:,:,k), BoxBlur, 'valid');
end
y1=uint8(y1);
figure(2); clf
image(y1);
axis equal; colormap gray
title('Box Blur Test Card [aka40]');
print -dpng IP1_EX8_Plot1.png

gb5= [1 4 6 4 1; 4 16 24 16 4; 6 24 36 24 6;  4 16 24 16 4; 1 4 6 4 1]/256;
for k=1:3
    y2(:,:,k) = conv2(img(:,:,k), gb5, 'valid');
end
y2=uint8(y2);
figure(3); clf
image(y2);
axis equal; colormap gray
title('Gaussian Blur Test Card [aka40]');
print -dpng IP1_EX8_Plot2.png

SobelX=[-1 0 1; -2 0 2; -1 0 1];
for k=1:3
    y3(:,:,k) = conv2(img(:,:,k), SobelX, 'valid');
end
y3=uint8(y3);
figure(4); clf
image(y3);
axis equal; colormap gray
title('Sobel Vertical Edges Test Card [aka40]');
print -dpng IP1_EX8_Plot3.png

SobelY=[1 2 1; 0 0 0; -1 -2 -1];
for k=1:3
    y4(:,:,k) = conv2(img(:,:,k), SobelY, 'valid');
end

y4=uint8(y4);
figure(5); clf
image(y4);
axis equal; colormap gray
title('Sobel Horizontal Edges Test Card [aka40]');
print -dpng IP1_EX8_Plot4.png


```

```
Sobel = sqrt(SobelY.^2 + SobelX.^2);
for k=1:3
    y5(:,:,k) = (y4(:,:,k).^2 + y3(:,:,k).^2);
end

y5=uint8(y5);
figure(6); clf
imagesc(y5);
axis equal; colormap gray
title('Sobel Edges Test Card [aka40]');
print -dpng IP1_EX8_Plot5.png
```