Table of Contents

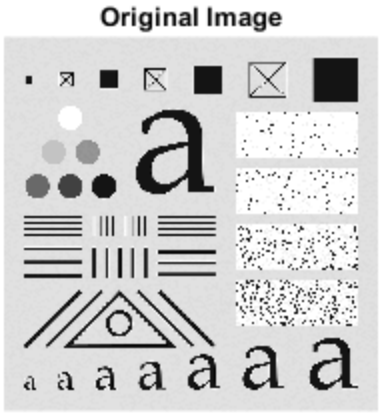

```
%Assignment 1 by Nikeet Pandit
%GS/MATH 6920 Harmonic Analysis and Image Processing
% --- Question List --- %
% 7a) Read image "testpattern1024.tif" and low-pass filter using Gaussian
%     Kernel large enough to blur "a"... while other letters are still
 readable
%
% 7b) Read image "testpattern1024.tif". Low-pass it, so when thresholded,
%     the filtered image only contains square in top right
%
% 7c) Read image "chekerboard1024-shaded.tif" and filter checkerboard to
%     superimposed shading pattern
%
% 8a) Read "blurry-moon.tif" and sharpen using unsharp masking. Blur with
%     gaussian kernel.
%
% 8b) Improve sharpeness of 8a) using high-boost filtering
%
% 9a) Read "blurry-moon.tif" and sharpen with Laplacian kernel
%
% 9b) Read "checkerboard1024-shaded.tif" and Sobel filter with two kernels
%
% --- Question List -- %

addpath(strcat(pwd, '\images\')); % adding image path
```

Question 7

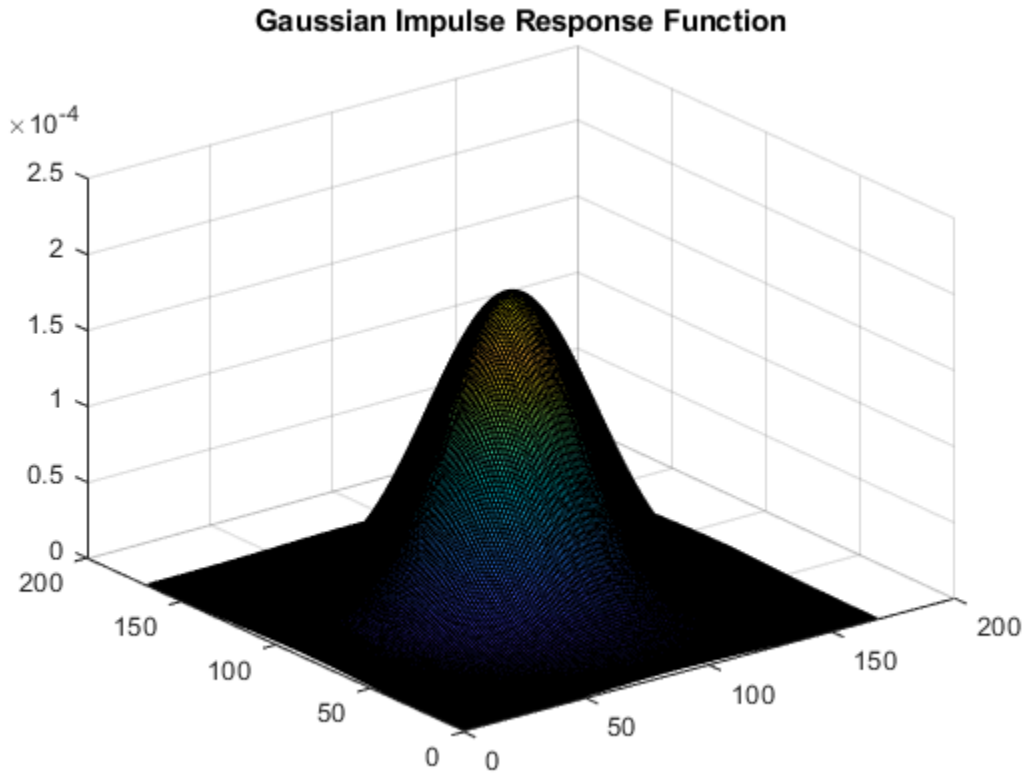
```
%----- Loading Image and Setting Variables

clearvars
File = "testpattern1024.tif"; Im = imread(File); close all
```

7a)

```
%----- %Setting Kernel Parameters

%Sizek generates kernel size from variance
%Ensures Odd Kernel Size + Size is 6x Variance
%Kernel on border contributes minimally ... nothing to gain from larger kernel

kernel_var = 28;
sizeK = kernel_var*6 - mod(kernel_var*6,2) + 1;

%----- Filtering Image
h = fspecial('gaussian',sizeK,kernel_var); %Constructs Impulse Response
Im_f = conv2(Im,h,'same'); %Convolves Impulse Response with Image
                           %Same output size as original image
%----- Plotting Image
figure;
subplot(1,2,1);
imshow(mat2gray(Im));
title('Original Image');
subplot(1,2,2);
imshow(mat2gray(Im_f)); title('Filtered Image');
xlabel(['Kernel Size is: ',sprintf('%d',sizeK),'. Kernel Var is: ',
 sprintf('%d',kernel_var)]);
figure;
surf(h);
title('Gaussian Impulse Response Function');
```

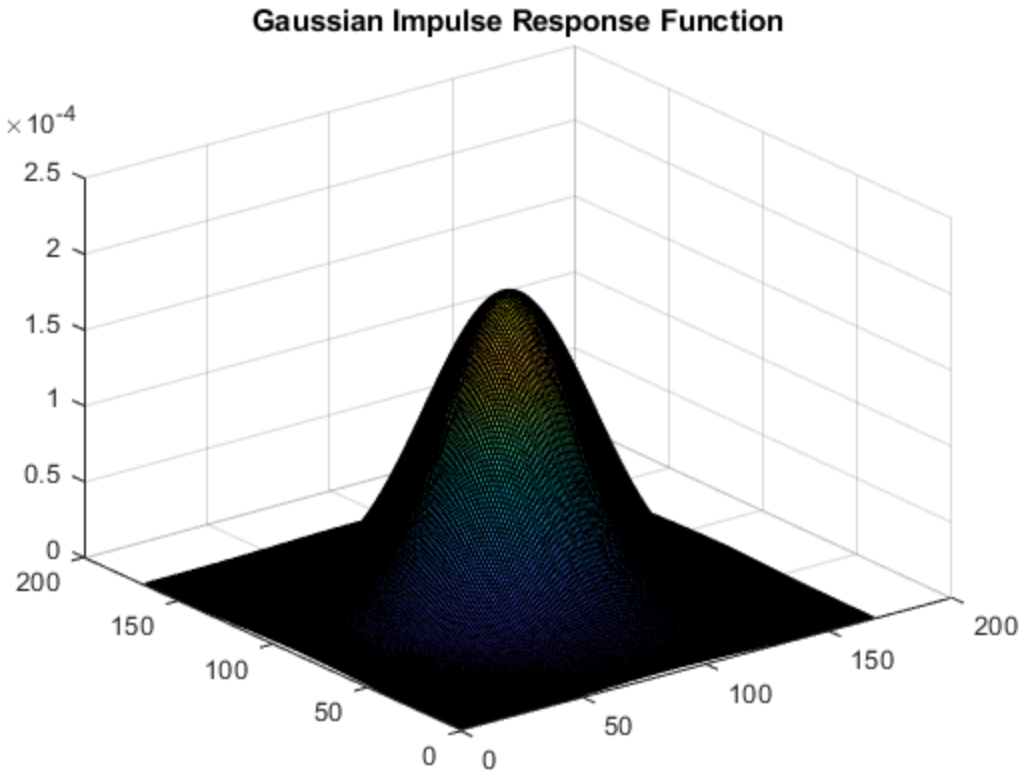


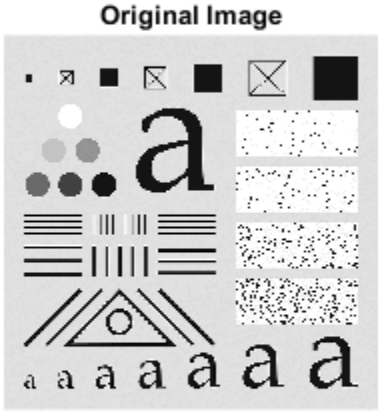
Kernel Size is: 169. Kernel Var is: 28



7b)

```
%----- Loading Image and Setting Variables
clearvars

File = "testpattern1024.tif";
Im = imread(File); ImInfo =  imfinfo(File); [rows,col] = size(Im);

%--- Taking Image Negative
L = 2^ImInfo.BitDepth-1;
Im_Vector = reshape(Im,[1,numel(Im)]); %Reshapes Image into Vector
s = (L - 1) - Im_Vector; %Image Transformation (Negative)
Im_neg = mat2gray(reshape(s,[rows,col])); %Reshaping Vector to Mat
kernel_var = 30;

%---- Kernel Size
x = kernel_var*6;
sizeK = x-mod(x,2)+1;
%----- Filtering
h = fspecial('gaussian',sizeK,kernel_var);
Im_f = conv2(Im_neg,h,'same');

%-----Thresholding
Threshold = 0.64;
Im_Threshold = (Im_f > Threshold).*Im_f; %Thresholding and Masking by Logical
 Array Vector

%-----Taking Inverse Negative
Im_Final = mat2gray((L - 1) - Im_Threshold);

%----Plotting
figure
subplot(1,2,1); imshow(Im); title('Original Image');
subplot(1,2,2); imshow(Im_Final); title('Square On Top Right Only');
```

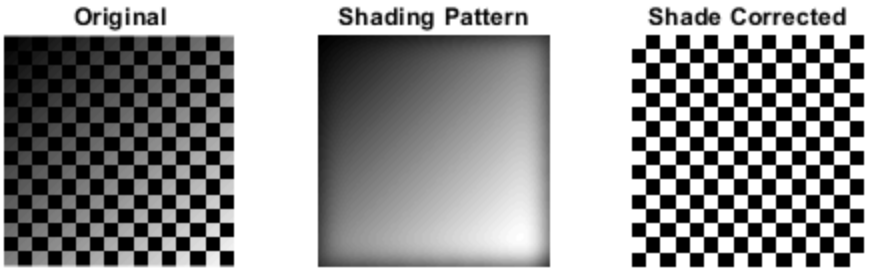

Gaussian Impulse Response Function
×10^-4
2.5
2
1.5
1
0.5
0
200
150
100
50
0
0
50
100
150
200


Original Image


Square On Top Right Only

7c)

```
%----- Loading Image and Setting Variables
clearvars

File = "checkerboard1024-shaded.tif";
Im = imread(File); ImInfo =  imfinfo(File); [rows,col] = size(Im);

%---- Kernel Size
kernel_var = 58; %Variance Equal to Size of Square
x = kernel_var*6;
sizeK = x-mod(x,2)+1;

%----- Filtering
figure
h = fspecial('gaussian',sizeK,kernel_var);
Im_f = conv2(Im,h,'same'); %Low Pass Gradient

%----- Shading Correction
Im_Shade_Corr = mat2gray(Im_f)./mat2gray(Im);

%----- Plotting
figure; subplot(1,3,1); imshow(Im); title('Original');
subplot(1,3,2); imshow(mat2gray(Im_f)); title('Shading Pattern');
subplot(1,3,3); imshow(mat2gray(Im_Shade_Corr)); title('Shade Corrected');
```



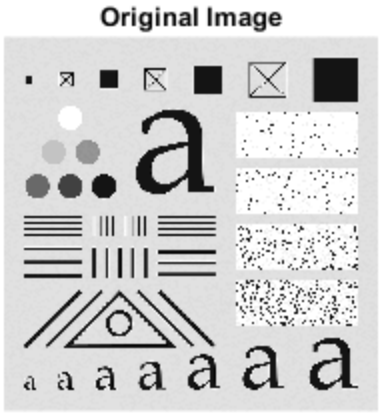
Question 8

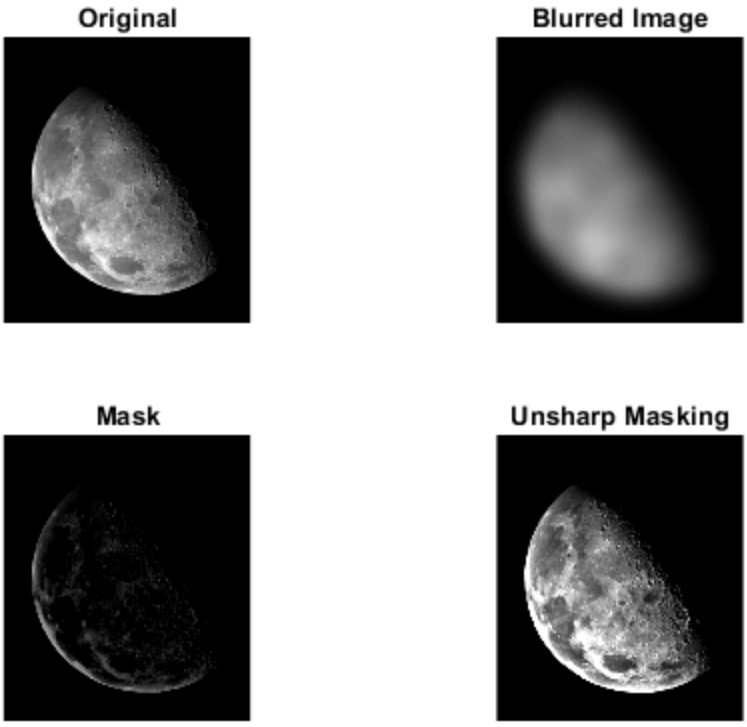
```
%----- Loading Image and Setting Variables
clearvars
File = "blurry-moon.tif"; Im = imread(File);
```

8a)

```
%---- Kernel Size
kernel_var = 30; %High smoothing to reveal lots of high freq components
x = kernel_var*6;
sizeK = x-mod(x,2)+1;

%----- Filtering (Blur)
figure
h = fspecial('gaussian',sizeK,kernel_var);
Im_f = conv2(Im,h,'same'); %Low Pass Gradient

%----- Get Mask
Im_Mask = double(Im) - double(Im_f);

%---- Add Mask (Unsharp Marking)
k = 1;
Im_Unsharp = double(Im) + k*(Im_Mask);
```

```
%---- Plotting
subplot(2,2,1); imshow(uint8(Im)); title('Original');
subplot(2,2,2); imshow(uint8(Im_f)); title('Blurred Image');
subplot(2,2,3); imshow(uint8(Im_Mask)); title('Mask');
subplot(2,2,4); imshow(uint8(Im_Unsharp)); title('Unsharp Masking');
```

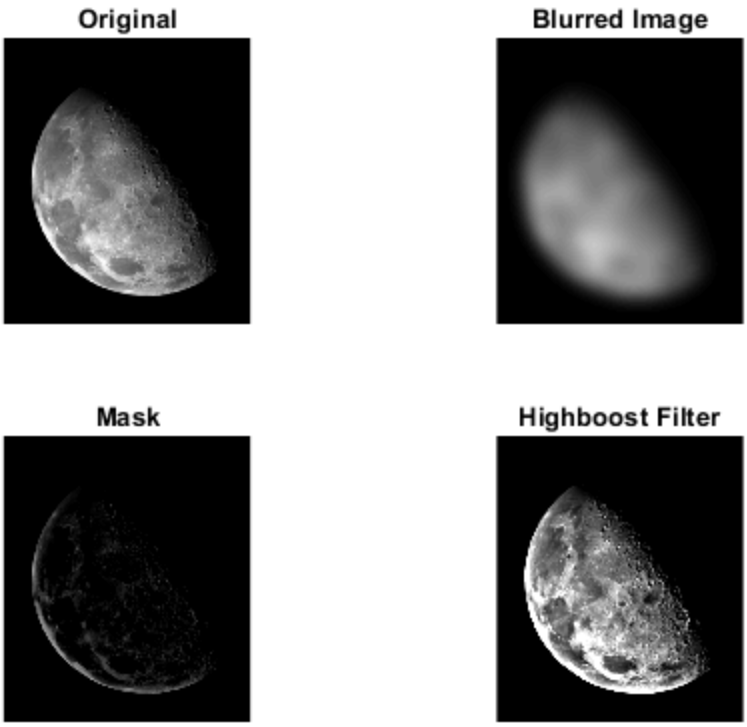




8b)

```
%---- Kernel Size
kernel_var = 25; %High smoothing to reveal lots of high freq components
x = kernel_var*6;
sizeK = x-mod(x,2)+1;

%----- Filtering (Blur)
h = fspecial('gaussian',sizeK,kernel_var);
Im_f = conv2(Im,h,'same'); %Low Pass Gradient

%----- Get Mask
Im_Mask = double(Im) - double(Im_f);

%---- Add Mask (Unsharp Marking)
k = 1.5;
Im_HighBoost = double(Im) + k*(Im_Mask);

%---- Plotting
figure;
subplot(2,2,1); imshow(uint8(Im)); title('Original');
subplot(2,2,2); imshow(uint8(Im_f)); title('Blurred Image');
subplot(2,2,3); imshow(uint8(Im_Mask)); title('Mask');
subplot(2,2,4); imshow(uint8(Im_HighBoost)); title('Highboost Filter');
figure;
```

```
subplot(1,2,1); imshow(uint8(Im_Unsharp)); title('Unsharp Masking');
subplot(1,2,2); imshow(uint8(Im_HighBoost)); title('Highbost Filter');
```

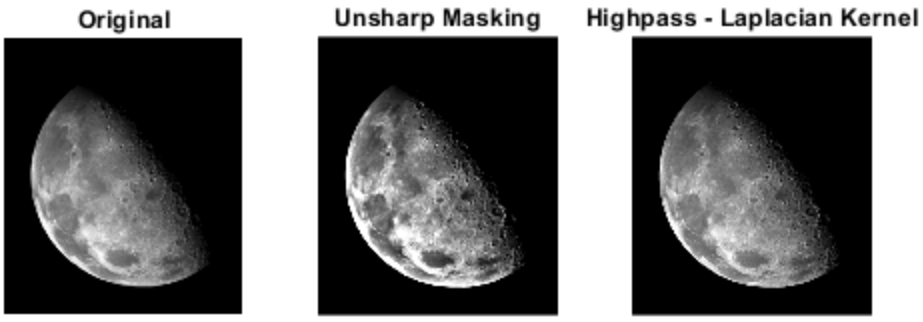


Question 9

```
%----- Loading Image and Setting Variables
clearvars -except Im_Unsharp Im
```

9a)

```
%----- Lapacian Kernel Construct
h = [0 -1 0; -1 4 -1; 0 -1 0];
%----- Filtering
Im_f = double(Im) + double(conv2(Im,h,'same')); %Image Sharpening w/ Lapacian
 Kernel
subplot(1,3,1); imshow(uint8(Im)); title('Original');
subplot(1,3,2); imshow(uint8(Im_Unsharp)); title('Unsharp Masking');
subplot(1,3,3); imshow(uint8(Im_f)); title('Highpass - Laplacian Kernel');
```



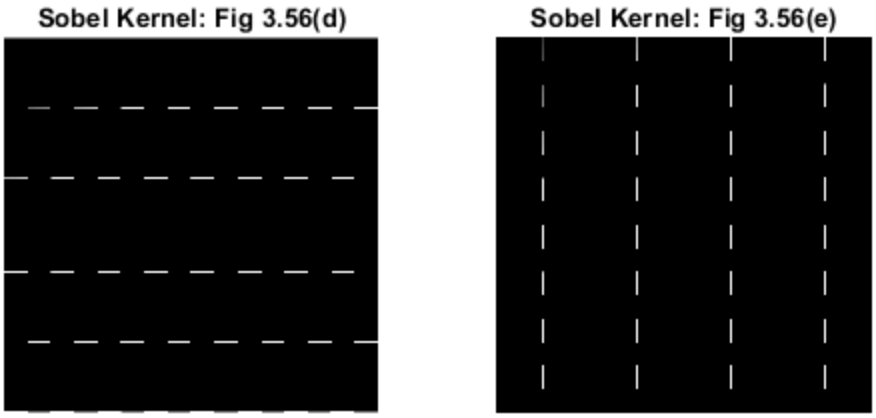
9a) Written Response

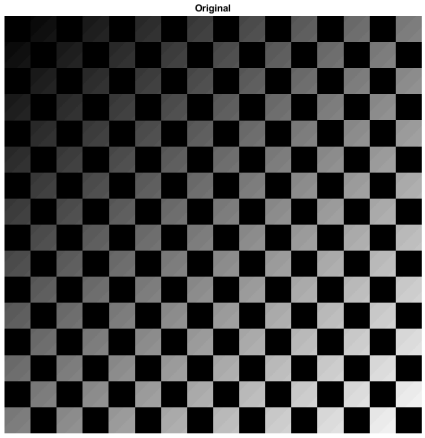
The image with unsharp masking had the effect that contrast was boosted so high that the edge enhancement effect on the image made it visually look unappealing and clearly processed. For a casual observer of the image, this type of filtering effect may not be desireable. However, if for some scientific objective it is essential to have edges that are extremely pronounced and contrasted, then this effect may be desirable. On the other hand, the highpass - laplacian kernel provided edge enhancement to the blurred image, but found a balance between not enhancing the edges too much (like prior) and the original image (which was blurry) that was visually very appealing without looking like the image had been processed, another desirable charactertistic for a casual observer of the image.

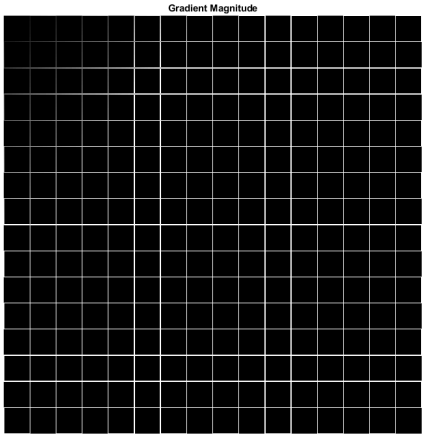
9b)

```
%----- Loading Image and Setting Variables
clearvars
File = "checkerboard1024-shaded.tif"; Im = imread(File);

%---- Construct Kernel
h1 = [-1 -2 -1; 0 0 0; 1 2 1]; %Sobel Kernel Fig. 3.56(d)
h2 = [-1 0 1; -2 0 2; -1 0 1]; %Sobel Kernel Fig. 3.56(e)

%---- Filtering
Im_f1 = conv2(Im,h1,'same');
Im_f2 = conv2(Im,h2,'same');
Im_f3 = sqrt(Im_f1.^2+Im_f2.^2); %Calculating Gradient Magnitude
```

```
%---- Plotting
figure;
subplot(1,2,1); imshow(uint8(Im_f1)); title('Sobel Kernel: Fig 3.56(d)');
subplot(1,2,2); imshow(uint8(Im_f2)); title('Sobel Kernel: Fig 3.56(e)');
figure;
subplot(1,2,1); imshow(uint8(Im)); title('Original');
subplot(1,2,2); imshow(uint8(Im_f3)); title('Gradient Magnitude');
set(gcf, 'Position', get(0, 'Screensize'));
```







9b) Written Response

Sobel Kernel Fig 3.56(d) picks up edge only in x direction. While Kernel Fig 3.56(e) picks up edge only in y direction. These correspond to the row and column directions, respectively. The sobel operators are derivative operators so they magnify high frequency changes in intensity. In the checkboard images, this corresponds to the edge changes traced out by the boxes on the board. When plotting the magnitude of the gradient operator, it may be seen that all the edges (i.e., boxes traced out by the game) are detected.

Published with MATLAB® R2022a