Table of Contents

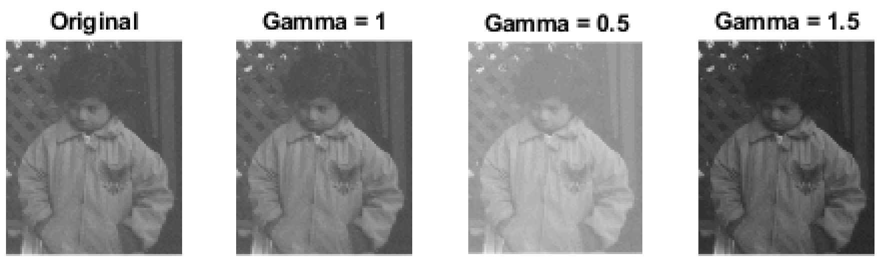

```
%ECES435 Assignment 1 - By Wanyu Li and John Seitz
close all; clear all; clc;
```

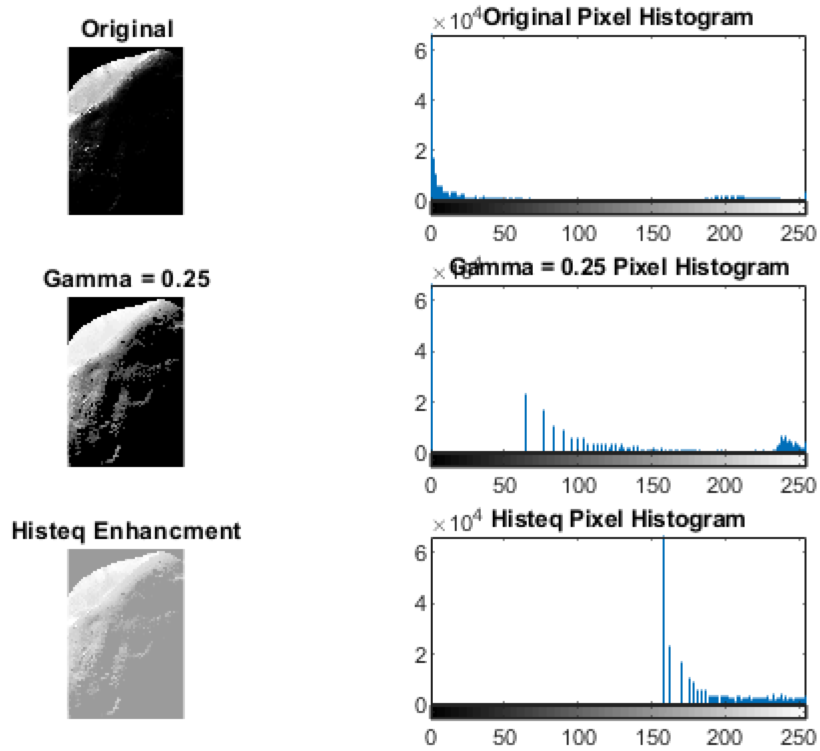
Part 1 - Contrast Enhancment on Images

```
%Gamma correction of "pout" image
pout = imread('pout.tif'); %Read in the grayscale image file as unit8
type gammacorrection

figure(1);
subplot(1,4,1) %plot the original image
imshow(pout)
title 'Original'

gamma = 1; %user specified gamma value
Newimg = gammacorrection(gamma,pout); %gamma correction function
newpout = uint8(Newimg); %change corrected image to unit8 type
subplot(1,4,2)
imshow(newpout)
title 'Gamma = 1'

gamma = .5; %user specified gamma value
Newimg = gammacorrection(gamma,pout); %gamma correction function
newpout = uint8(Newimg); %change corrected image to unit8 type
subplot(1,4,3)
imshow(newpout)
title 'Gamma = 0.5'

gamma = 1.5; %user specified gamma value
Newimg = gammacorrection(gamma,pout); %gamma correction function
newpout = uint8(Newimg); %change corrected image to unit8 type
subplot(1,4,4)
imshow(newpout)
title 'Gamma = 1.5'

%Gamma Correction and Histeq On 'moonPhobos' Image

moonPhobos = imread('MoonPhobos.tif');
figure(2);

subplot(3,2,1);
imshow(moonPhobos)
```

```
title 'Original';
subplot(3,2,2);
imhist(moonPhobos);
title 'Original Pixel Histogram';

%Gamma correction of moonPhobos image

gamma = 0.25; %user specified gamma value
Newimg = gammacorrection(gamma,moonPhobos); % perform gamma correction
 function
newmoonPhobos = uint8(Newimg); %change corrected image to unit8 type
subplot(3,2,3);
imshow(newmoonPhobos)
title 'Gamma = 0.25'
subplot(3,2,4);
imhist(newmoonPhobos);
title 'Gamma = 0.25 Pixel Histogram'

%Histogram Equalization contrast enhancement
subplot(3,2,5);
newmoonPhobos = histeq(moonPhobos,256);
imshow(newmoonPhobos)
title 'Histeq Enhancment'
subplot(3,2,6);
imhist(newmoonPhobos);
title 'Histeq Pixel Histogram'

%Comments and Discussion
%Below are four images of "Pout", the affect of the gamma correction
%contrast enhancment can be seen. When gamma=1, the image is identical
 to
%the original due to the equation canceling out. When gamma<1, the
 image
%becomes lighter and therefore the pixel value histogram is skewed
 towards
%255. When gamma>1, the opposite happens: the image becomes darker,
 and
%therefore the pixel value histogram is skewed towards 0. After
 applying
%the gamma correction contrast enhancment on the moonPhobos image, it
 was
%decided that the image looks best when gamma=0.25. This value makes
 the
%enhanced image much more clear and the contrast appears much higher
 when
%looking at subtle edges. Matlab's histeq enhancement, below the gamma
%enhancment, looks worse than the gamma enhancement. The contrast can
 still
%be seen in subtle edges (like the craters), but the entire image is
 washed
%out and unclear. Furthermore, the pixel histogram of the gamma
 enhanced
%image is much more evenly distributed compared to Matlab's histeq
%enhancement, which is skewed towards 255 with a much lower dynamic
```

```
%range.')

function [Newimg] = gammacorrection(gamma,img)
img = double(img); %Make image value be double type
[Rows,Cols]= size(img); %Get dimensions of image
for I= 1:Rows %perform correction on each pixel value
    for K = 1:Cols
        Newimg(I,K) = 255*( img(I,K) /255).^(gamma); %Gamma correction
 equation
    end
end

end
```

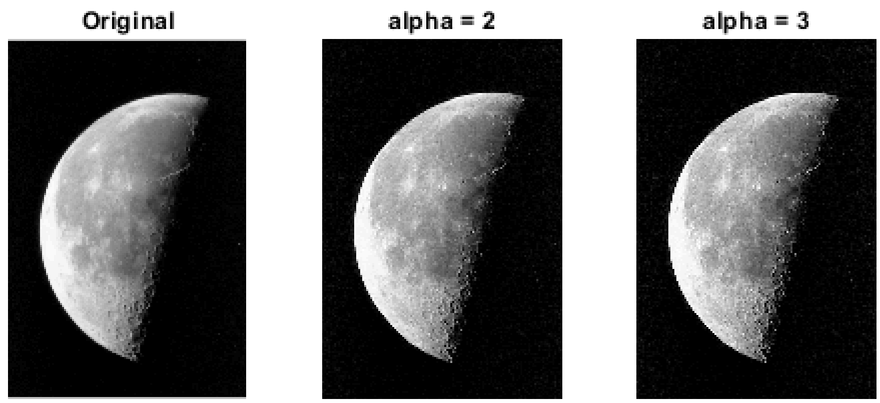


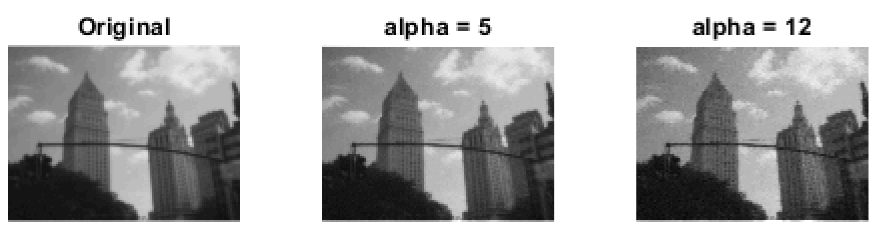


Part 2 - Sharpening Enhancment on Images

```
%Sharpen enhancment of "moon" image using laplacian filter

moon = imread('moon.tif'); %Read in "moon" image
type sharpen

figure(3)
subplot(1,3,1);
imshow(moon); %Plot the original image
title("Original");

subplot(1,3,2);
alpha = 2; %User specified scaling constant alpha
NewImg = sharpen(alpha,moon); %Apply the sharpen function to the image
imshow(NewImg) %Plot the alpha = 2 enhanced image
title 'alpha = 2';

subplot(1,3,3);
alpha = 3; %User specified scaling constant alpha
NewImg2 = sharpen(alpha,moon); %Apply the sharpen function to the
 image
imshow(NewImg2) %Plot the alpha = 3 enhanced image
title 'alpha = 3';
```

```
% Sharpen outoffocus

figure(4)
outoffocus = imread('outoffocus.tif');
subplot(1,3,1);
imshow(outoffocus); %Plot the original image
title("Original");

subplot(1,3,2);
alpha = 5;
newimg = sharpen(alpha,outoffocus); %Apply the sharpen function to the
 image
imshow(newimg) %Plot the alpha = 5 sharpened image
title 'alpha = 5';

subplot(1,3,3);
alpha = 12;
newimg = sharpen(alpha,outoffocus); %Apply the sharpen function to the
 image
imshow(newimg) %Plot the alpha = 12 sharpened image
title 'alpha = 12';


function [Newimg] = sharpen(alpha,img)
img = double(img); %Make image value be double type
[Rows,Cols]= size(img); %Get dimensions of image
LF = [0     -0.25 0; %Create the Laplacian Filter matrix
    -0.25 1 -0.25;
    0     -0.25 0];

    gxy = filter2(LF,img); %use the laplacian filter to obtain g(x,y)

    Newimg = uint8(img+alpha*gxy); %Obtain the sharpened image
 according to equation 2, and made into unit8 type
end
```


Original
alpha = 2
alpha = 3

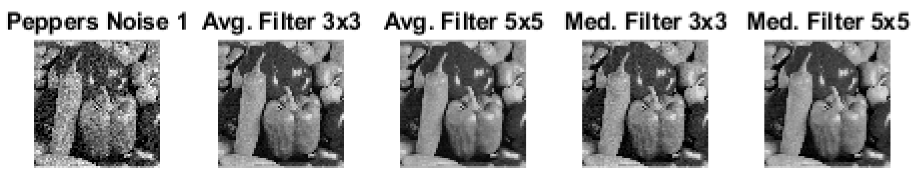

Original
alpha = 5
alpha = 12

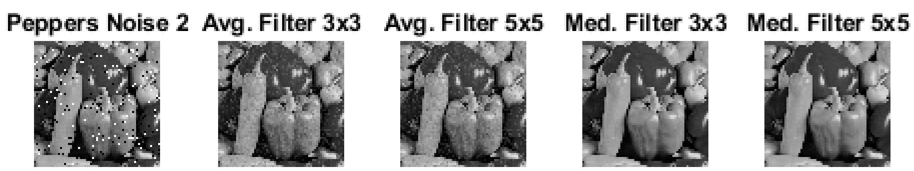
Part 3 - Denoising Images

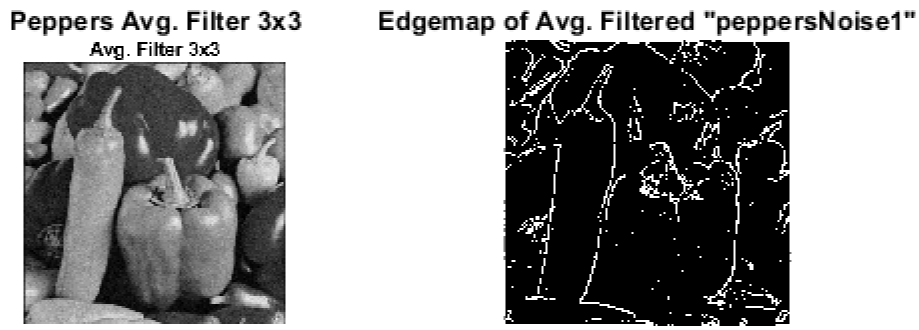
```
%Denoising of two Peppers images, using median filter and the
 averaging
%filter, window sizes including 3 ×3 pixels and 5 × 5 pixels

figure(5)

subplot(2,5,1); %Create a 2x5 plot to compare denoised images
peppersnoise1 =  imread('peppersNoise1.tiff'); %Read in image
imshow(peppersnoise1); %Display Noised image
title('Peppers Noise 1');

subplot(2,5,2);
avgfilter3x3= ones(3)/9; %create the averaging filter matrix 3x3
 window size
peppers1Avg3= filter2(avgfilter3x3,peppersnoise1);
imshow(uint8(peppers1Avg3)) %Convert and show image back to unit8
title('Avg. Filter 3x3');

subplot(2,5,3);
avgfilter5x5= ones(5)/25; %create the averaging filter 5x5 matrix
 window size
peppers1Avg5= filter2(avgfilter5x5,peppersnoise1);
imshow(uint8(peppers1Avg5)) %Convert and show image back to unit8
title('Avg. Filter 5x5');

subplot(2,5,4);
peppers1Med3= medfilt2(peppersnoise1,[3 3]); %apply the median filter
 with 3x3 window size
imshow(peppers1Med3)
title('Med. Filter 3x3')

subplot(2,5,5);
peppers1Med5= medfilt2(peppersnoise1,[5 5]); %apply the median filter
 with 5x5 window size
imshow(peppers1Med5)
title('Med. Filter 5x5')

subplot(2,5,6);
peppersnoise2 =  imread('peppersNoise2.tiff'); %Read in image
imshow(peppersnoise2); %Show original image
title('Peppers Noise 2');

subplot(2,5,7);
avgfilter3x3 = ones(3)/9; %create the averaging filter matrix 3x3
 window size
peppers2Avg3= filter2(avgfilter3x3,peppersnoise2);
imshow(uint8(peppers2Avg3)) %Convert and show image back to unit8
title('Avg. Filter 3x3');

subplot(2,5,8);
```

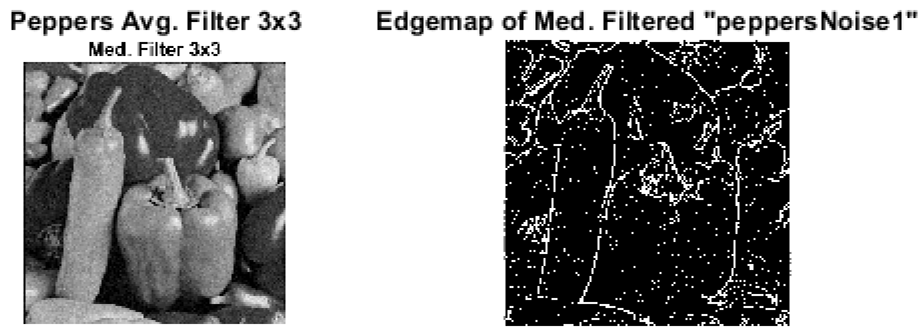
```
avgfilter5x5= ones(5)/25; %create the averaging filter matrix 5x5
 window size
peppers2Avg5= filter2(avgfilter3x3,peppersnoise2);
imshow(uint8(peppers2Avg5)) %Convert and show image back to unit8
title('Avg. Filter 5x5');

subplot(2,5,9);
peppers2Med3= medfilt2(peppersnoise2,[3 3]); %apply the median filter
 with 3x3 window size
imshow(peppers2Med3)
title('Med. Filter 3x3')

subplot(2,5,10);
peppers2Med5= medfilt2(peppersnoise2,[5 5]); %apply the median filter
 with 5x5 window size
imshow(peppers2Med5)
title('Med. Filter 5x5')


%Edgemaps using Sobel filters on Med. and Avg. filtered peppersNoise1
 Image

SobelX =  [ -1 0 1
            -2 0 2
            -1 0 1 ]; % Create the first Sobel filter (x)
SobelY = SobelX'; % Create the second Sobel filter (y) by transposing
 Sy

figure(6)


peppersavg3 = imread('peppersavg3.tif'); %Read in denoised image
subplot(2,2,1) %Create a 1x3 matrix subplot to compare the denoised
 image and two filtered images
imshow(peppersavg3) %Show the Denoised image
title('Peppers Avg. Filter 3x3');

peppersmed3 = imread('peppersmed3.tif'); %Read in denoised image
subplot(2,2,3) %Create a 1x3 matrix subplot to compare the denoised
 image and two filtered images
imshow(peppersmed3) %Show the Denoised image
title('Peppers Avg. Filter 3x3');

Gx1= filter2(SobelX,peppers1Avg3); % Apply the sobel filter to the
 rows, x-axis
Gy1= filter2(SobelY,peppers1Avg3); % Apply the sobel filter to the
 columns, y-axis
gradient1 = (Gx1.^2 + Gy1.^2).^.5; % create the gradient equation
threshold= 128; %User defined threshold value
edgemap1 = gradient1 > threshold; %Create the edgemap comparing the
 gradient to the threshold
subplot(2,2,2)
imshow(edgemap1)
title('Edgemap of Avg. Filtered "peppersNoise1"');
```

```
Gx2 = filter2(SobelX,peppers1Med3); % Apply the sobel filter to the rows, x-axis
Gy2 = filter2(SobelY,peppers1Med3); % Apply the sobel filter to the columns, y-axis
gradient2 = (Gx2.^2 + Gy2.^2).^.5; % create the gradient equation
edgemap2 = gradient2 > threshold; % Create the edgemap comparing the gradient to the threshold
subplot(2,2,4)
imshow(edgemap2)
title('Edgemap of Med. Filtered "peppersNoise1"');
```









Published with MATLAB® R2019b