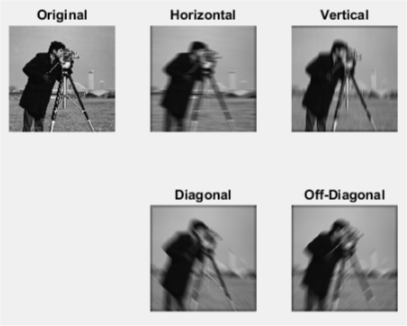
**DIP SESSIONAL 3**

**150222**

**EXERCISE 1**

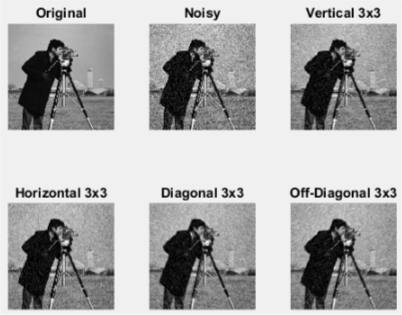
This problem was to understand directional filtering.

1. This told to create convolution kernel for user defined dimensions and to create function to do son and with the result filter the image. For this purpose, ‘cameraman.tif’ was taken then using input() the dimension was taken then it was being processed so that one vertical, one horizontal, one diagonal and one off-diagonal filer of the dimension could be found and then using imfilter() the image was processed.

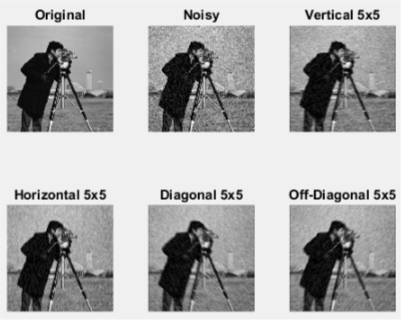


This result was obtained by taking the dimension 15 to understand the direction effect. The images were all blurred but those blurred images have direction of the blur effect. Vertical effect showed the blur effect to be appeared as vertical and same goes as for the other three directions. This was understood that the filtering effected depended on the values of the filter.

1. This told to read ‘cameraman.tif’ image then using randn() add noise to the image and scale it by 10. Then using the previous method but fixing the dimensions at 3 ,5 and 7 the image was processed.

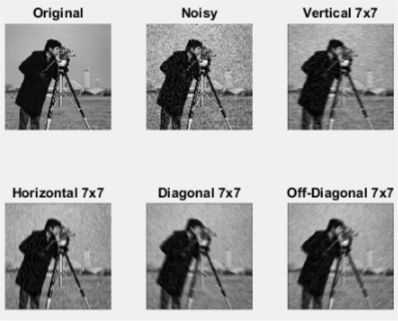


This is after the noise is added then processed with 3x3 directional filter, here the directional effect is clearer than the normal image. Also seen that noise seem to be reduced by applying the filter.



This is processed with 5x5 filter. This also clears the direction effect also for diagonal and off-diagonal the reduction of noise is observed much more.

And finally, the last image shows the result of processing the image by 7x7 directional filtering. The noise reduction is much quicker but the blurred effect is much more observed here.



The 3x3 filter keep the image less blurry but fails to remove more noise, on the other hand, 7x7 filter removes more noise but the image becomes blurrier. So, it is seen that 5x5 filter does approximately better job than others.

**CODE :**

% reading the image

I = imread('cameraman.tif');

% taking the dimension from user

D = input('Enter the dimension of the filter : ');

% displaying the original image

figure;

subplot(2,3,1),imshow(I),title('Original');

% using vertical filter function and displaying image

Ih = dirfilt(I,D);

subplot(2,3,2),imshow(Ih),title('Horizontal');

% using horizontal filter function and displaying the image

Iv = dirfilt2(I,D);

subplot(2,3,3),imshow(Iv),title('Vertical');

% diagonal filtering by taking a diagonal matrix using eye() to generate

% matrix

Id = imfilter(I,(eye(D)/D));

subplot(2,3,5),imshow(Id),title('Diagonal');

% same as above but rotating it by 90 degrees

Iod = imfilter(I,rot90(eye(D)/D));

subplot(2,3,6),imshow(Iod),title('Off-Diagonal');

% adding noise scaled by 10

In = im2double(I) + (randn(size(I))/10);

% showing for dimension 3

figure;

subplot(2,3,1),imshow(I),title('Original');

subplot(2,3,2),imshow(In),title('Noisy');

subplot(2,3,3),imshow(dirfilt(In,3)),title('Vertical 3x3');

subplot(2,3,4),imshow(dirfilt2(In,3)),title('Horizontal 3x3');

subplot(2,3,5),imshow(imfilter(In,(eye(3)/3))),title('Diagonal 3x3');

subplot(2,3,6),imshow(imfilter(In,rot90(eye(3)/3))),title('Off-Diagonal 3x3');

% showing for dimension 5

figure;

subplot(2,3,1),imshow(I),title('Original');

subplot(2,3,2),imshow(In),title('Noisy');

subplot(2,3,3),imshow(dirfilt(In,5)),title('Vertical 5x5');

subplot(2,3,4),imshow(dirfilt2(In,5)),title('Horizontal 5x5');

subplot(2,3,5),imshow(imfilter(In,(eye(5)/5))),title('Diagonal 5x5');

subplot(2,3,6),imshow(imfilter(In,rot90(eye(5)/5))),title('Off-Diagonal 5x5');

% showing for dimension 5

figure;

subplot(2,3,1),imshow(I),title('Original');

subplot(2,3,2),imshow(In),title('Noisy');

subplot(2,3,3),imshow(dirfilt(In,7)),title('Vertical 7x7');

subplot(2,3,4),imshow(dirfilt2(In,7)),title('Horizontal 7x7');

subplot(2,3,5),imshow(imfilter(In,(eye(7)/7))),title('Diagonal 7x7');

subplot(2,3,6),imshow(imfilter(In,rot90(eye(7)/7))),title('Off-Diagonal 7x7');

**FUNCTIONS :**

% function for vertical direction filtering

function FI = dirfilt(I,D)

if D>1

h = zeros(D);

for i = 1:D

for j = 1:D

if (i == ceil(D/2))

h(i,j) = 1;

else

h(i,j) = 0;

end

end

end

h = h/D;

end

FI = imfilter(I,h);

% function for horizontal direction filtering

function FI = dirfilt2(I,D)

if D>1

h = zeros(D);

for i = 1:D

for j = 1:D

if (j == ceil(D/2))

h(i,j) = 1;

else

h(i,j) = 0;

end

end

end

h = h/D;

end

FI = imfilter(I,h);

**EXERCISE 2**

This problem was to learn intensity transform by learning unsharp masking. This could be easily done by using fspecial() and giving the parameter ‘unsharp’ which will achieve what was asked in the question. Then imfilter() will finally filter the image and sharpen it.



The unsharp masking sharpens the image. From the observation it is shown that the spots which were not sharp became more sharp using the unsharp masking and as the question told to give a 3x3 mask was not needed to be handled as the functions handle the mask size and default is 3x3.

**CODE :**

I = imread('moon.tif');

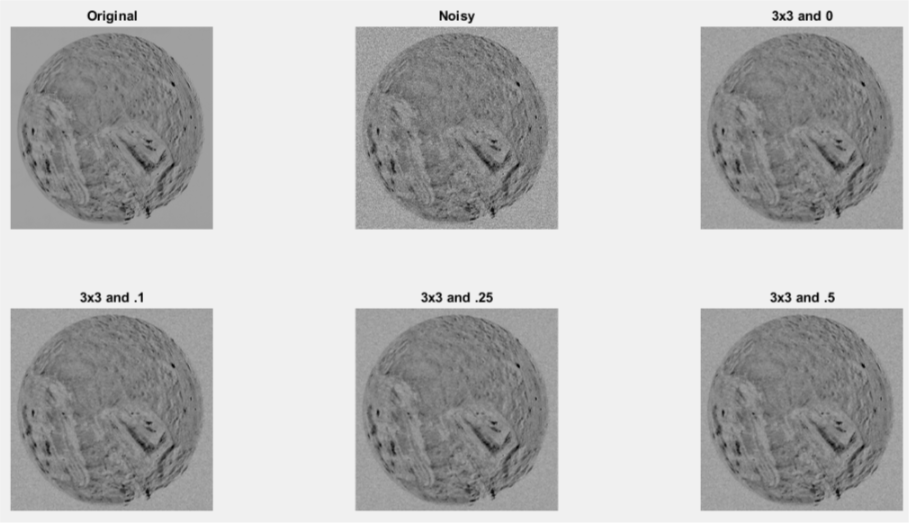
subplot(1,2,1),imshow(I),title('Original');

subplot(1,2,2),imshow(imfilter(I,fspecial('unsharp'))),title('Sharpened');

**EXERCISE 3**

This problem told to use a median+threshold filter rather than using one of those as that is supposed to give better output.

1. This told to load the threshold median filter medfilt\_th.m to load in the working directory. After downloading the .m file it was kept to the working directory.
2. The function was read.
3. Noise was added on the image using imnoise() and the Poisson effect was given as noise.
4. Finally experimentation was done by giving the matrix size and threshold value and using the medfilt\_th() function.



The observation tells that is the filter size is increased for any threshold value the noise is removed more but the image loses its sharpness so it was found that for filter size 3x3 the image remained much sharper. And for 3x3 filter the increase in threshold removes more noise but the image losses its sharpness. It is seen that for threshold 0 and filter size 3x3 the noise eliminated much and the image preserve its sharpness. But for ‘speckle’ noise the lesser filter did not do any good in removing filter. For that kind of noise higher filter with higher threshold value gives better result.

**CODE :**

I = imread('miranda.tif');

subplot(2,3,1),imshow(I),title('Original');

In = imnoise(I,'poisson');

subplot(2,3,2),imshow(In),title('Noisy');

Im = medfilt\_th(In,3,0);

subplot(2,3,3),imshow(Im),title('3x3 and 0');

Im = medfilt\_th(In,3,.1);

subplot(2,3,4),imshow(Im),title('3x3 and .1');

Im = medfilt\_th(In,3,.25);

subplot(2,3,5),imshow(Im),title('3x3 and .25');

Im = medfilt\_th(In,3,.5);

subplot(2,3,6),imshow(Im),title('3x3 and .5');