3EC401ME24 Image Processing

Tutorial - 3 Date: 23/08/2023 Roll No :- 22bec121

1. Write a MATLAB code to smoothen the image using average and weighted average filter. Use different filter sizes and compare the outputs.

```
img = imread("C:\Users\snehs\Downloads\WhatsApp Image 2024-04-11 at 4.26.29
PM.jpeg");
imshow(img);
```

```
grayimg = uint8(mean(img,3));
figure(2); imshow(grayimg);
```

```
h = (1/9) * [1 1 1;1 1 1;1 1 1] ;
filtimg = imfilter(grayimg,h,'replicate') ;

imshow(filtimg) ; title("Averaging Filter 3x3") ;
```

Averaging Filter 3x3

```
h = (1/36) * [ones(6,6)] ;
filtimg1 = imfilter(grayimg,h,'replicate') ;

imshow(filtimg1) ; title("Averaging Filter 6x6") ;
```

Averaging Filter 6x6

```
newimg = grayimg - filtimg1 ;
imshow(newimg) ;
```

Inference :- Here , we learnt to apply Basic Averaging Filter using self defined masks for 3x3 and 6x6 filter . We can clearly see that the imaging is smoothening , i.e. the edges are slowly vanishing .

```
havg = (1/81) * [1 2 3 2 1;
                 2 4 5 4 2;
                 3 5 9 5 3;
                 2 4 5 4 2;
                 1 2 3 2 1];
filtimg2 = imfilter(grayimg,havg,'replicate') ;

imshow(filtimg2) ; title("Weighted Averaging Filter 5x5") ;
```

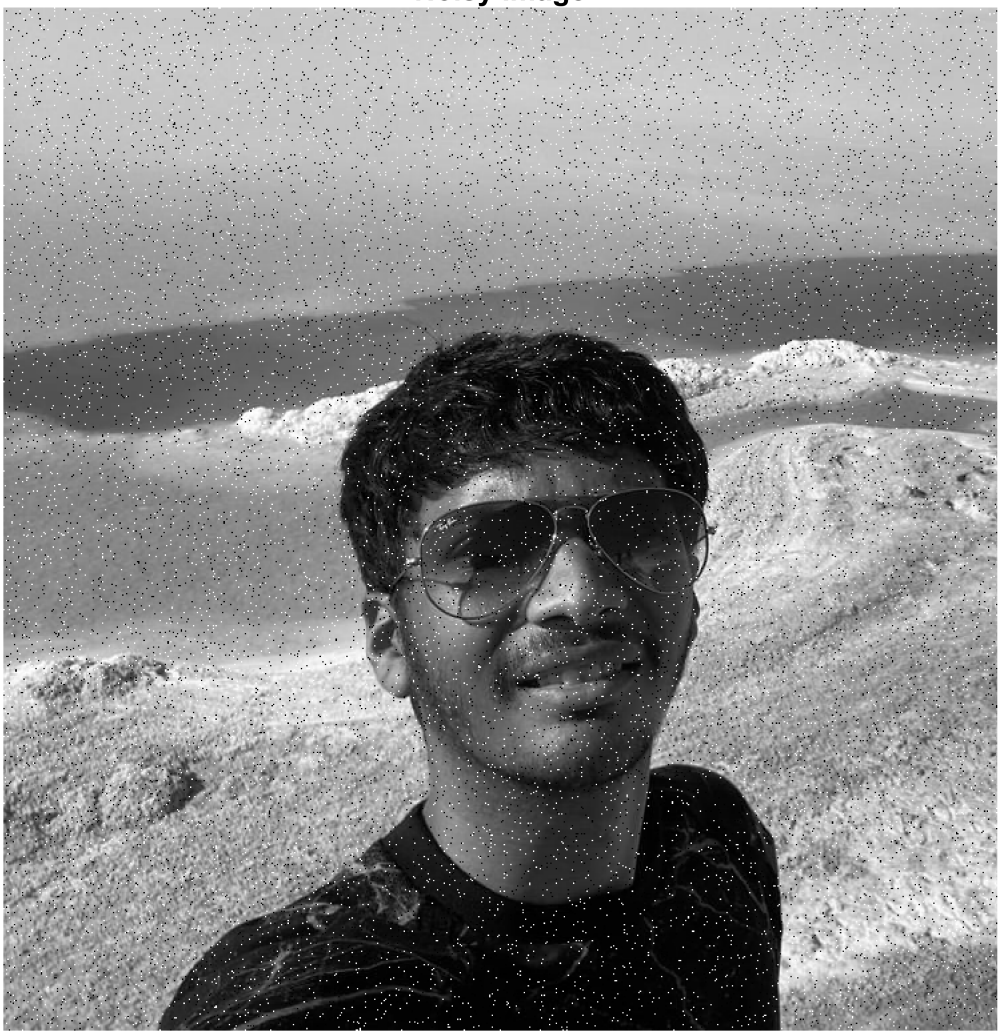
Weighted Averaging Filter 5x5

Inference :- Here , we learnt to apply 6x6 Weighted Average filter on the gray image , we also observed that the above image is better than Simple Averaging Filter of 6x6 order .

2. Write a program to remove salt and pepper noise from an image using smoothing filters. Plot the noisy and average images and observe the difference in the output.

```
noisimg = imnoise(grayimg,'salt & pepper',0.03) ;
imshow(noisimg) ; title("Noisy Image") ;
```

Noisy Image



```
h = (1/36) * [ones(6)] ;
filtimg3 = imfilter(noisimg,h,'replicate') ;

imshow(filtimg3) ; title("Averaging Filter 6x6") ;
```

Averaging Filter 6x6

```
noise_remove = imfilter(grayimg,h);
imshow(noise_remove);
title("Smoothened Image after removing noise");
```

Smoothened Image after removing noise

```
montage({noisimg,noise_remove});
```

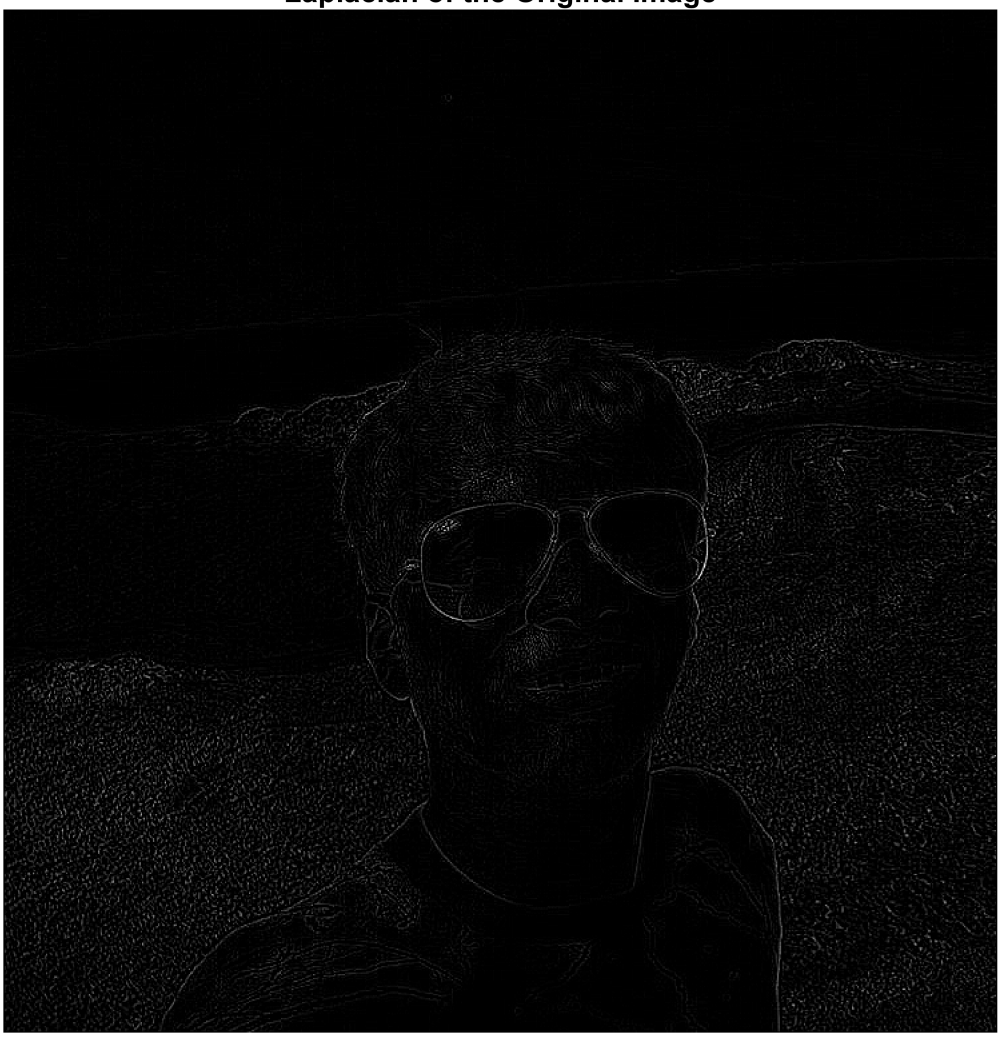
Inference :- Here , we learnt to add Salt and Pepper Noise in the image and then use the Averaging Filter to remove the noise as well as use the in-built mask to filter out the noise from the image .

We can clearly see that the noise is almost removed from the image .

3. Write a MATLAB code to sharpen an image smoothen in exercise 1 using sharpening filters. Compare the original image (before smoothing) and the sharpened image and plot them.

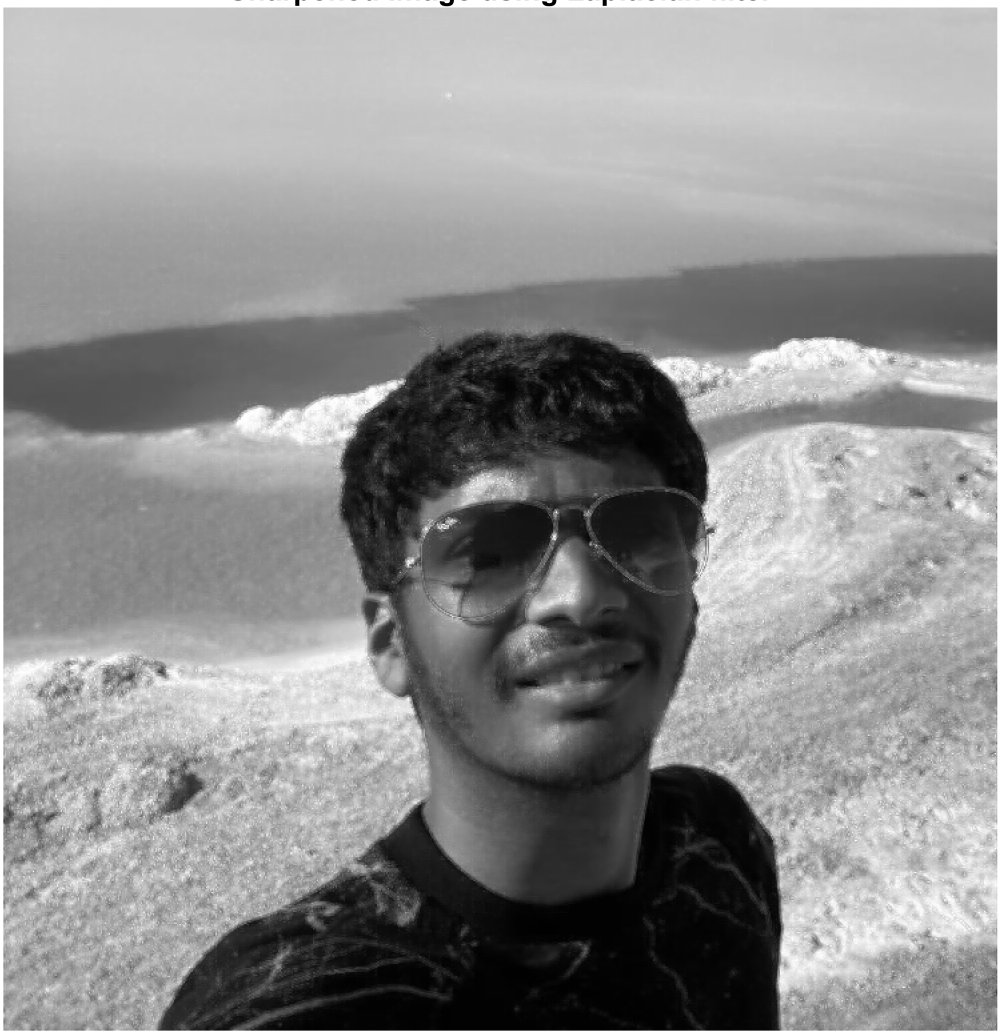
```
f = fspecial('laplacian');
lap_img = imfilter(grayimg,f); imshow(lap_img);
title("Laplacian of the Original Image");
```

Laplacian of the Original Image



```
sharimg = grayimg + lap_img ;
imshow(sharimg) ; title("Sharpened Image using Laplacian filter") ;
```

Sharpened Image using Laplacian filter



```
montage({grayimg,sharimg});
```

Inference :- Here , we learnt to apply sharpening filter like Laplacian filter to detect edged and further added the image to the original gray image to get a Sharpened Image . We can clearly see that the edged are highlighted in the Sharpened Image compared to the gray image .

4. Create a file to sharpen an image by:

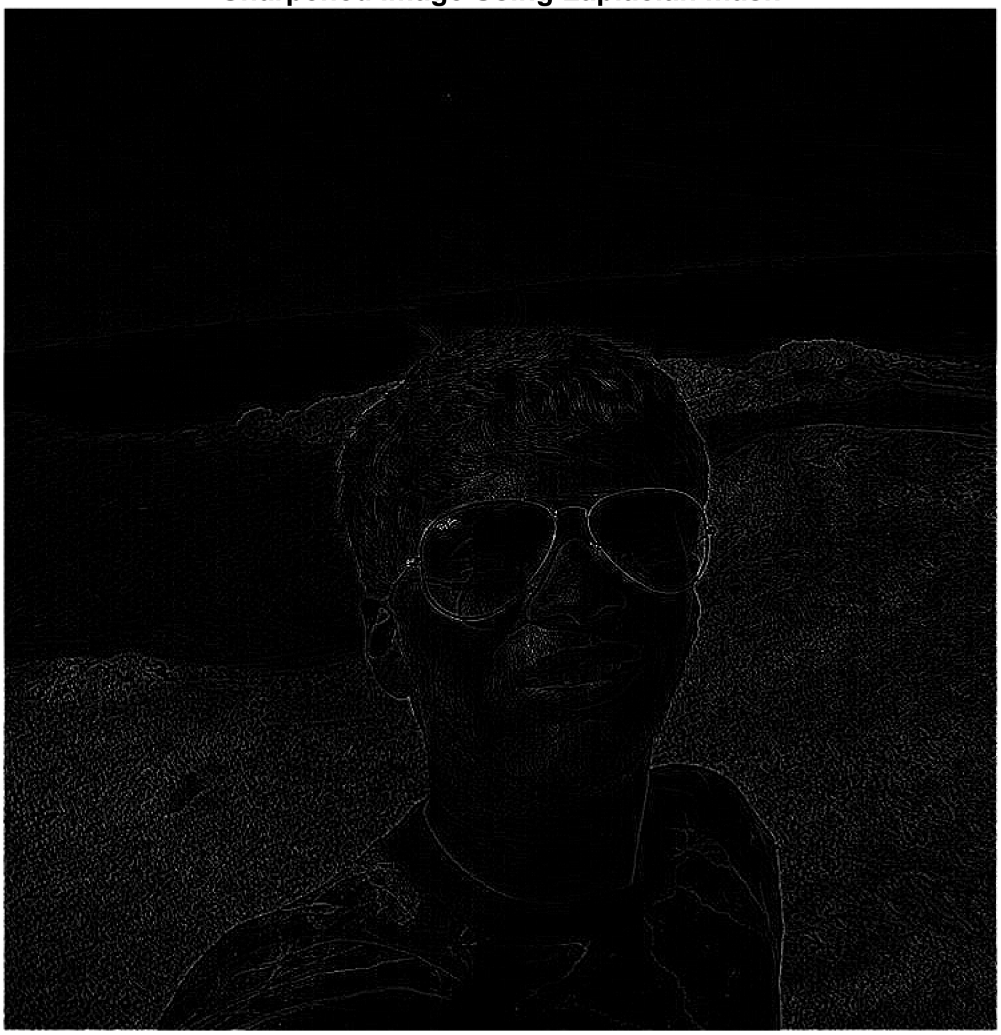
a. creating a filter/ mask on your own

b. subtracting the output of Laplacian from the original image

Plot both the images and comment upon the output.

```
h2 = [0 -1 0 ; -1 4 -1 ; 0 -1 0]; % laplacian mask
sharp = imfilter(grayimg,h2);
imshow(sharp); title("Sharpened Image Using Laplacian mask") ;
```

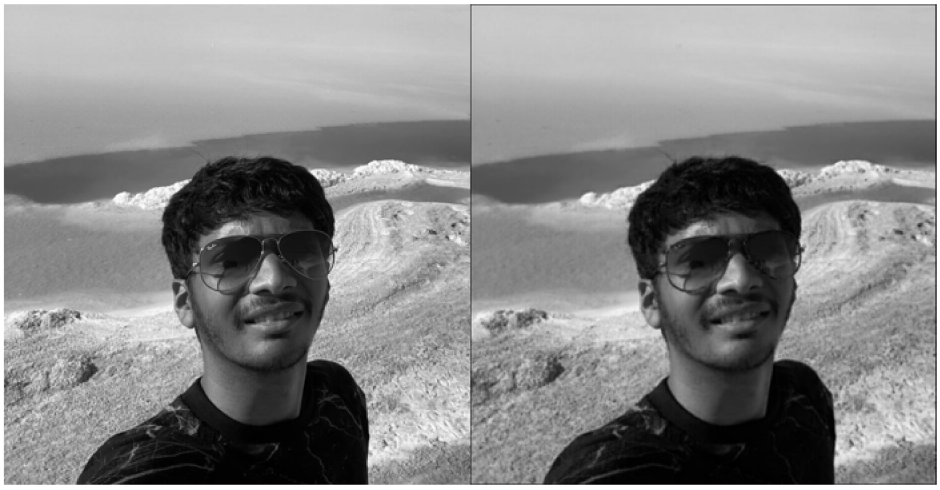
Sharpened Image Using Laplacian mask



```
sharimg2 = grayimg + sharp ;
montage({grayimg,sharimg2}) ;
```

Inference :- Here , we created 3x3 Laplcian mask and appied it to a gray image which gives output as only edge image then we added the gray image on the edge image , so we get the final sharpened image .

```
subimg = grayimg - sharp ;
montage({grayimg , subimg});
```

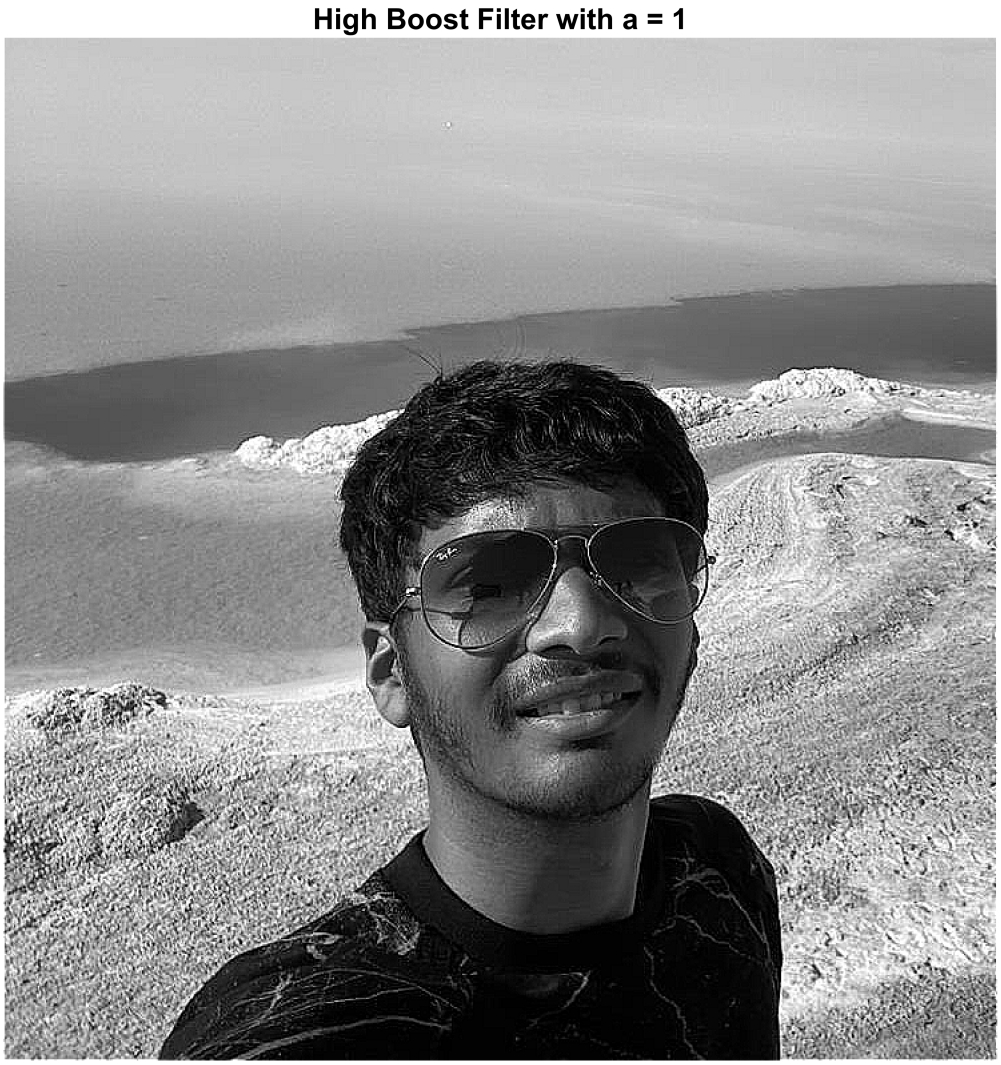


Inference :- Here , we subtracted the edge image from the original gray image so we get a smoothened image , because the edged have been removed so we get a smooth image as the output.

5. Write a program to perform high boost filtering on the image. Choose different values of ‘A’ and comment on the effect of ‘A’ on the output.

```
% for a = 1:6
% h3 = [0 -1 0 ; -1 (a+6) -1 ; 0 -1 0]; % defining high boost filtering
% mask

h = fspecial('unsharp');
for a = [1, 2, 3, 4, 5]
    high_boost = (a - 1) * grayimg + imfilter(grayimg, h);
    figure, imshow(high_boost, []);
    title(['High Boost Filter with a = ', num2str(a)]);
end
```



High Boost Filter with a = 2

High Boost Filter with a = 3

High Boost Filter with a = 4

High Boost Filter with a = 5

Inference :- Here , we applied High Boost Transform on the gray image for different values of A . We can clearly see that as we increase the value of A , we get to observe that the intensities of th pixels keep increasing . Hence , at A = 5 , we can only see the black color part properly like the Tshirt and Hair .

Conclusion :-

From this experiment , we can demonstrate the effectiveness of various filtering techniques in image processing, showcasing how smoothing reduces noise and sharpening enhances details. The high-boost filter highlights the importance of tuning parameters like 'A' to balance detail enhancement with overall image quality.