

Name: _____

Roll No. _____

Batch: _____

Experiment No.: _____

Title of Experiment: _____

Date of Performance: _____

Date of Submission: _____

Grade and Faculty Signature: _____

Experiment No. 4**Spatial Filtering: Smoothing**

Aim: Spatial Filtering using

- A) Average Filter
- B) Median Filtering

Objectives:

1. To blur the image using average and median filters.
2. To remove the noise (Gaussian, Salt & Pepper and Speckle Noise) from given images and observe effectiveness of filters.

Learning Outcome:

1. Students will be able to choose appropriate filter to blur the image depending upon the application.
2. Students will be able to select proper filtering technique for removal of specific noise.

Software Used: Scilab with SIVP toolbox

Theory:

In image processing, to smooth a data set is to create an approximating function that attempts to capture important patterns in the data, while leaving out noise or other fine-scale structures/rapid phenomena. In smoothing, the data points of a signal are modified so individual points (presumably because of noise) are reduced, and points that are lower than the adjacent points are increased leading to a smoother signal.

Mask Processing:

An image can be modified by applying a particular function to each pixel value. Neighbourhood processing be considered as an extension of this, where a function is applied to a neighbourhood of each pixel. The idea is to move a 'mask': a rectangle (usually with sides of odd length) or other shape over the given image. As we do this, we create a new image whose pixels have gray values calculated from the gray values under the mask, as shown in Figure 3.1. The combination of mask and function is called a filter. If the function by which the new gray value is calculated is a linear function of all the gray values in the mask, then the filter is called a linear filter.

Mechanics of Spatial Filtering:

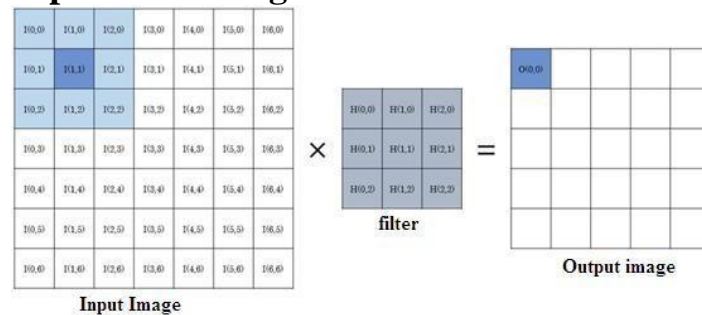


Fig. 3.1: Mechanics of spatial filtering

Two smoothing filters are used in this experiment:

- Average filter
- Median Filter

A) Average Filter:

Average filtering is a method of reducing the amount of intensity variation between one pixel and the next. It is often used to reduce noise in images. The idea of mean filtering is simply to replace each pixel value in an image with the mean ('average') value of its neighbours, including itself. This has the effect of eliminating pixel values which are unrepresentative of their surroundings. Mean filtering is usually thought of as a convolution filter. Like other convolutions it is based around a kernel, which represents the shape and size of the neighbourhood to be sampled when calculating the mean. Often a 3×3 square kernel is used, although larger kernels (e.g. 5×5 squares) can be used for more severe smoothing.

$$\frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

B) Median Filtering:

Median filtering is order statistic based smoothing technique works similar to linear Gaussian filtering. Median filter considers neighbours pixels in block/window to decide whether or not it is representative of its surroundings. Instead of simply replacing the pixel value with the *mean* of neighbouring pixel values, it replaces it with the *median* of those values. The median is calculated by first sorting all the pixel values from the surrounding neighbourhood into numerical order and then

replacing the pixel being considered with the middle pixel value as shown in Fig 3.2. The smoothing techniques remove noise, but adversely affect the edges. Edges have critical importance to the visual appearance of images. Obviously, it is necessary to reduce the noise whilst preserve the edges.

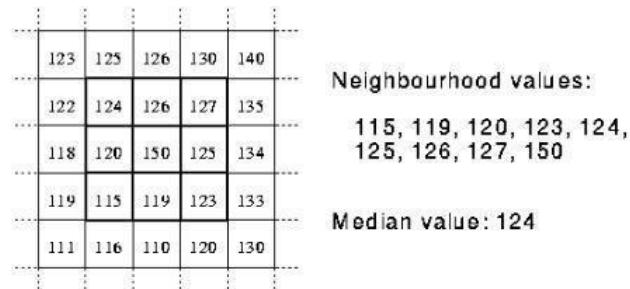


Fig. 3.2: Median Filtering process

For small to moderate levels of Gaussian noise, the median filter is demonstrably better than Gaussian blur at removing noise whilst preserving edges for a given fixed window size. However, its performance is much better for speckle noise and salt-and-pepper noise (impulsive noise) compared to Gaussian blur for high levels of noise.

Algorithm:

1. Read image.
2. Take mask elements /window size for median filtering from user.
3. Apply 2D convolution.
4. Replace corresponding value of centre pixel.
5. Apply average and median filtering on normal image, Gaussian noise image, salt&pepper noise image or speckle noise image
6. Display output images with input image simultaneously.

Conclusion:

1. Blurring is more in average filter than median filter. Median filter preserves the edges.
2. Average filters are better for Gaussian noise removal compared to median filter.
3. Median filter is the most effective smoothing technique for salt & pepper noise than Gaussian noise.

Code:**1. Code for Average Filtered Image**

```
//EXPT 4: SPATIAL FILTERING
```

```
//NAME: JIGNESH BHANDI
```

```
//ROLL NO:07
```

```
clc;
```

```
clear;
```

```
a=imread('cat 256.bmp');
```

```
b1=imnoise(a,'gaussian');//gaussian noise
```

```
b1=imnoise(a,'salt & pepper');//salt & pepper noise
```

```
b1=imnoise(a,'speckle');//speckle noise
```

```
b=double(b1);
```

```
p=double(a)
```

```
mask=[1 1 1;1 1 1;1 1 1]
```

```
[row col]=size(b);
```

```
for x=2:row-1
```

```
    for y=2:col-1
```

```
        temp=mask(1,1)*b(x-1,y-1)+mask(1,2)*b(x-1,y)+mask(1,3)*b(x-1,y+1)+mask(2,1)*b(x,y-1)+mask(2,2)*b(x,y)+mask(2,3)*b(x,y+1)+mask(3,1)*b(x+1,y-1)+mask(3,2)*b(x+1,y)+mask(3,3)*b(x+1,y+1);
```

```
        p(x,y)=round(temp/9);
```

```
    end
```

```
end
```

```
d=[a uint8(b1) uint8(p)];
```

```
imshow(d)
```

2. Code for Median Filtered Image

```
//EXPT 4a: SPATIAL FILTERING
//NAME: Jignesh Bhandi
//ROLL NO:07

clc;
clear;

a=imread('cat 256.bmp');
b1=imnoise(a,'gaussian'); //gaussian noise
b1=imnoise(a,'salt & pepper'); //salt & pepper noise
b1=imnoise(a,'speckle'); //speckle noise










b=double(a);
p1=double(b1);
mask=[1 1 1;1 1 1;1 1 1]
[row col]=size(b);

for x=2:row-1
    for y=2:col-1
        temp=[b(x-1,y-1),b(x-1,y),b(x-1,y+1),b(x,y-1),b(x,y),b(x,y+1),b(x+1,y-1),b(x+1,y),b(x+1,y+1)];
        temp1=gsort(temp);
        p1(x,y)=temp1(5);
    end
end

d=[a uint8(b1) uint8(p1)];

imshow(d)
```


Expected output:

Input Image	Average filtered Image	Median filtered image
Normal image 		
Gaussian noise 		
Salt & Pepper Noise 		

Speckle Noise

