DIGITAL IMAGE PROCESSING-1

Unit 3: Lecture 33-34

Dr. Shikha Tripathi
Department of Electronics &
Communication Engineering

DIGITAL IMAGE PROCESSING-1

Unit 3: Image Enhancement

Dr. Shikha Tripathi
Department of Electronics & Communication Engineering

Last Session

- Spatial / Neighborhood processing
  - Convolution
  - Correlation

Today's Session

- Spatial / Neighborhood processing Cont…
  - Convolution
  - Correlation
  - Averaging
  - Filtering

Types of Enhancement Techniques

Image Enhancement

Point Operations

- Contrast Stretching
- Noise Clipping
- Window Slicing
- Histogram Modeling

Spatial Operations

- Noise Smoothing
- Median Filtering
- Unsharp Masking
- LP,HP,BP filtering

Transform Operations

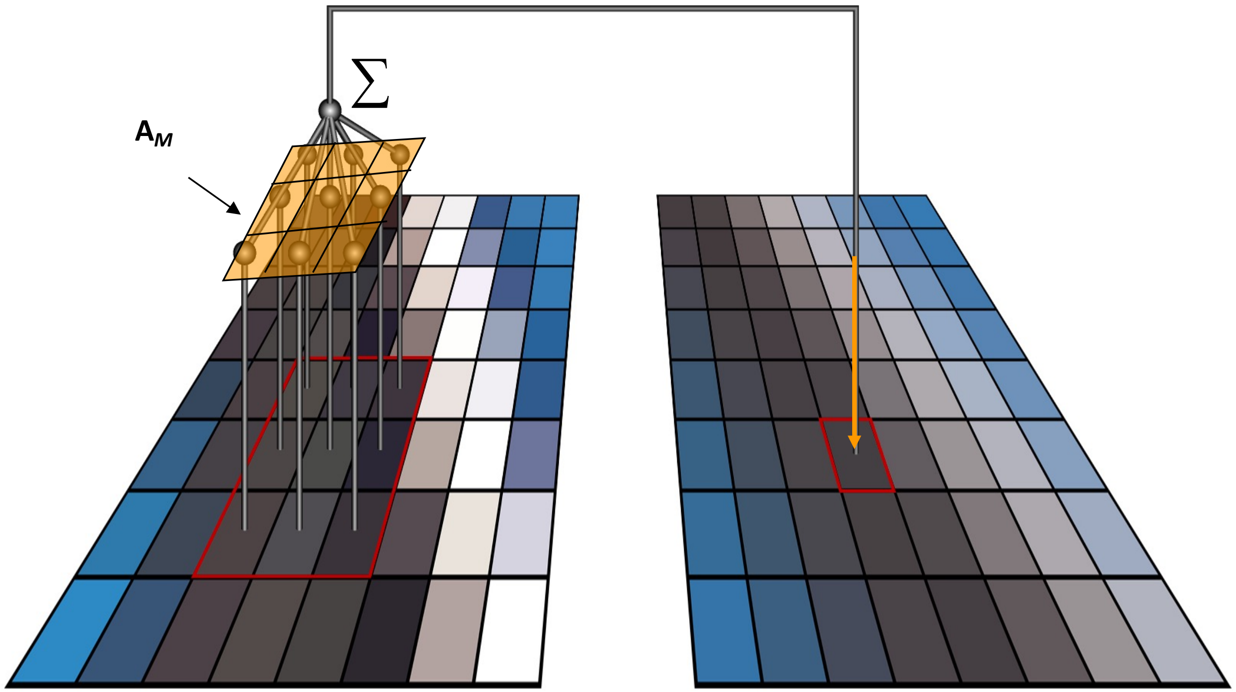
- Linear filtering
-
- Homomorphic Filtering

Pseudocoloring

- False Coloring
- Pseudocoloring

Spatial Domain

Spatial Operations (Neighbourhood Processing)



Local or Neighborhood Operations

- The pixels are modified based on some functions of the pixels in their neighborhood
- Work with the values of image pixels in the neighborhood and the corresponding values of a subimage that has the same dimension as the neighborhood
  - The subimage is called filter, mask, kernel, template or window
- Values in **filter** subimage (kernel) are referred to as **coefficients** rather than pixels

Neighborhood Pixel Processing

- **3 x 3 Neighborhood filter / Mask / Window / Kernel / Template:**

	(y - 1)	y	(y + 1)	Y
(x - 1)	W1 g(x-1, y-1)	W2 g(x-1, y)	W3 g(x-1, y+1)	
x	W4 g(x, y-1)	W5 g(x, y)	W6 g(x, y+1)	
(x + 1)	W7 g(x+1, y-1)	W8 g(x+1, y)	W9 g(x+1, y+1)	
X				

$$
\begin{aligned}
f(x, y) = {} & g(x-1,y-1).w1 + g(x-1,y).w2 + g(x-1,y+1).w3 \\
& + g(x,y-1).w4 + g(x, y).w5 + g(x,y+1).w6 \\
& + g(x+1,y-1).w7 + g(x+1,y).w8 + g(x+1,y+1).w9
\end{aligned}
$$

Image Convolution

- Convolution and correlation are used to extract information from images
- They are linear and shift invariant operations
  - Linear: Pixel is replaced by linear combination of its neighbors
  - Shift Invariant:

- **2D Convolution**

The *convolution* of a kernel w of size $m \times n$ with an image $f(x, y)$, denoted by $(w \bigstar f)(x, y)$,

$$(w \bigstar f)(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x - s, y - t)$$

Basic operations: Folding, Shifting, Multiplying, Adding

- **2D Correlation**

The correlation of a kernel w of size $m\ x\ n$ with an image $f(x, y)$, denoted as $(w \star f)(x, y)$

$$(w \star f)(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x + s, y + t)$$

Basic operations: Shifting, Multiplying, Adding

2D Convolution

Example 1: Perform linear convolution between the two matrices

$$x(m,n) = \begin{bmatrix} 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \qquad h(m,n) = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Basic operations: Folding, Shifting, Multiplying, Adding

- Convolved matrix is

$$y(m,n) = \begin{bmatrix} 4 & 5 & 6 \\ 11 & 13 & 15 \\ 11 & 13 & 15 \\ 7 & 8 & 9 \end{bmatrix}$$

Dimension of resultant matrix = {No. of rows of x(m,n) + No. of rows of h(m,n) -1} x {No. of columns of x(m,n) + No. of columns of h(m,n) -1}

Dimension of the given convolution =(2+3-1)x(3+1-1) = 4 x 3

2D Convolution

Example2: Perform linear convolution between the two matrices

$$x = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} ; h = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$y(x, y) = \begin{bmatrix} 5 & 16 & 12 \\ 22 & 60 & 40 \\ 21 & 52 & 32 \end{bmatrix}$$

Dimension of the given convolution

=(2+2-1)x(2+2-1)=3 x 3

Example 3: Perform convolution between the two matrices

$$x = \begin{bmatrix} 3 & 1 \\ 2 & 4 \end{bmatrix} ; h = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}$$

$$f(x,y) = \begin{bmatrix} 3 & 7 & 2 \\ 8 & 13 & 9 \\ 4 & 10 & 4 \end{bmatrix}$$

Correlation

Example 4: Perform correlation between the two matrices

$$x = \begin{bmatrix} 3 & 1 \\ 2 & 4 \end{bmatrix} ; h = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}$$

$$\gamma_{xy} = \begin{bmatrix} 3 & 7 & 2 \\ 8 & 13 & 9 \\ 4 & 10 & 4 \end{bmatrix}$$

The results of 2D convolution and correlation are the same if the kernel is symmetric

Properties of Correlation and Convolution

- 2D Correlation

$$(w \star f)(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t)$$

- 2D Convolution

$$(w \bigstar f)(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x-s, y-t)$$

Property	Convolution	Correlation
Commutative	$f \bigstar g = g \bigstar f$	—
Associative	$f \bigstar (g \bigstar h) = (f \bigstar g) \bigstar h$	—
Distributive	$f \bigstar (g+h) = (f \bigstar g) + (f \bigstar h)$	$f \star (g+h) = (f \star g) + (f \star h)$

Linear Filtering

- In general linear filtering of an image of size M x N with a filter mask of size m x n is given by (Since kernels are generally symmetric correlation and convolution give same result) :

$$g(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t)$$

where x =0,1,2, ……, M-1 and y = 0,1,2,….,N-1

a = (m-1)/2 and b = (n-1)/2

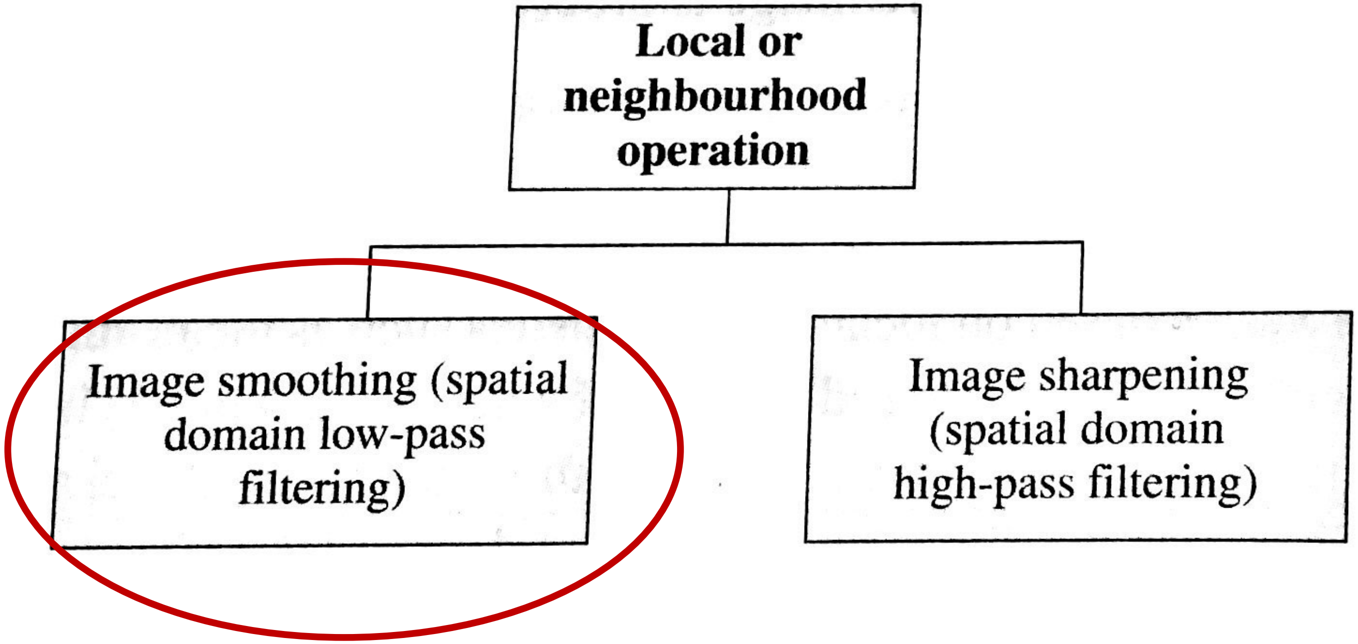
- Linear filtering is also called as Convolution
  - Convolving a mask with an image

Types of Spatial Filters

- **Spatial Filters**
    - Convolution based filters
    - Order-statistics(rank) filters
    - Hybrid filters

- **Convolution based filters:**
  - Use spatial masks known as kernels, templates or windows
  - A spatial mask is **convolved** with the given image to achieve the required **smoothing or sharpening effect**
- **Order-Statistics (rank/rank-order/order) filter:**
  - Do not use any convolution technique
  - Arrange the pixels under the mask in desired order
  - For ex. For median filter middle value is the chosen value
    - Used for image restoration
- **Hybrid filters:**
  - Use both ranking and convolution
    - Ex. Unsharp masking

Types of Local Operations



Spatial Smoothing (LPF Operations): Convolution Based

- **Smoothing linear filters:** Output is the average of the pixels contained in the neighborhood of the filter mask (also called as averaging filters or lowpass filters)
  - are used for blurring and noise reduction
    - Results in reduced sharp transitions in gray levels hence reduces noise (random noise consists of sharp transitions in gray levels)
  - Used in preprocessing steps (such as removal of small details from an image prior to object extraction)
  - Bridging of small gaps in lines or curves

Smoothing (Averaging) Filters

- Undesirable effect: blurring
  - Increasing the size of the mask results is more blurring
- A spatial averaging filter with equal coefficients is called **Box filter (separable)**

Box Filter (Separable)

- An m × n box filter is an m × n array of 1's, with a normalizing constant in front, whose value is 1 divided by the sum of the values of the coefficients (i.e., 1/mn when all the coefficients are 1's).
- This normalization, which we apply to all lowpass kernels, has two purposes.
  - First, the average value of an area of constant intensity would equal that intensity in the filtered image, as it should.
  - Second, normalizing the kernel in this way prevents introducing a bias during filtering; that is, the sum of the pixels in the original and filtered images will be the same

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

3 x 3 Averaging Mask

$$\frac{1}{25}\begin{bmatrix}1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\end{bmatrix}$$

5 x 5 Averaging Mask

Weighted average Filter

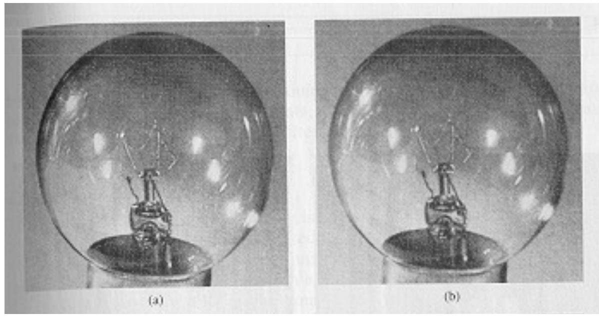
- This mask yields weighted average
  - Pixels are multiplied by different coefficients giving unequal importance to pixels

$$\frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

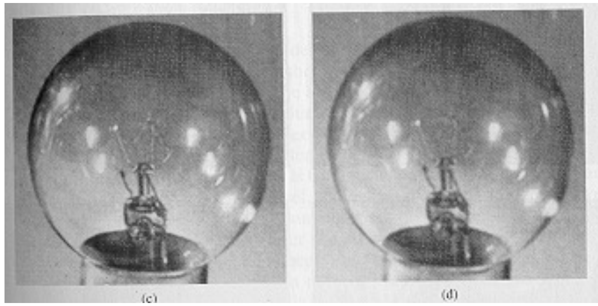
Effect of Size of Mask

- Mask size determines the degree of smoothing and loss of detail.

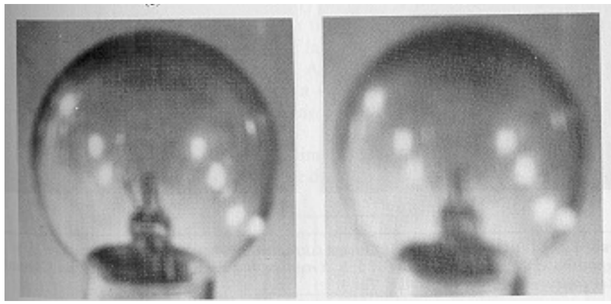
original 3x3



5x5 7x7



15x15 25x25



Spatial Smoothing (LPF Operations): Example

Neighborhood Pixels Processing

Ex. 1) 8x8 Pseudo image with a single edge (High Frequency) of 10 & 50. Remove using a 3x3 size averaging mask.

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

8x8 Image

Spatial Smoothing (LPF Operations): Example

Neighborhood Pixels Processing

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

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

DIGITAL IMAGE PROCESSING-1

Spatial Smoothing (LPF Operations): Example

Neighborhood Pixels Processing

0	0	0						
0	10	10	10	10	10	10	10	10
0	10	10	10	10	10	10	10	10
	10	10	10	10	10	10	10	10
	10	10	10	10	10	10	10	10
	50	50	50	50	50	50	50	50
	50	50	50	50	50	50	50	50
	50	50	50	50	50	50	50	50
	50	50	50	50	50	50	50	50

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

Spatial Smoothing (LPF Operations): Example

Neighborhood Pixels Processing

0	0	0						
0	4.44	10	10	10	10	10	10	10
0	10	10	10	10	10	10	10	10
	10	10	10	10	10	10	10	10
	10	10	10	10	10	10	10	10
	50	50	50	50	50	50	50	50
	50	50	50	50	50	50	50	50
	50	50	50	50	50	50	50	50
	50	50	50	50	50	50	50	50

$$\frac{1}{9}\begin{bmatrix}0 & 0 & 0\\ 0 & 10 & 10\\ 0 & 10 & 10\end{bmatrix}$$

DIGITAL IMAGE PROCESSING-1

Spatial Smoothing (LPF Operations): Example

Neighborhood Pixels Processing

0	0	0	0					
0	4.44	6.66	10	10	10	10	10	10
0	10	10	10	10	10	10	10	10
	10	10	10	10	10	10	10	10
	10	10	10	10	10	10	10	10
	50	50	50	50	50	50	50	50
	50	50	50	50	50	50	50	50
	50	50	50	50	50	50	50	50
	50	50	50	50	50	50	50	50

$$\frac{1}{9}\begin{bmatrix}0 & 0 & 0\\ 10 & 10 & 10\\ 10 & 10 & 10\end{bmatrix}$$

Spatial Smoothing (LPF Operations): Example

Neighborhood Pixels Processing

0	0	0	0	0				
0	4.44	6.66	6.66	10	10	10	10	10
0	10	10	10	10	10	10	10	10
	10	10	10	10	10	10	10	10
	10	10	10	10	10	10	10	10
	50	50	50	50	50	50	50	50
	50	50	50	50	50	50	50	50
	50	50	50	50	50	50	50	50
	50	50	50	50	50	50	50	50

$$\frac{1}{9}\begin{bmatrix} 0 & 0 & 0 \\ 10 & 10 & 10 \\ 10 & 10 & 10 \end{bmatrix}$$

Spatial Smoothing (LPF Operations): Example

Neighborhood Pixels Processing

0	0	0	0	0	0			
0	4.44	6.66	6.66	6.66	10	10	10	10
0	10	10	10	10	10	10	10	10
	10	10	10	10	10	10	10	10
	10	10	10	10	10	10	10	10
	50	50	50	50	50	50	50	50
	50	50	50	50	50	50	50	50
	50	50	50	50	50	50	50	50
	50	50	50	50	50	50	50	50

$$\frac{1}{9}\begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 10 & 10 & 10 \\ \hline 10 & 10 & 10 \\ \hline \end{array}$$

Spatial Smoothing (LPF Operations): Example

Neighborhood Pixels Processing

0	0	0	0	0	0	0	0	0	
0	4.44	6.66	6.66	6.66	6.66	6.66	6.66	4.44	0
0	6.66	10	10	10	10	10	10	10	0
	10	10	10	10	10	10	10	10	
	10	10	10	10	10	10	10	10	
	50	50	50	50	50	50	50	50	
	50	50	50	50	50	50	50	50	
	50	50	50	50	50	50	50	50	
	50	50	50	50	50	50	50	50	

$$\frac{1}{9}\begin{bmatrix}10 & 10 & 10\\ 10 & 10 & 10\\ 10 & 10 & 10\end{bmatrix}$$

Spatial Smoothing (LPF Operations): Example

Neighborhood Pixels Processing

0	0	0	0	0	0	0	0	0	
0	4.44	6.66	6.66	6.66	6.66	6.66	6.66	4.44	0
0	6.66	10	10	10	10	10	10	6.66	0
0	6.66	10	10	10	10	10	10	6.66	0
0	15.55	10	10	10	10	10	10	10	0
0	50	50	50	50	50	50	50	50	0
	50	50	50	50	50	50	50	50	
	50	50	50	50	50	50	50	50	
	50	50	50	50	50	50	50	50	

$$\frac{1}{9}\begin{array}{|c|c|c|}\hline 0 & 10 & 10 \\ \hline 0 & 10 & 10 \\ \hline 0 & 50 & 50 \\ \hline \end{array}$$

Spatial Smoothing (LPF Operations): Example

Neighborhood Pixels Processing

0	0	0	0	0	0	0	0	0	
0	4.44	6.66	6.66	6.66	6.66	6.66	6.66	4.44	0
0	6.66	10	10	10	10	10	10	6.66	0
0	6.66	10	10	10	10	10	10	6.66	0
0	15.55	23.33	10	10	10	10	10	10	0
0	50	50	50	50	50	50	50	50	0
	50	50	50	50	50	50	50	50	
	50	50	50	50	50	50	50	50	
	50	50	50	50	50	50	50	50	

$$\frac{1}{9}\begin{bmatrix} 10 & 10 & 10 \\ 10 & 10 & 10 \\ 50 & 50 & 50 \end{bmatrix}$$

Spatial Smoothing (LPF Operations): Example

Neighborhood Pixels Processing

0	0	0	0	0	0	0	0	0	
0	4.44	6.66	6.66	6.66	6.66	6.66	6.66	4.44	0
0	6.66	10	10	10	10	10	10	6.66	0
0	6.66	10	10	10	10	10	10	6.66	0
0	15.55	23.33	23.33	23.33	23.33	23.33	23.33	15.55	0
0	24.44	36.66	50	50	50	50	50	50	0
	50	50	50	50	50	50	50	50	
	50	50	50	50	50	50	50	50	
	50	50	50	50	50	50	50	50	

$$\frac{1}{9}\begin{bmatrix} 10 & 10 & 10 \\ 50 & 50 & 50 \\ 50 & 50 & 50 \end{bmatrix}$$

Spatial Smoothing (LPF Operations): Example

Neighborhood Pixels Processing

0	0	0	0	0	0	0	0	0	
0	4.44	6.66	6.66	6.66	6.66	6.66	6.66	4.44	0
0	6.66	10	10	10	10	10	10	6.66	0
0	6.66	10	10	10	10	10	10	6.66	0
0	15.55	23.33	23.33	23.33	23.33	23.33	23.33	15.55	0
0	24.44	36.66	36.66	36.66	36.66	36.66	36.66	24.44	0
0	33.33	50	50	50	50	50	50	50	
	50	50	50	50	50	50	50	50	
	50	50	50	50	50	50	50	50	

$$\frac{1}{9}\begin{bmatrix} 50 & 50 & 50 \\ 50 & 50 & 50 \\ 50 & 50 & 50 \end{bmatrix}$$

Spatial Smoothing (LPF Operations): Example

Neighborhood Pixels Processing

0	0	0	0	0	0	0	0	0	
0	4.44	6.66	6.66	6.66	6.66	6.66	6.66	4.44	0
0	6.66	10	10	10	10	10	10	6.66	0
0	6.66	10	10	10	10	10	10	6.66	0
0	15.55	23.33	23.33	23.33	23.33	23.33	23.33	15.55	0
0	24.44	36.66	36.66	36.66	36.66	36.66	36.66	24.44	0
0	33.33	50	50	50	50	50	50	33.33	0
0	33.33	50	50	50	50	50	50	33.33	0
0	22.22	33.33	33.33	33.33	33.33	33.33	33.33	22.22	0
	0	0	0	0	0	0	0	0	0

$$\frac{1}{9}\begin{bmatrix} 50 & 50 & 0 \\ 50 & 50 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Spatial Smoothing (LPF Operations): Example

Neighborhood Pixels Processing

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	23.33	23.33	23.33	23.33	23.33	23.33	10
50	36.66	36.66	36.66	36.66	36.66	36.66	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

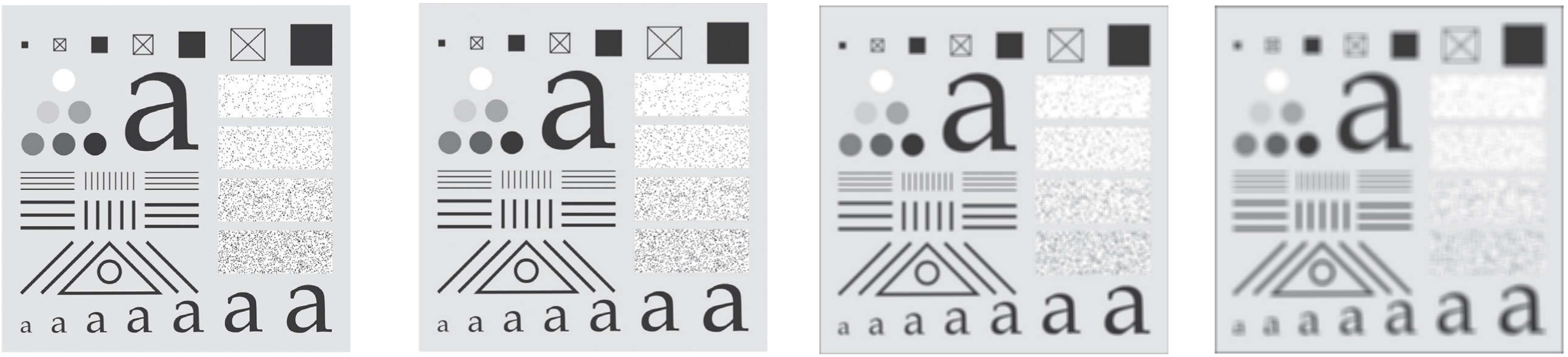
Retaining the boundary pixels and updating the edge pixels blurs the edge (reduces the difference)

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

- In the resultant image the Low frequency region has remained unchanged.
- Sharp transition between 10 & 50 has changed from 10 to 23.33 to 36.66 and finally to 50.
- Thus, Sharp edges has become blurred.
- Best result when used over image corrupted by Gaussian noise.

LP Filtering with a Box Kernel

- Figure below shows a test pattern image of size 1024 × 1024 pixels and the results obtained using box filters of size × with = 3, 11, and 21, respectively.

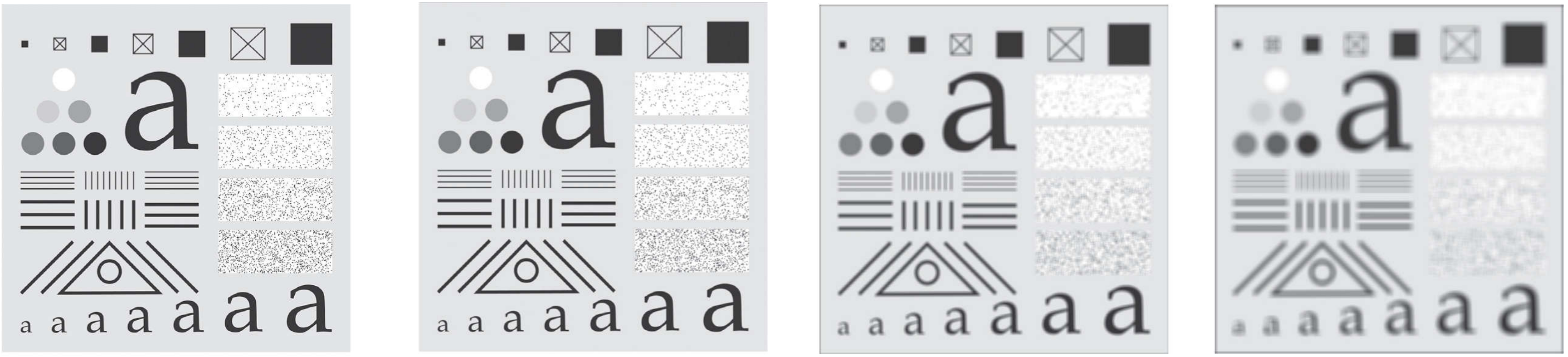


Test pattern of size 1024 × 1024 pixels.

- Note a slight overall blurring of the image, with the image features whose sizes are comparable to the size of the kernel being affected significantly more.
  - Such features include the thinner lines in the image and the noise pixels contained in the boxes on the right side of the image.
- **The filtered image also has a thin gray border, the result of zero-padding the image prior to filtering**

LP Filtering with a Box Kernel

- Figure below shows a test pattern image of size 1024 × 1024 pixels and the results obtained using box filters of size × with = 3, 11, and 21, respectively.



Test pattern of size 1024 × 1024 pixels.

- Using the 11 × 11 kernel resulted in more pronounced blurring throughout the image, including a prominent dark border.
- The result with the 21 × 21 kernel shows significant blurring of all components of the image,
  - the loss of the characteristic shape of some components
  - the small square on the top left and the small character on the bottom left.
- **The dark border resulting from zero padding is proportionally thicker than before.**

Limitations of Box Filters

- Box filters are suitable for quick experimentation (simplicity) and they often yield visually acceptable smoothing results
- They are useful also when it is desired to reduce the effect of smoothing on edges

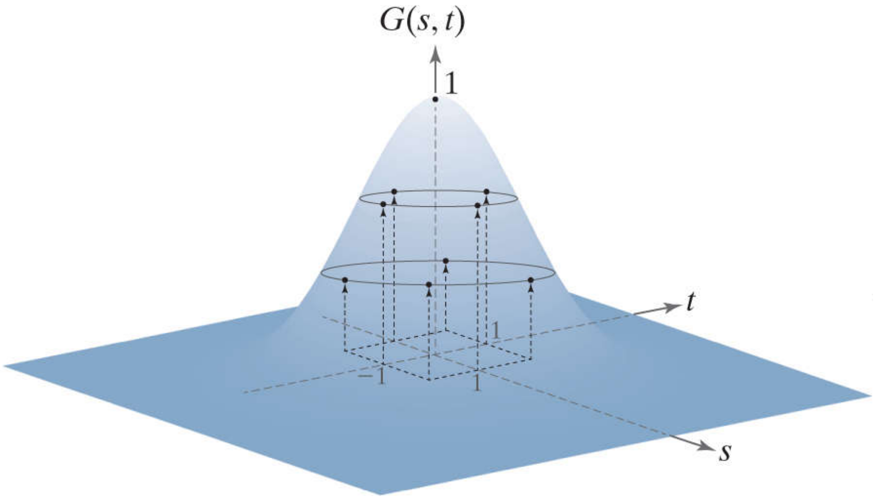
Limitations:

- Box filters favor blurring along perpendicular directions
  - In applications involving images with a high level of detail, or with strong geometrical components, the directionality of box filters often produces undesirable results

Gaussian Kernel (Separable)

- **Circularly symmetric (also called isotropic, (their response is independent of orientation) kernels handle the limitations of box kernels**
- Gaussian kernels are the only circularly symmetric kernels that are also separable
  - Gaussian filters enjoy the same computational advantages as box filters, but have a host of additional properties that make them ideal for image processing,

$$w(s,t) = G(s,t) = Ke^{-\frac{s^2+t^2}{2\sigma^2}}$$



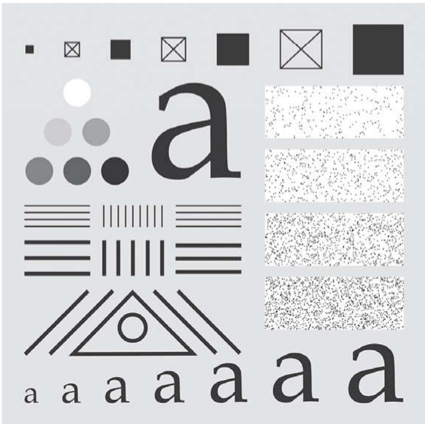
$\frac{1}{4.8976} \times$

0.3679	0.6065	0.3679
0.6065	1.0000	0.6065
0.3679	0.6065	0.3679

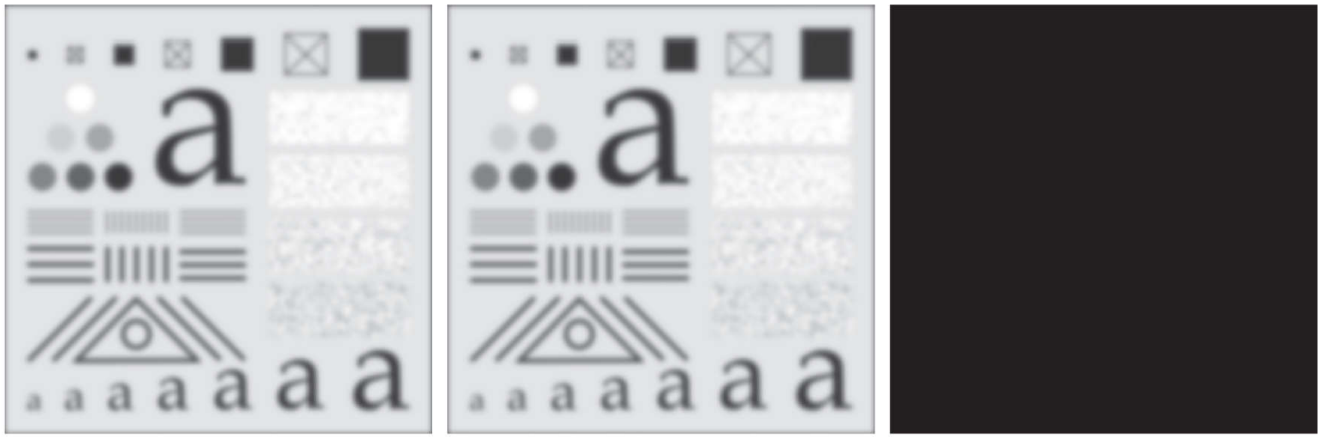
Sampling a Gaussian function to obtain a discrete Gaussian kernel. The values shown are for k = 1 and σ = 1.

LP Filtering with Gaussian Kernel

- Use a size equal to the closest odd integer to $\lceil 6\sigma \rceil \times \lceil 6\sigma \rceil$ where σ controls the "spread" of a Gaussian function about its mean (larger than this does not give better results)



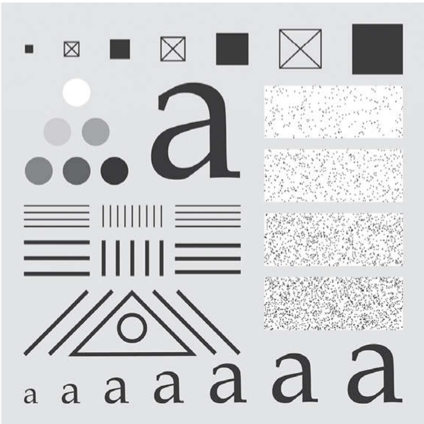
Test pattern of size 1024×1024



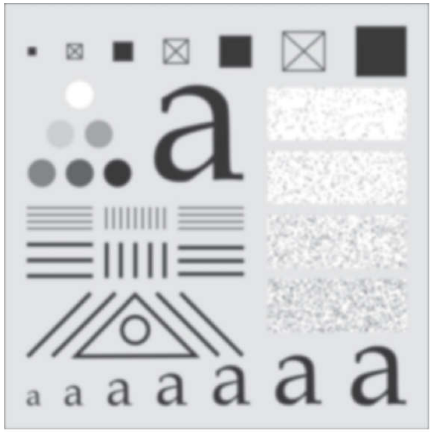
(a) Result of filtering using a Gaussian kernels of size 43×43, with $\sigma = 7$. (b) Result of using a kernel of 85×85, with the same value of σ (c) Difference image.

LP Filtering with a Box and Gaussian Kernel

- Compare Gaussian and box kernel filtering
- Figure below shows the result of lowpass filtering the test pattern with Gaussian and box kernel.

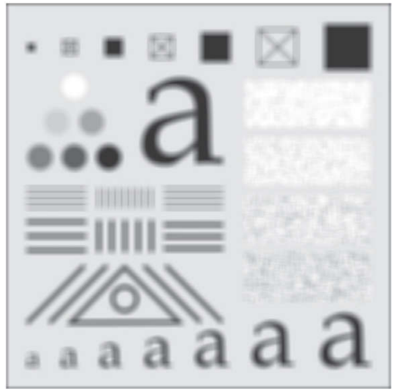


test pattern of size 1024 × 1024



Result of lowpass filtering with a Gaussian kernel of size 21 × 21, with standard deviations σ = 3.5.

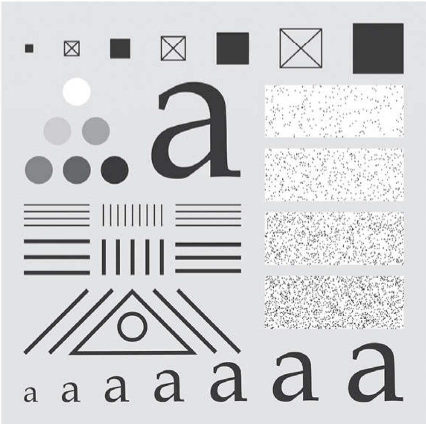
using a Gaussian kernel of size 43 × 43, with σ = 7.



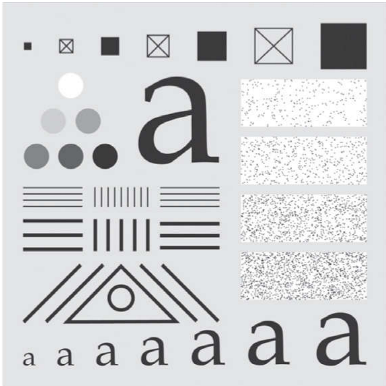
using a box kernel of size 21 × 21

- Gaussian kernels have to be larger than box filters to achieve the same degree of blurring.
  - This is because, whereas a box kernel assigns the same weight to all pixels, the values of Gaussian kernel coefficients (and hence their effect) decreases as a function of distance from the kernel center.

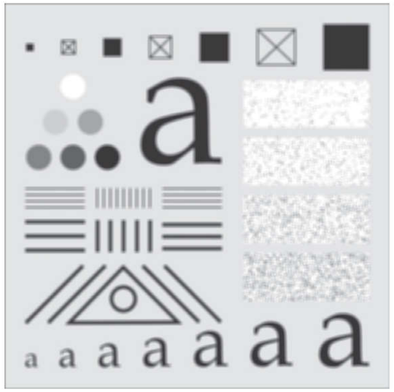
LP Filtering with a Box and Gaussian Kernel



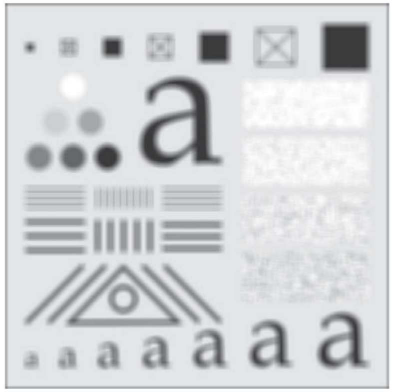
test pattern of size 1024 × 1024



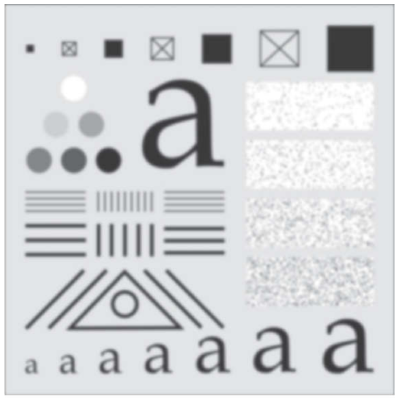
Using a box kernel of size 3 × 3



Using a box kernel of size 11 x11



Using a box kernel of size 21 × 21



Using a Gaussian kernel of size 21 × 21

using a kernel of size 43 × 43

Gaussian Kernel of size 43 x 43 gives comparable results as box kernel of size 21 x 21 but edges are smoother in latter

Box and Gaussian Filter Smoothing Characteristics

- Consider an image of a rectangle that was smoothed using a box and a Gaussian kernel

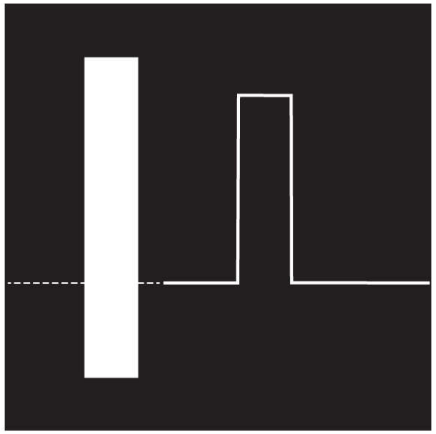
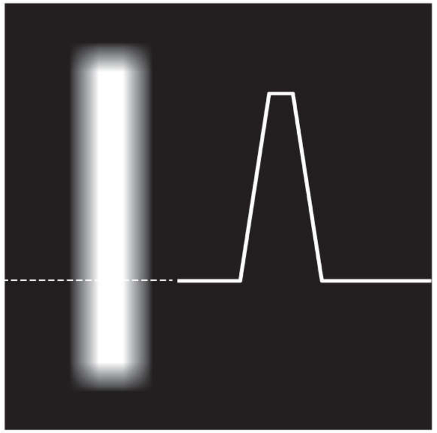
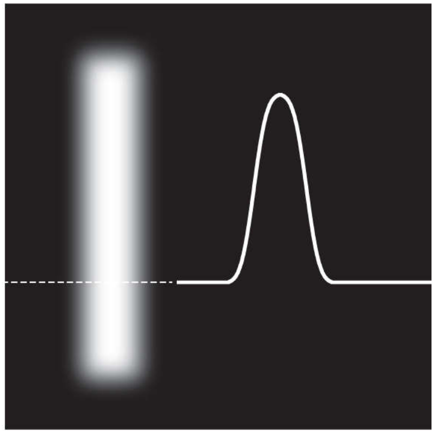


Image of a white rectangle on a black background, and a horizontal intensity profile along the scan line shown dotted. The image and rectangle are of sizes 1024 × 1024 and 768 × 128 pixels, respectively.



Result of smoothing this image with a box kernel of size 71 × 71, and corresponding intensity profile.



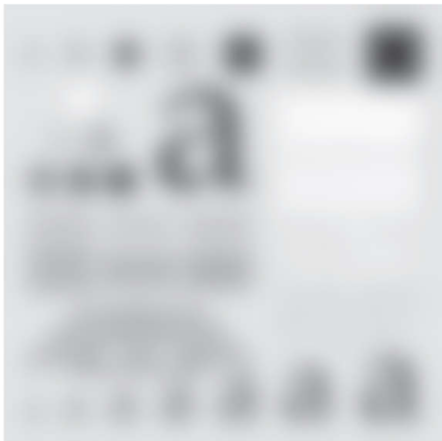
Result of smoothing the image using a Gaussian kernel of size 151 × 151, with K = 1 and σ = 25

From intensity profiles: the box filter produced linear smoothing, with the transition from black to white (i.e., at an edge) having the shape of a ramp. The important features here are hard transitions at the onset and end of the ramp. **We would use this type of filter when less smoothing of edges is desired.**
Conversely, the Gaussian filter yielded significantly smoother results around the edge transitions. **We would use this type of filter when generally uniform smoothing is desired.**

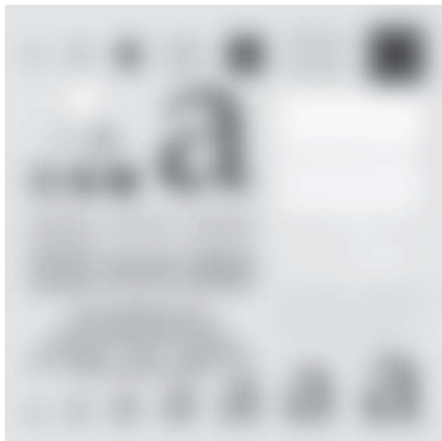
Padding Effects

- Zero padding introduces dark borders in the filtered result, with the thickness of the borders depending on the size and type of the filter kernel used
- Methods of image padding:
  - Mirror (also called symmetric) padding, in which values outside the boundary of the image are obtained by mirror-reflecting the image across its border
  - Replicate padding, in which values outside the boundary are set equal to the nearest image border value.
- The latter padding is useful when the areas near the border of the image are constant.
- Conversely, mirror padding is more applicable when the areas near the border contain image details. In other words, these two types of padding attempt to "extend" the characteristics of an image past its borders.

Result of filtering using zero padding



Using mirror padding



Using replicate padding

mirror and replicate padding yield more visually appealing results by eliminating the dark borders resulting from zero padding.

Types of Spatial Filters

- **Spatial Filters**
  - Convolution based filters
  - **Order-statistics(rank) filters**
  - Hybrid filters

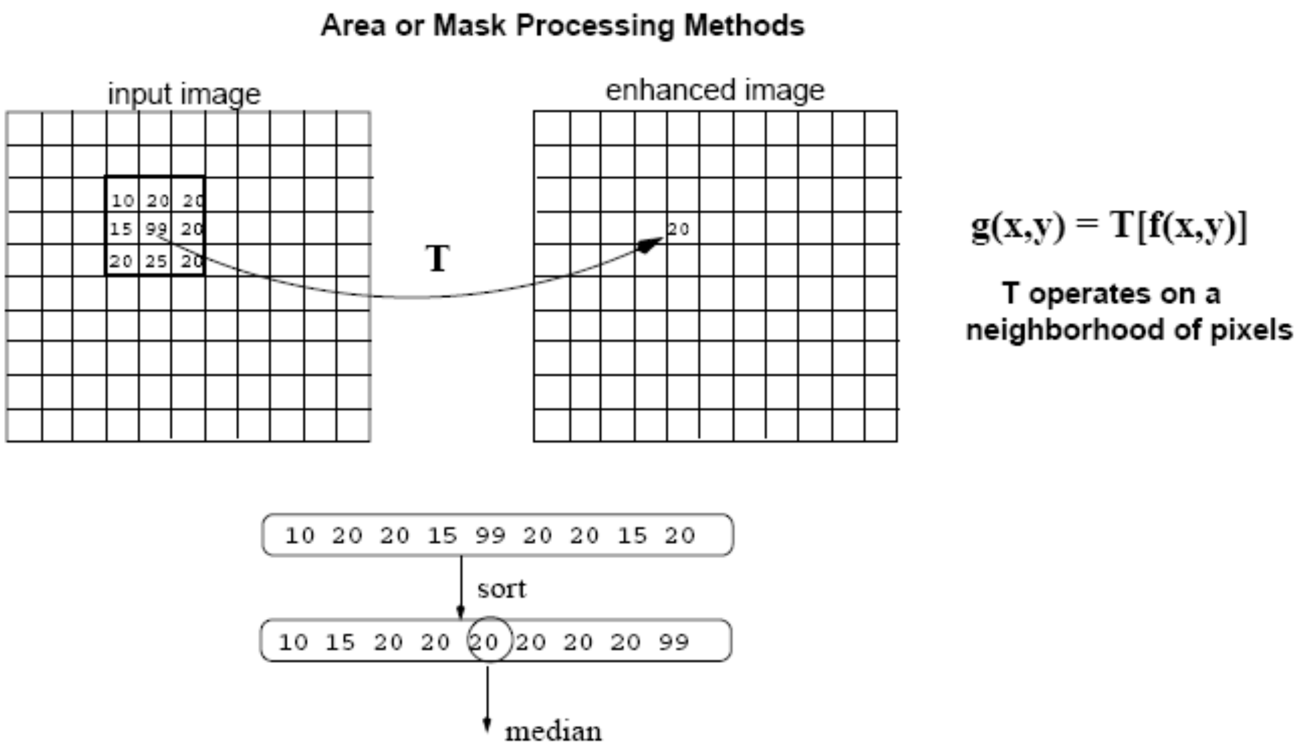
Order Statistics / Median Filters

- Here the input pixel is replaced by the median of the pixels contained in a window around the pixel
- First arrange the pixels in sorted order and pick the middle value
- Generally window size is odd. If window size is even then the median is the average of the two values in the middle.

$$v(m, n) = \text{median}\{y(m-k, n-l), (k, l) \in W\}$$

Order Statistics / Median Filters

- Replace each pixel by the median in a neighborhood around the pixel



Steps to perform Median Filtering:

1. Assume a 3x3 empty mask.
2. Place the empty mask at the Left Hand corner.
3. Arrange the 9 pixels in ascending or descending order.
4. Choose the median from these 9 values.
5. Place the median at the centre.
6. Move the mask in same manner as averaging filter and repeat till all pixels are updated

Order Statistics / Median Filter Example

- Compute the median value of the marked pixels shown using 3 x 3 mask

$$\begin{bmatrix} 18 & 22 & 33 & 25 & 32 & 24 \\ 34 & 128 & 24 & 172 & 26 & 23 \\ 22 & 19 & 32 & 31 & 28 & 26 \end{bmatrix}$$

- Ans:

$$\begin{bmatrix} 18 & 22 & 33 & 25 & 32 & 24 \\ 34 & 24 & 31 & 31 & 26 & 23 \\ 22 & 19 & 32 & 31 & 28 & 26 \end{bmatrix}$$

Median FIlter: Example

Neighborhood Pixels Processing

- ❑ Apply 3x3 median filter to find a new image.

3	4	2	3
1	7	3	2
4	5	3	8
2	3	1	7

Noisy Image S & P noise

3x3 blank mask

Median FIlter: Example

Neighborhood Pixels Processing

- Apply 3x3 median filter to find a new image.

3	4	2	3
1	7	3	2
4	5	3	8
2	3	1	7

3	4	2	3
1	3		2
4			8
2	3	1	7

1) 1 2 3 3 3 4 4 5 7

Median FIlter: Example

Neighborhood Pixels Processing

- Apply 3x3 median filter to find a new image.

3	4	2	3
1	7	3	2
4	5	3	8
2	3	1	7

3	4	2	3
1	3	3	2
4			8
2	3	1	7

1) 1 2 3 3 3 4 4 5 7
2) 2 2 3 3 3 4 5 7 8

Median FIlter: Example

Neighborhood Pixels Processing

- Apply 3x3 median filter to find a new image.

3	4	2	3
1	7	3	2
4	5	3	8
2	3	1	7

3	4	2	3
1	3	3	2
4	3		8
2	3	1	7

1) 1 2 3 3 3 4 4 5 7
2) 2 2 3 3 3 4 5 7 8
3) 1 1 2 3 3 3 4 5 7

Median FIlter: Example

Neighborhood Pixels Processing

- Apply 3x3 median filter to find a new image.

3	4	2	3
1	7	3	2
4	5	3	8
2	3	1	7

3	4	2	3
1	3	3	2
4	3	3	8
2	3	1	7

1) 1 2 3 3 3 4 4 5 7
2) 2 2 3 3 3 4 5 7 8
3) 1 1 2 3 3 3 4 5 7
4) 1 2 3 3 3 5 7 7 8

Low Pass Median Filtering (Non Linear)

- Averaging Filter removes the noise by blurring till it is no longer seen.
- It blurs the edges too.
- Bigger the averaging mask more the blurring.
- Sometimes the image contains ‘salt & pepper noise’.
- If averaging filter is used then it will remove the noise at the cost of ruined edges.
- **Thus a nonlinear filter like Median filter is required.**
- They are also called as order statistics filter since their response is based on ordering or ranking of pixels contained within the mask.
- Here we use a blank mask.

Median Filtering (Non Linear)

- Pixel values which are very different from their neighbors are replaced by a value equal to neighboring pixel value
- Capable of reducing salt (white) and pepper (black) noise

Median FIlter: Example

Neighborhood Pixels Processing

Remove the salt noise from the image shown below

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	250	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	250	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

8x8 Image

Median FIlter: Example

Neighborhood Pixels Processing

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	250	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	250	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

8x8 Image with blank mask

Median FIlter: Example

Neighborhood Pixels Processing

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	250	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	250	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

8x8 Image with blank mask

Median FIlter: Example

Neighborhood Pixels Processing

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	250	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	250	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	250	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

8x8 Image with blank mask

Median FIlter: Example

Neighborhood Pixels Processing

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	250	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

8x8 Image with blank mask

Median FIlter: Example

Neighborhood Pixels Processing

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	250	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

8x8 Image with blank mask

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	250	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	250	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

Median FIlter: Example

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	250	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	250	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

Neighborhood Pixels Processing

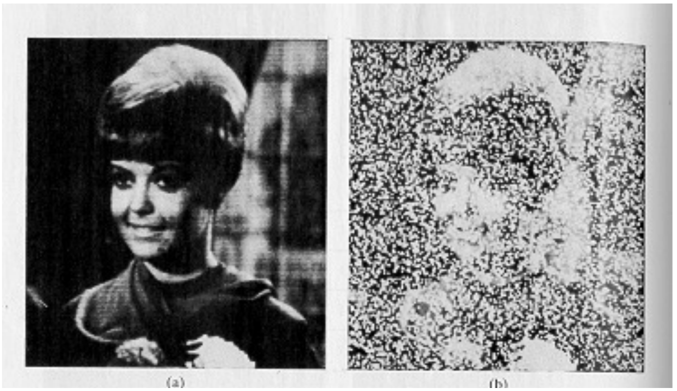
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

8x8 Image with blank mask

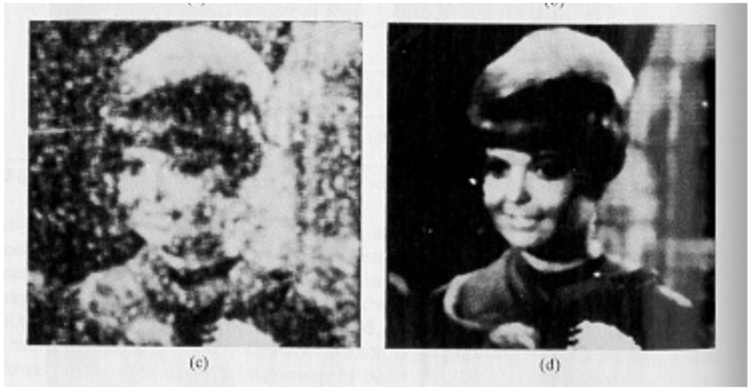
Effect of Smoothing and Median Filtering

- Very effective for removing “salt and pepper” noise (i.e., random occurrences of black and white pixels).

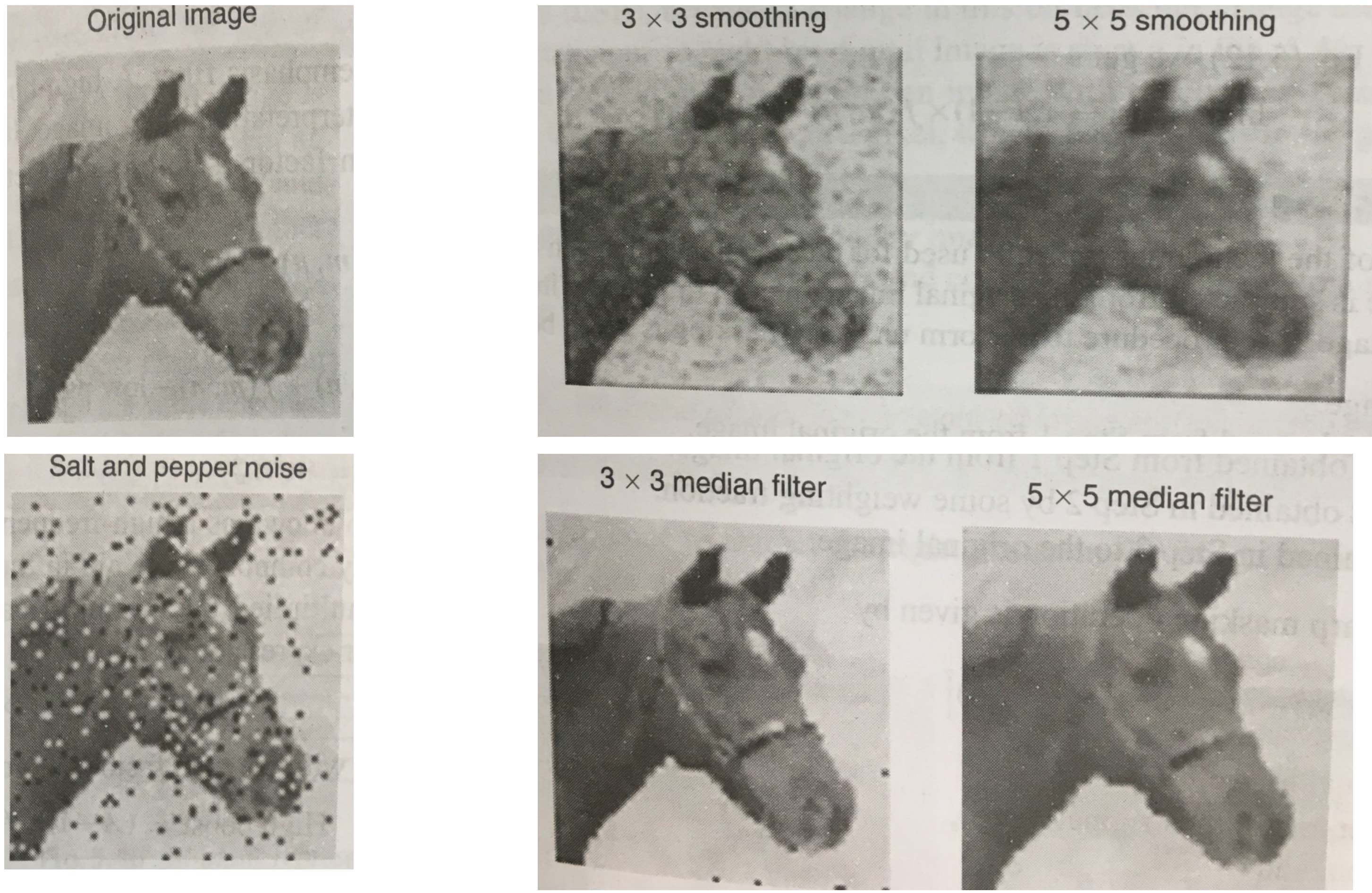
Original Image | **Image with salt & pepper noise**



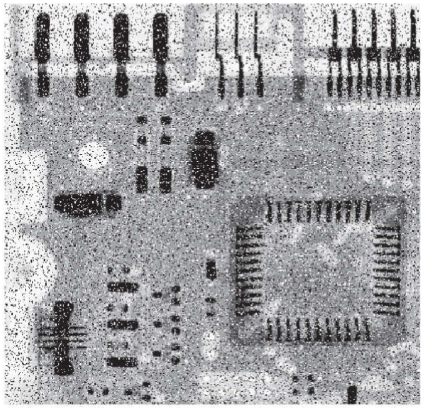
averaging | **median filtering**



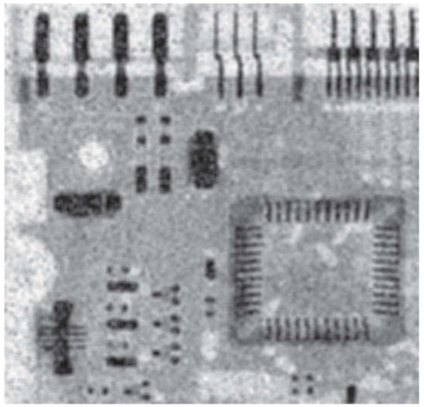
Effect of Smoothing and Median Filtering



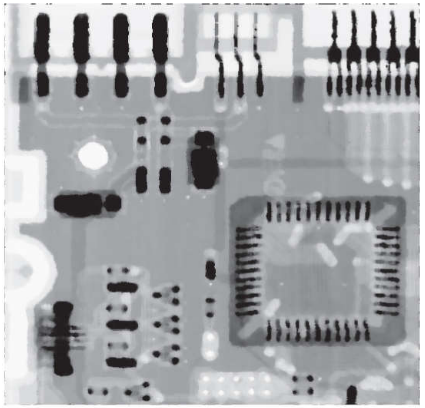
Effect of Smoothing and Median Filtering



X-ray image of a circuit board, corrupted by salt-and-pepper noise.



Noise reduction using a 19 × 19 Gaussian lowpass filter kernel with $\sigma = 3$.



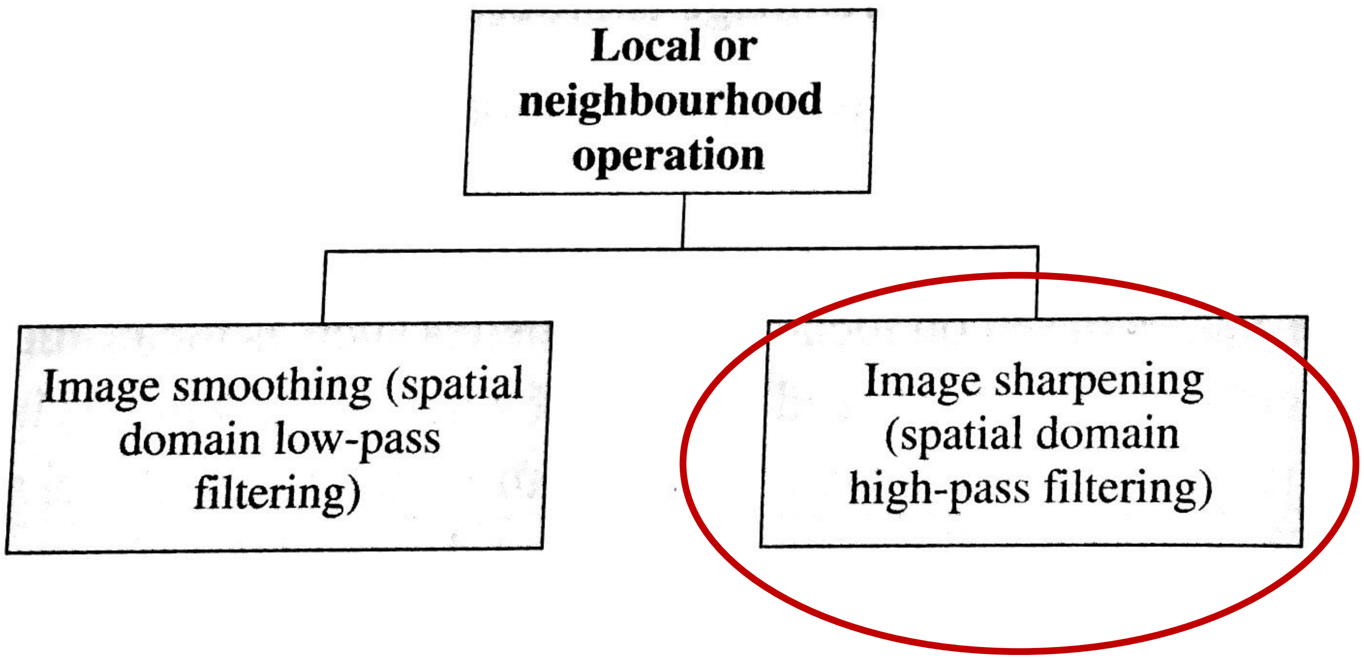
Noise reduction using a 7 × 7 median filter.

The lowpass filter blurred the image and its noise reduction performance was poor.
The superiority in all respects of median over lowpass filtering in this case is evident.

Types of Spatial Filters

- **Spatial Filters**
  - ✓Convolution based filters
  - ✓Order-statistics(rank) filters
  - Hybrid filters

Types of Local Operations



Next Session

- Sharpening using Laplacian operator
- Unsharp Masking

THANK YOU

Dr. Shikha Tripathi
Department of Electronics &
Communication Engineering

shikha@pes.edu
+91 9482219115