Digital Image Processing

Local Enhancement Through Spatial Filtering

Dr. Muhammad Sajjad

R.A: M.Abbas

Overview

- **Introduction to Spatial Filtering**
- **Spatial Filtering (Convolution)**
- **Example of Convolution Operation**
- **Padding in Spatial Filtering**
- **Convolution vs Correlation**
- **Smoothing Spatial Filters**
- **Sharpening Spatial Filters**

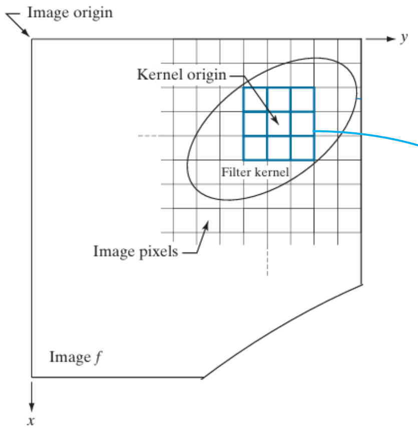
Introduction to Spatial Filtering

Spatial Filtering

- A technique used in image processing
- Each pixel’s value is modified based on the pixel itself and the values of its neighboring pixels.
- Used for image enhancement, noise reduction, and edge detection.

Filter (or Kernel, Mask, Template, Window):

- The kernel is an array whose size defines the neighborhood of operation, and whose coefficients determine the nature of the filter.
- We prefer odd-sized kernels (e.g., 3x3, 5x5, etc.).



Filter, Kernel, w(s, t)

W(-1, -1)	W(-1, 0)	W(-1, 1)
W(0, -1)	W(0, 0)	W(0, 1)
W(1, -1)	W(1, 0)	W(1, 1)

Kernel coefficients

f(x-1,y-1)	f(x-1,y)	f(x-1,y+1)
f(x,y-1)	f(x, y)	f(x,y+1)
f(x+1,y-1)	f(x+1,y)	f(x+1,y+1)

Pixel values under kernel when it is centered on (x, y)

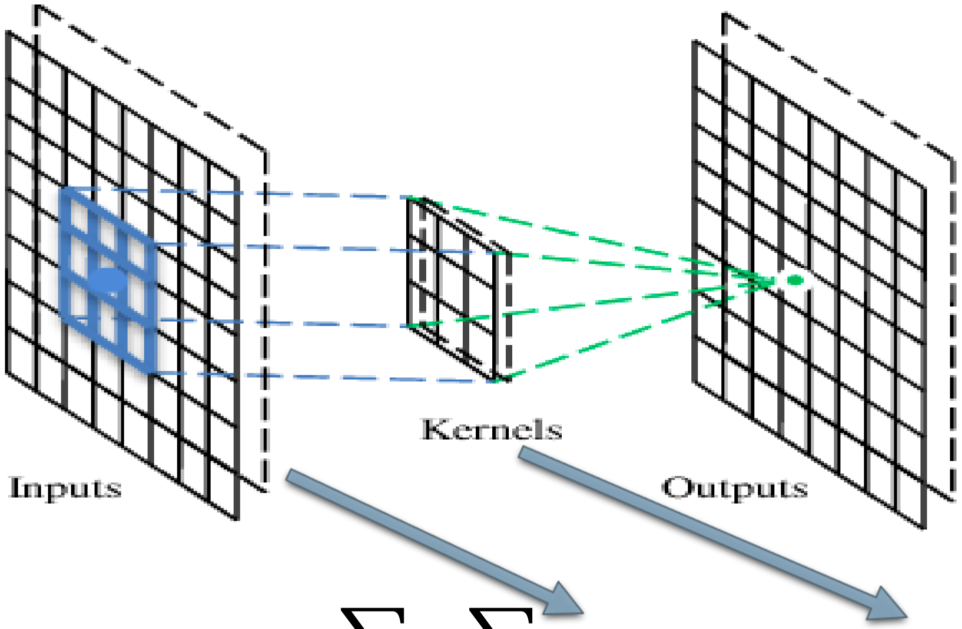
Noisy image

Result of Median Filter

Spatial Filtering (Convolution)

Linear Spatial Filtering

- Move the kernel across the image, pixel by pixel.
- Perform a sum-of-products operation between the local region of the image I and the filter kernel K.
- Place the computed value into the corresponding pixel in the output image.
- Examples: Averaging filter, Gaussian filter.



I is input Image
K kernel
m,n are filter indices

$$(I * K)(i,j) = \sum_m \sum_n I(i-m, j-n).K(m,n)$$

Non-Linear Filtering:

- Use operations other than the sum-of-products
- Example: selecting the median value in a neighborhood (e.g., Median filter)

4	0	0
4	1	2
3	0	1

Local region

*

1	0	-1
1	0	-1
1	0	-1

Filter

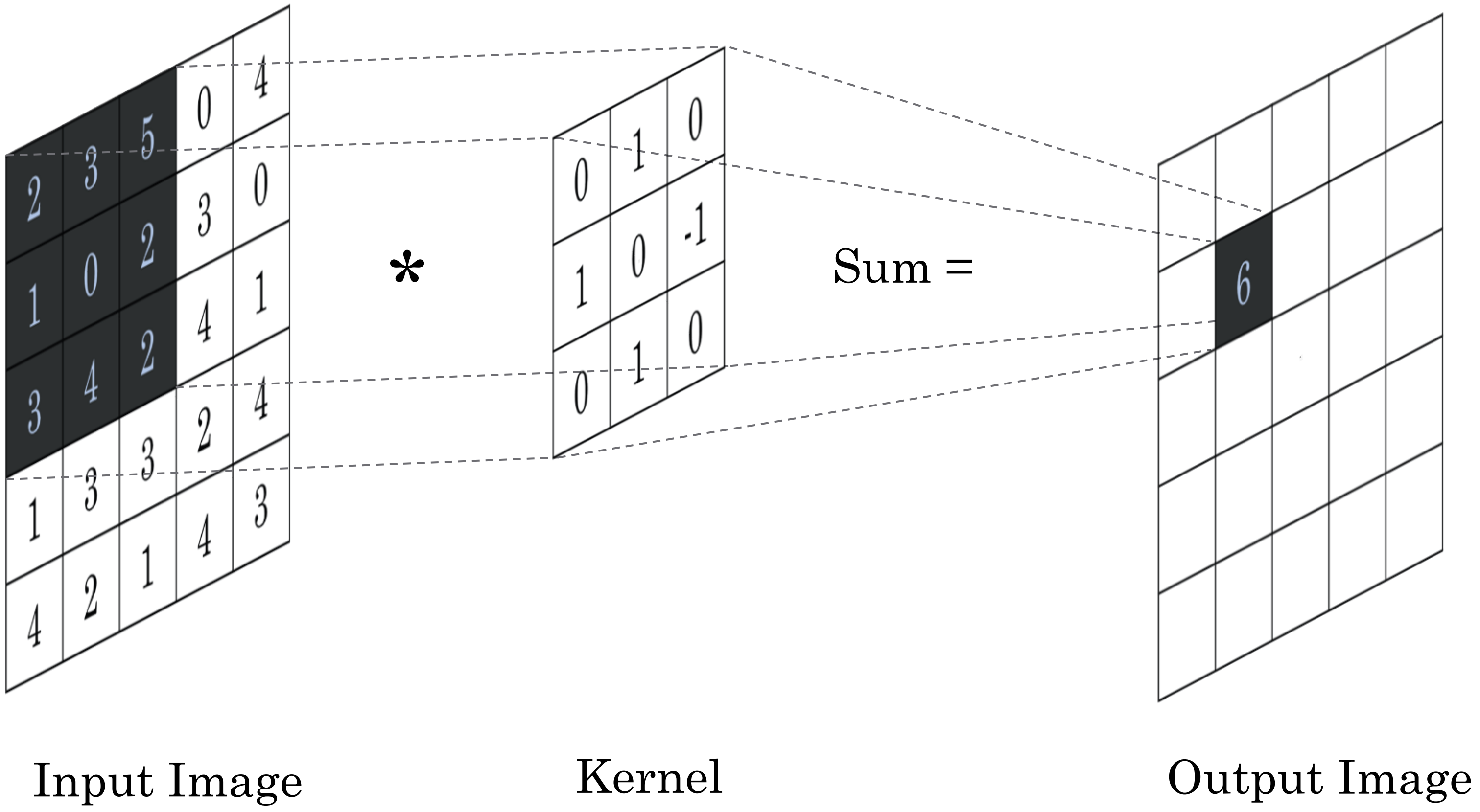
=

	9	

Example of Convolution Operation

$$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{1},\mathbf{1}) = (2*0) + (3*1) + (5*0) + (1*1) + (0*0) + (2*-1) + (3*0) + (4*1) + (2*0)$$

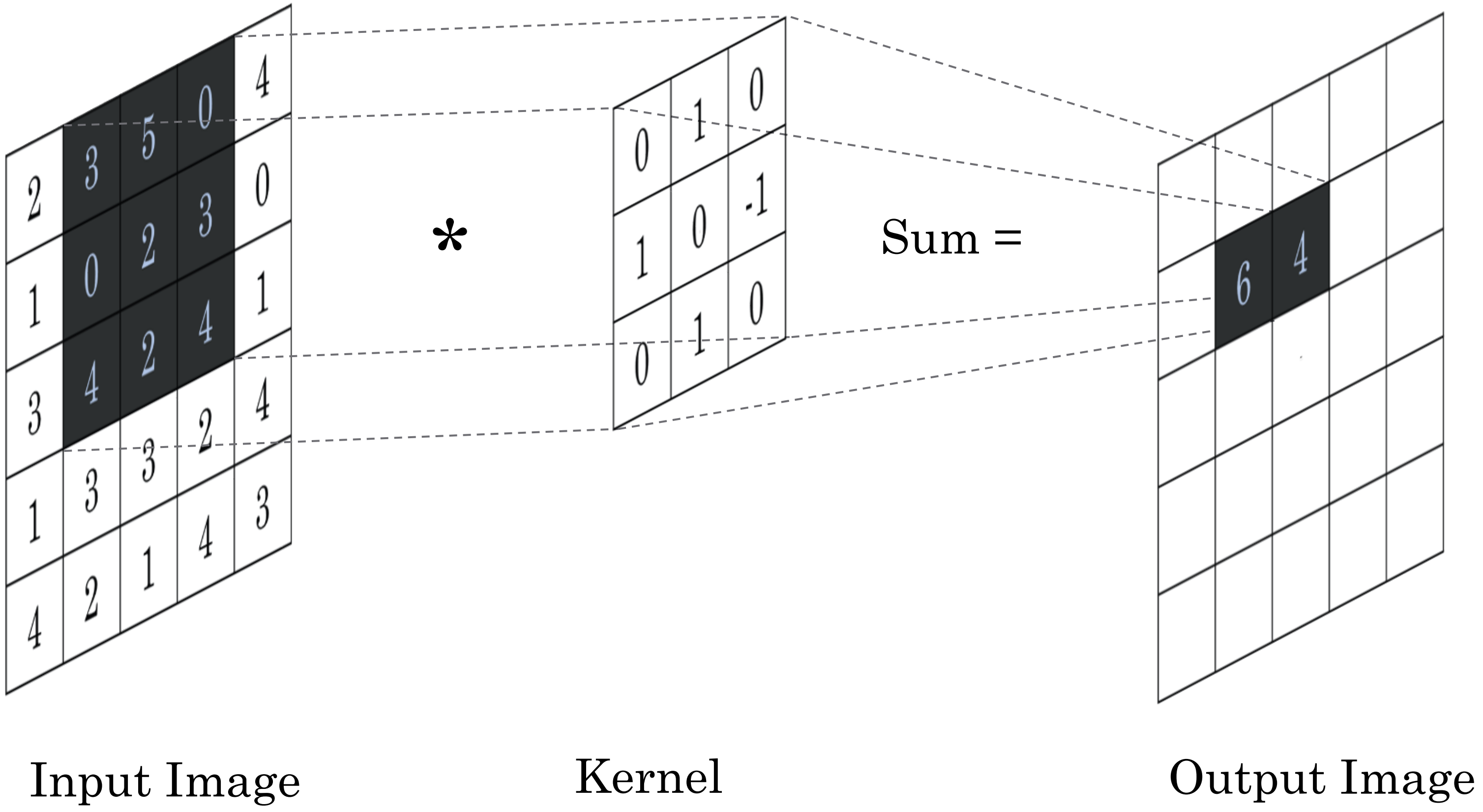
$$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{1},\mathbf{1}) = 6$$



Example of Convolution Operation

$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{1}, \mathbf{2}) = (3 * 0) + (5 * 1) + (0 * 0) + (0 * 1) + (2 * 0) + (3 * -1) + (4 * 0) + (2 * 1) + (4 * 0)$

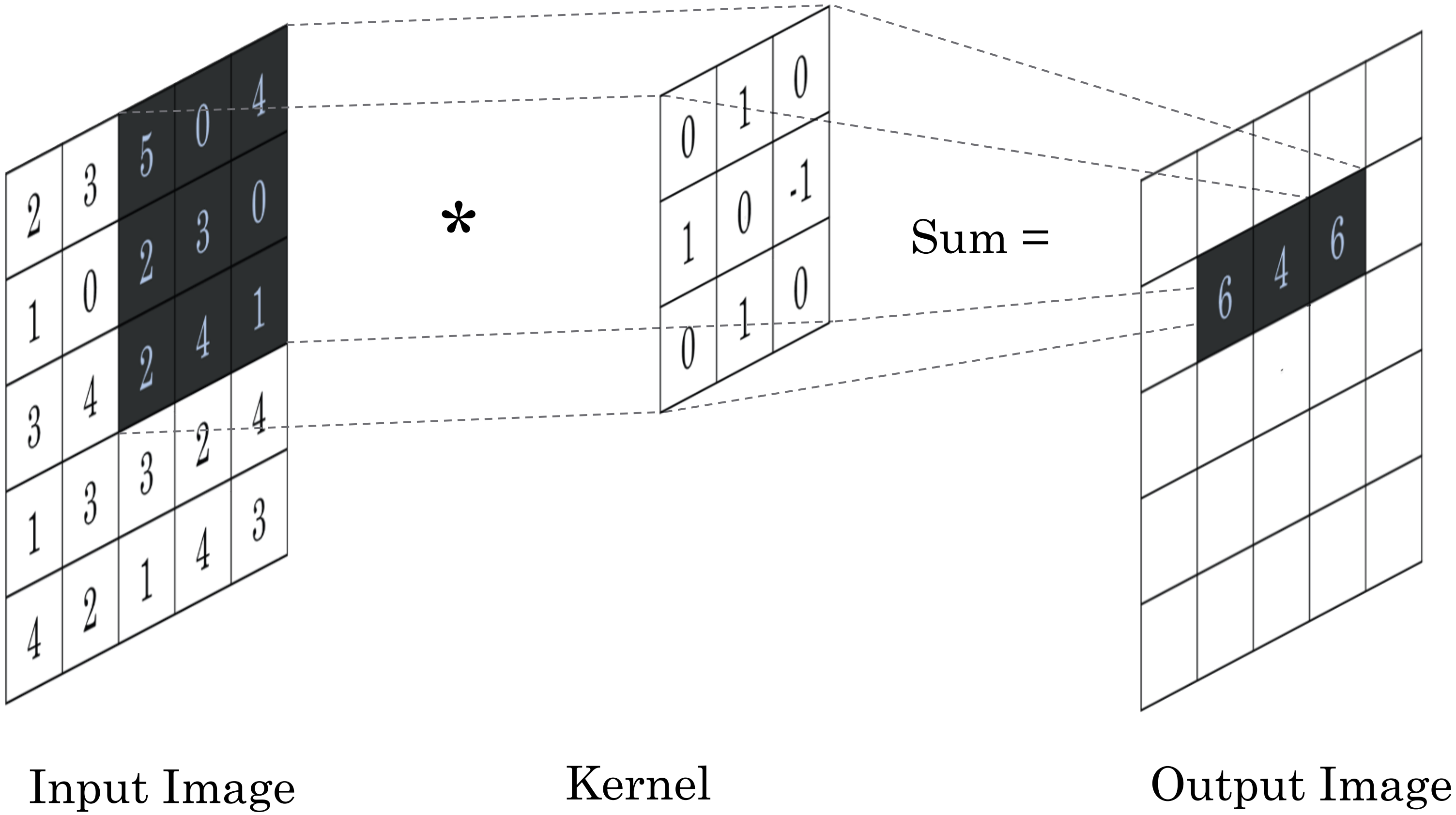
$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{1}, \mathbf{2}) = 4$



Example of Convolution Operation

$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{1},\mathbf{3}) = (5*0) + (0*1) + (4*0) + (2*1) + (3*0) + (0*-1) + (2*0) + (4*1) + (1*0)$

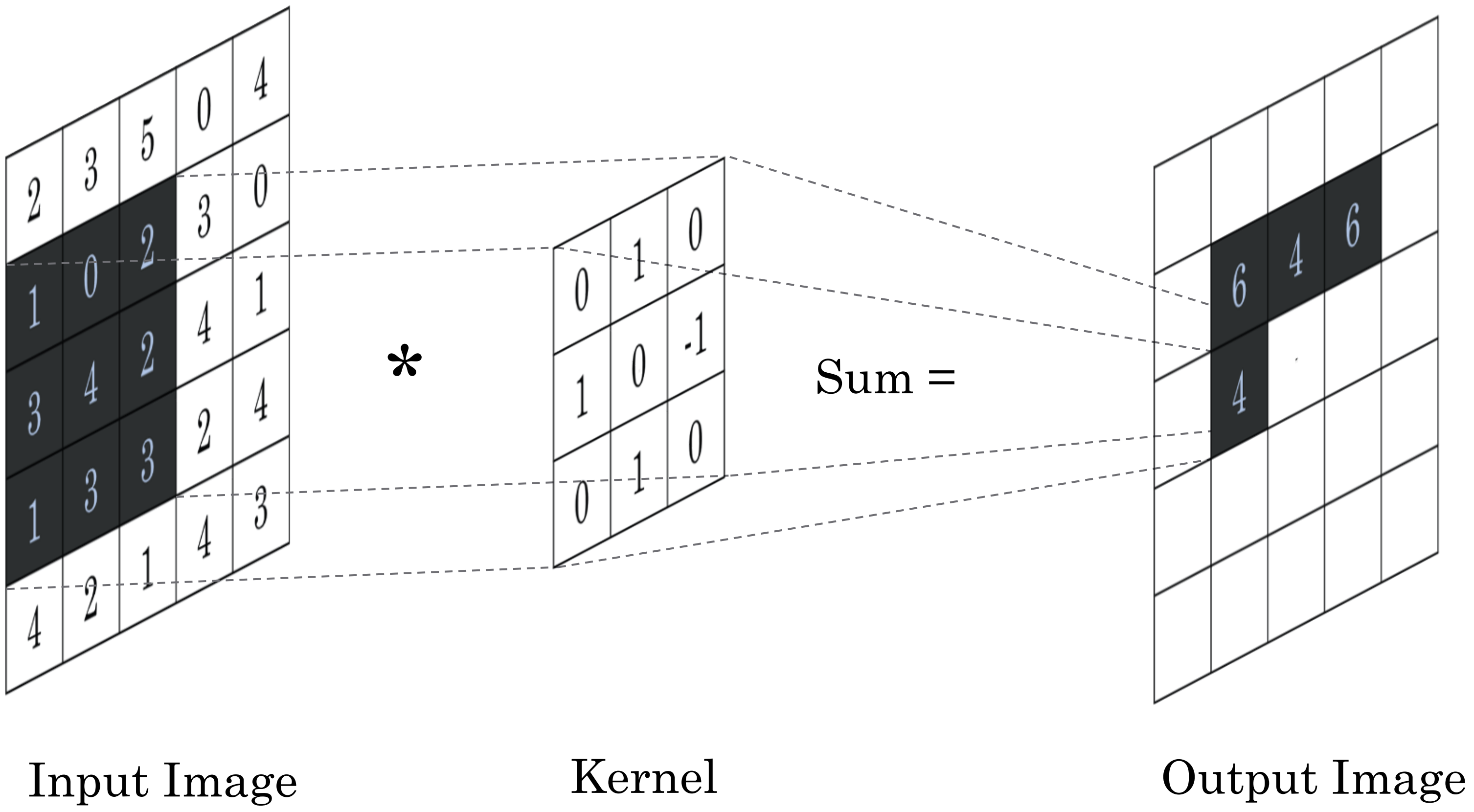
$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{1},\mathbf{3}) = 6$



Example of Convolution Operation

$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{2}, \mathbf{1}) = (1 * 0) + (0 * 1) + (2 * 0) + (3 * 1) + (4 * 0) + (2 * -1) + (1 * 0) + (3 * 1) + (3 * 0)$

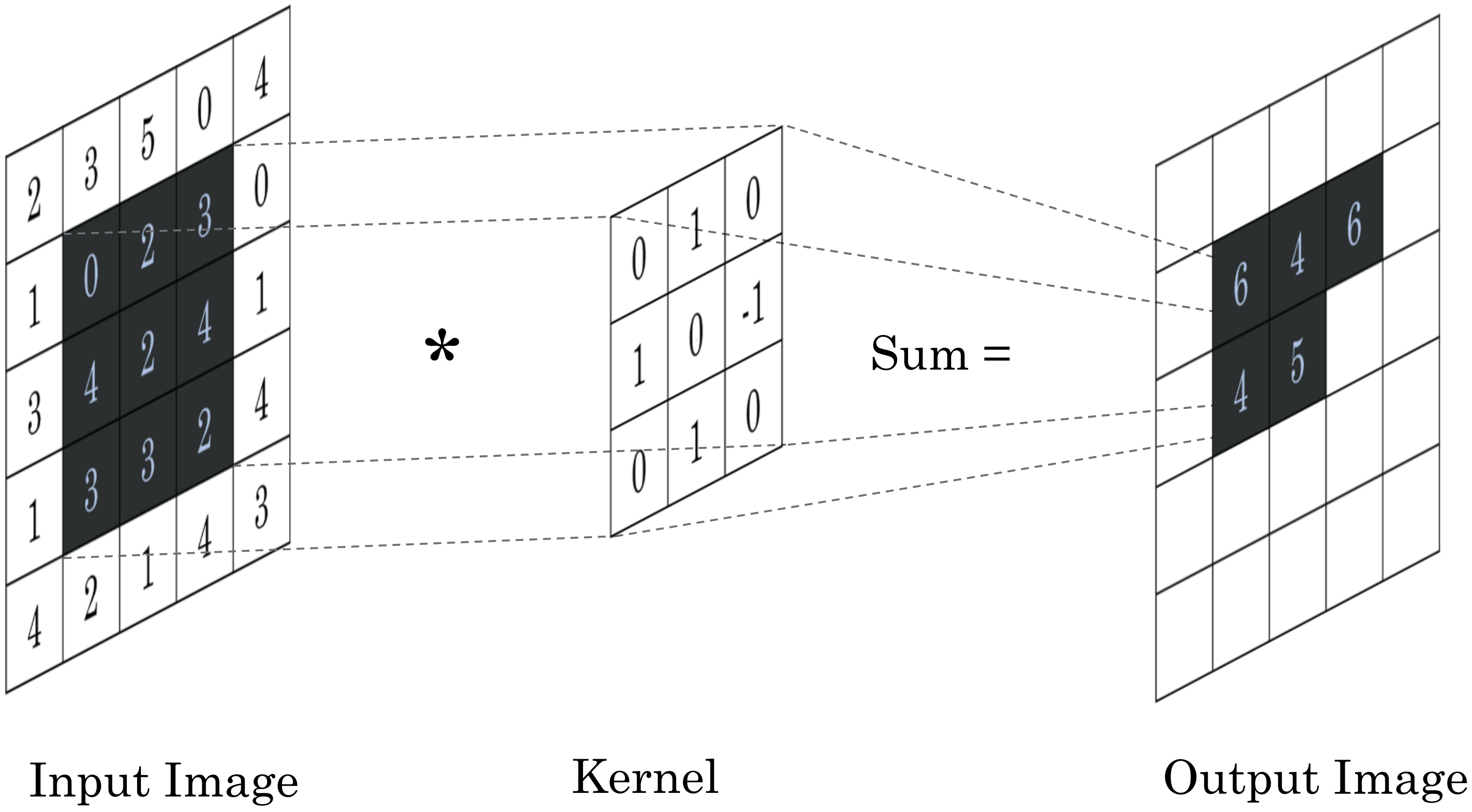
$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{2}, \mathbf{1}) = 4$



Example of Convolution Operation

$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{2}, \mathbf{2}) = (0 * 0) + (2 * 1) + (3 * 0) + (4 * 1) + (2 * 0) + (4 * -1) + (3 * 0) + (3 * 1) + (2 * 0)$

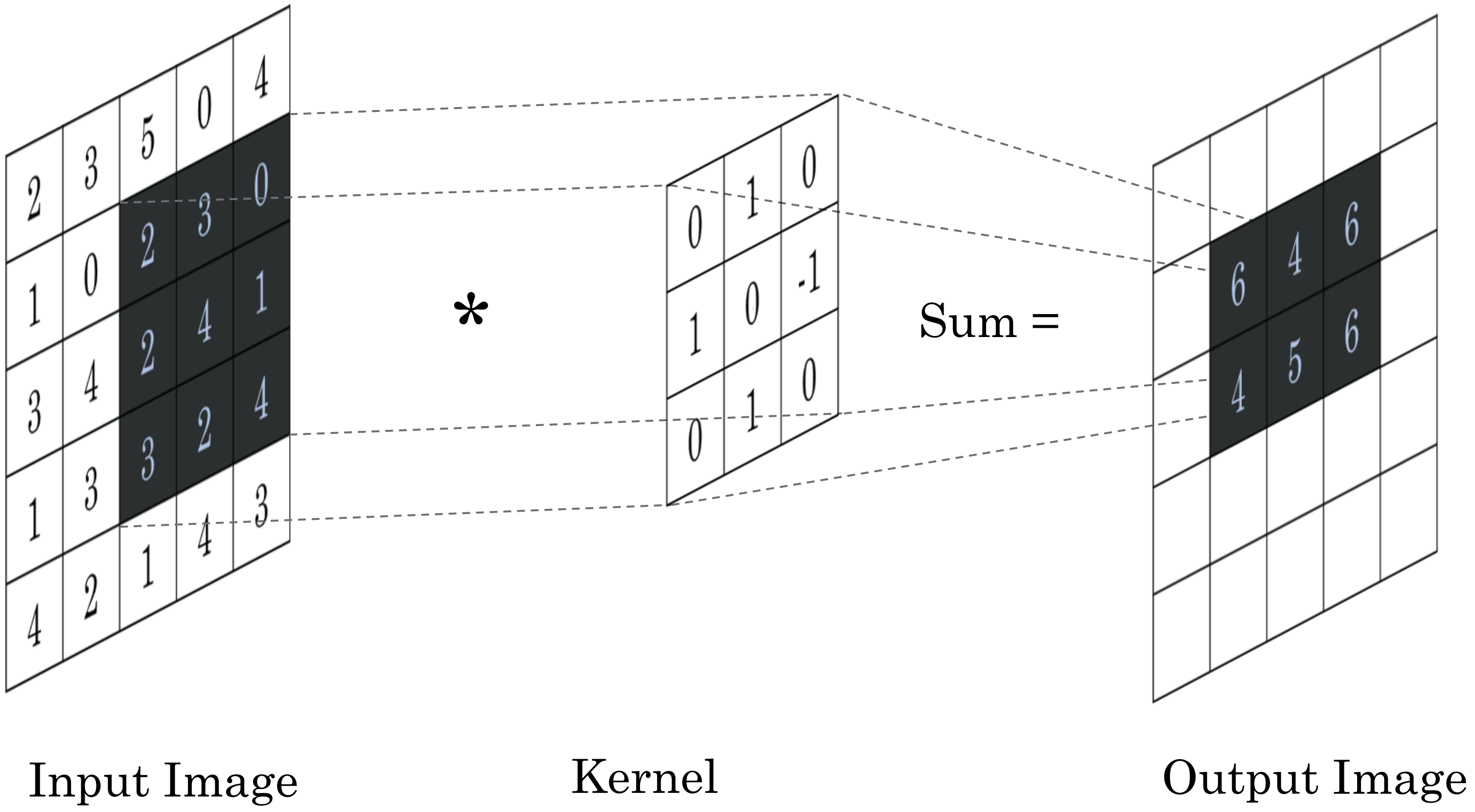
$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{2}, \mathbf{2}) = 5$



Example of Convolution Operation

$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{2}, \mathbf{3}) = (2 * 0) + (3 * 1) + (0 * 0) + (2 * 1) + (4 * 0) + (1 * -1) + (3 * 0) + (2 * 1) + (4 * 0)$

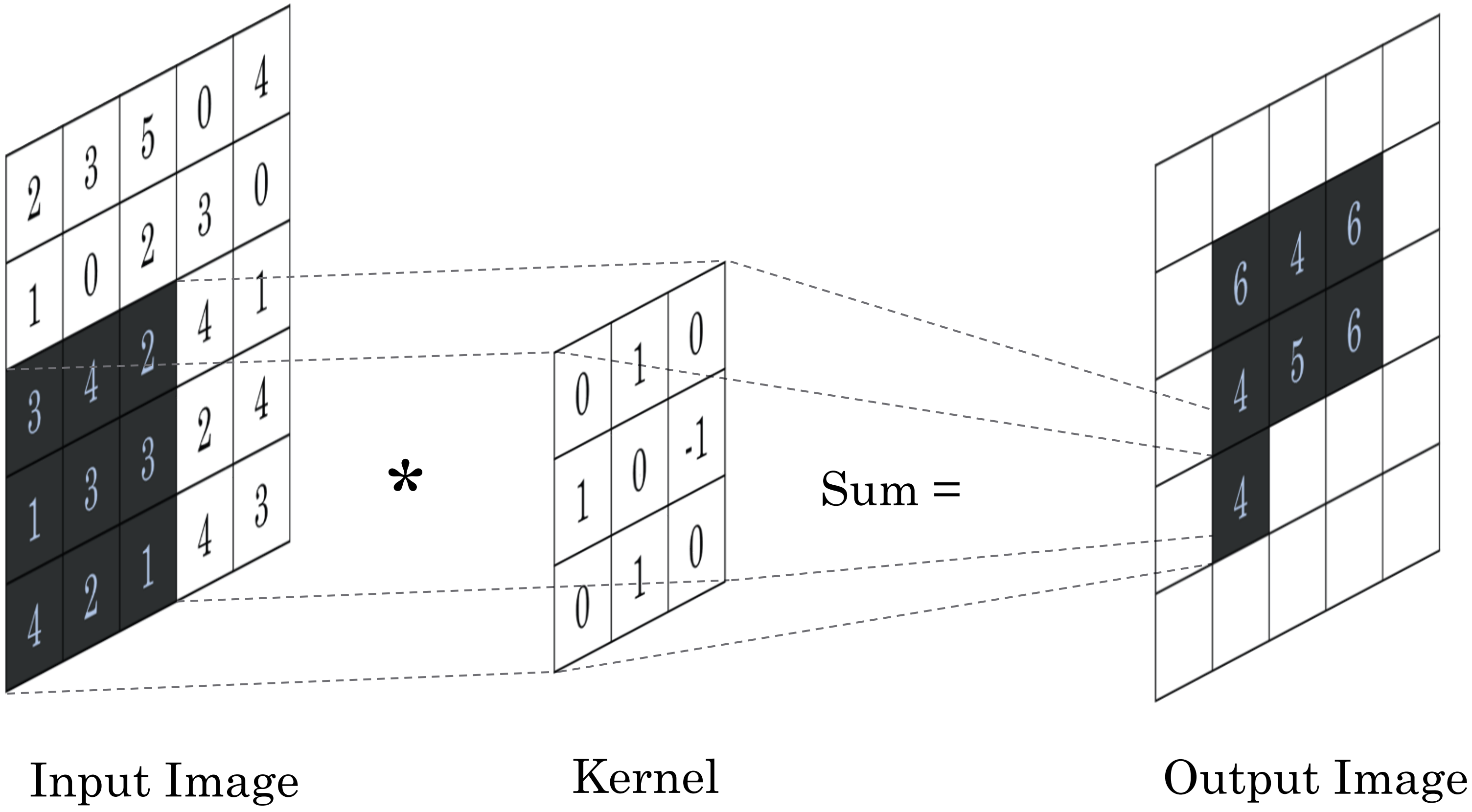
$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{2}, \mathbf{3}) = 6$



Example of Convolution Operation

$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{3}, \mathbf{1}) = (3 * 0) + (4 * 1) + (2 * 0) + (1 * 1) + (3 * 0) + (3 * -1) + (4 * 0) + (2 * 1) + (1 * 0)$

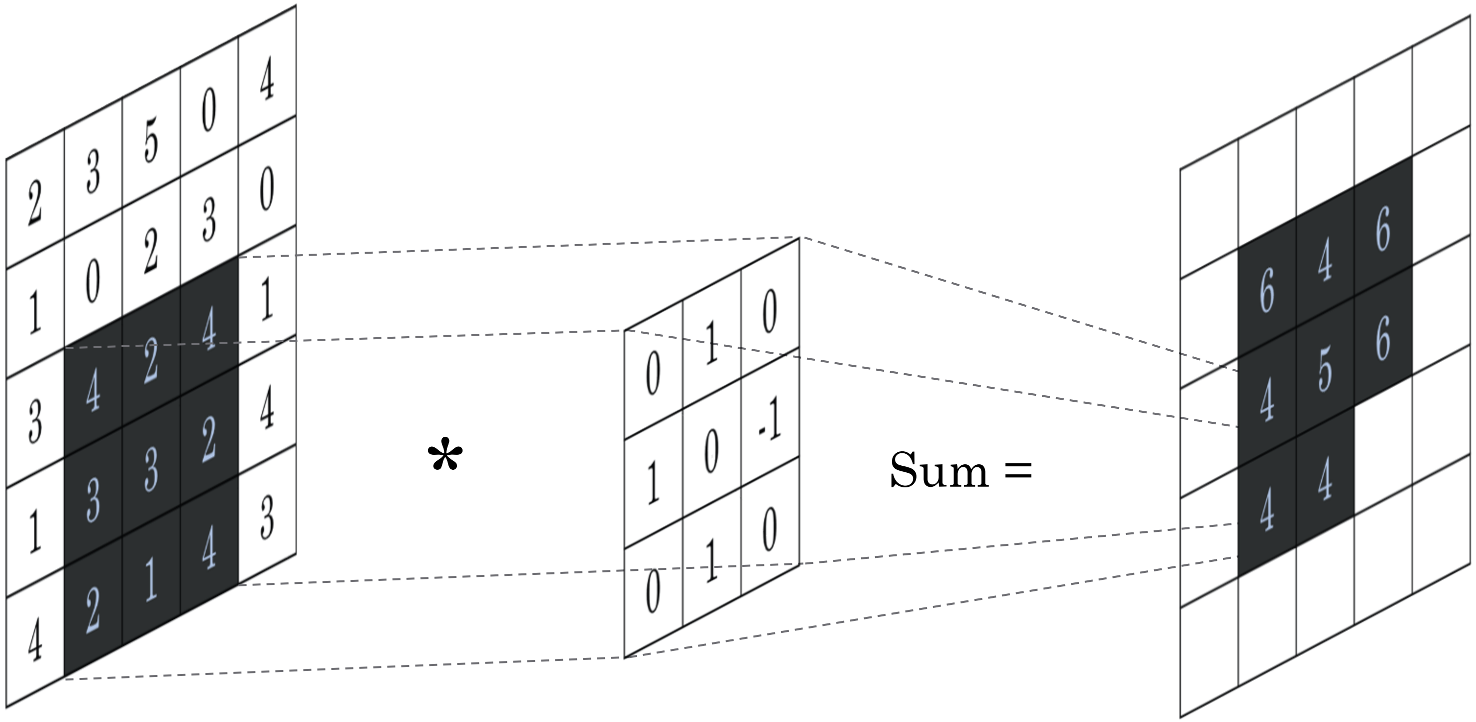
$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{3}, \mathbf{1}) = 4$



Example of Convolution Operation

$$(\boldsymbol{I} * \boldsymbol{K})(\boldsymbol{3}, \boldsymbol{2}) = (4 * 0) + (2 * 1) + (4 * 0) + (3 * 1) + (3 * 0) + (2 * -1) + (2 * 0) + (1 * 1) + (4 * 0)$$

$$(\boldsymbol{I} * \boldsymbol{K})(\boldsymbol{3}, \boldsymbol{2}) = 4$$



Input Image

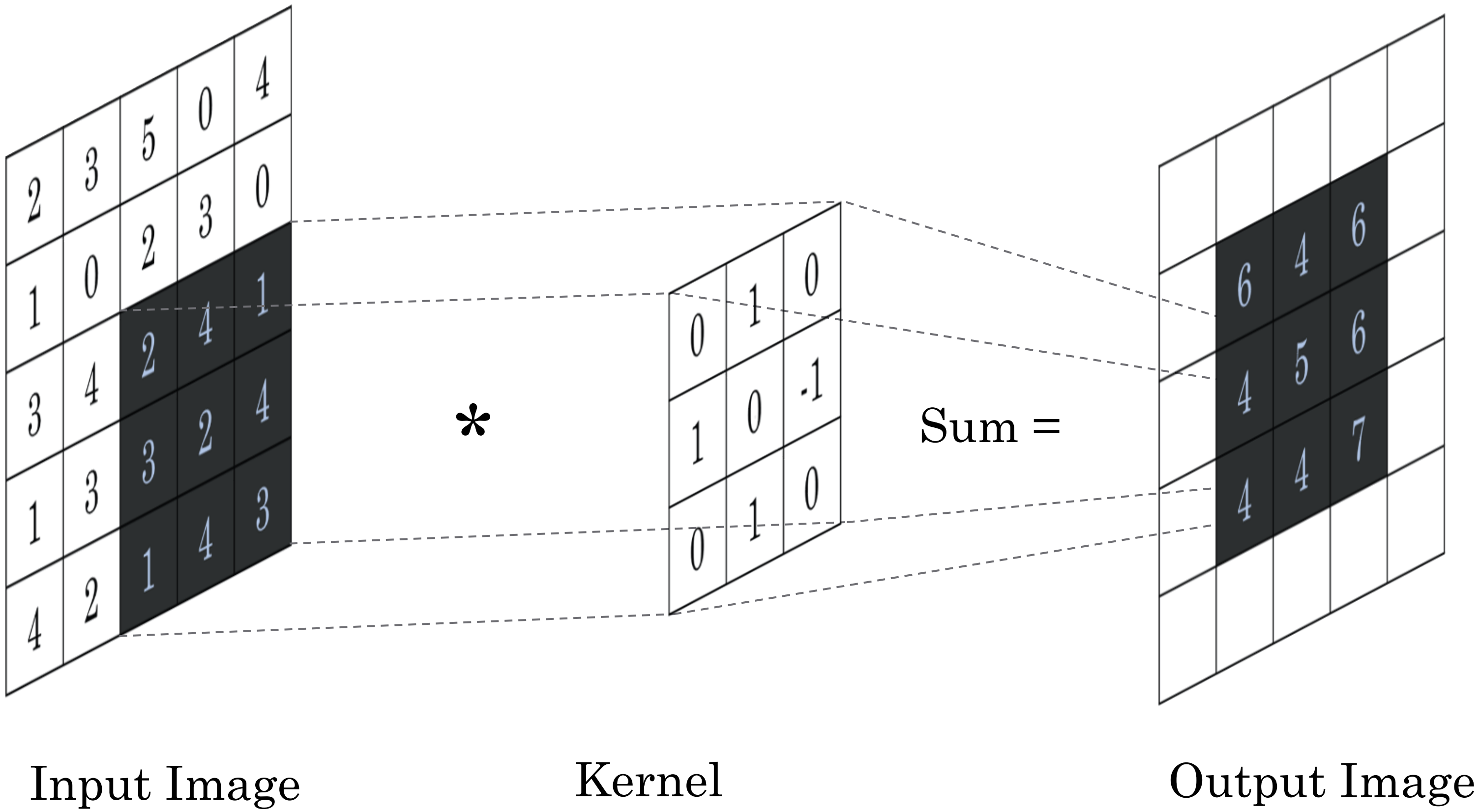
Kernel

Output Image

Example of Convolution Operation

$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{3}, \mathbf{3}) = (2 * 0) + (4 * 1) + (1 * 0) + (3 * 1) + (2 * 0) + (4 * -1) + (1 * 0) + (4 * 1) + (3 * 0)$

$(\boldsymbol{I} * \boldsymbol{K})(\mathbf{3}, \mathbf{3}) = 7$



Padding in Spatial Filtering

- Adding extra rows and columns around an image.
- Ensure that the filter can be applied evenly across all pixels.

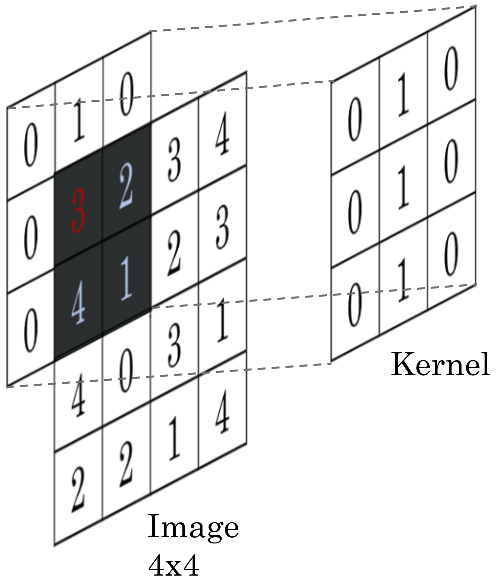
Zero Padding: Adds pixels with a value of zeros around the image.

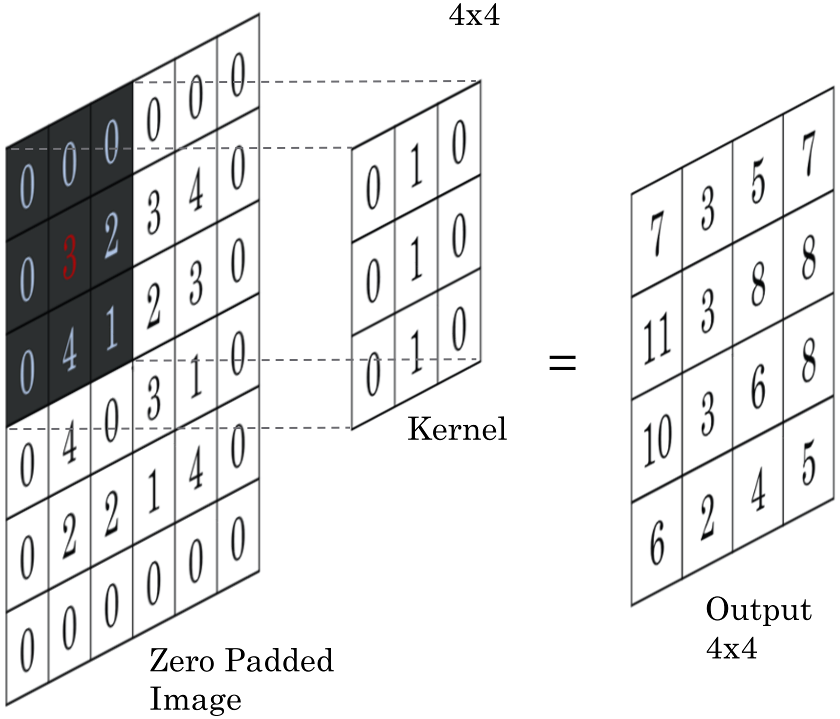
Replicate Padding: Replicates the border pixels of the image.

For a kernel of size m×n:

- Pad with **(m−1)/2,** rows of zeros (top and bottom).
- Pad with **(n−1)/2**, columns of zeros (left and right).

Part of w lies outside f





Convolution vs Correlation

Correlation

- Moving the center of a kernel over an image, and computing the sum of products at each location

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

Convolution

- Moving 180" rotated kernel over an image, and computing the sum of products at each location.

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

$a = (m-1)/2 \qquad b = (n-1)/2$

Origin f

M x N Image
$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$

w
$$\begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{matrix}$$
m x n Kernel

Initial position for w
$$\begin{matrix} 1 & 2 & 3 & 0 & 0 & 0 & 0 \\ 4 & 5 & 6 & 0 & 0 & 0 & 0 \\ 7 & 8 & 9 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} \Rightarrow$$

Correlation result
$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 9 & 8 & 7 & 0 \\ 0 & 6 & 5 & 4 & 0 \\ 0 & 3 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$

Rotated w
$$\begin{matrix} 9 & 8 & 7 & 0 & 0 & 0 & 0 \\ 6 & 5 & 4 & 0 & 0 & 0 & 0 \\ 3 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} \Rightarrow$$

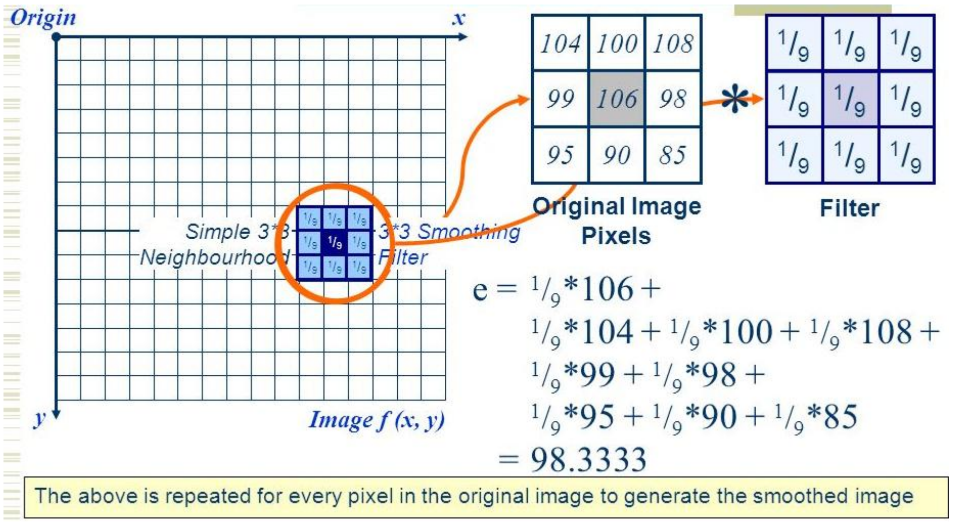
Convolution result
$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 \\ 0 & 4 & 5 & 6 & 0 \\ 0 & 7 & 8 & 9 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$

Smoothing Spatial Filters

Used for: Blurring and noise reduction.

Blurring helps to:

- Remove small details before object extraction.
- Bridge small gaps in lines or curves.



Averaging Linear Filters

Image M x N, Filter m x n

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

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

Box Filter
All coefficients are equal

$$\frac{1}{16} * \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

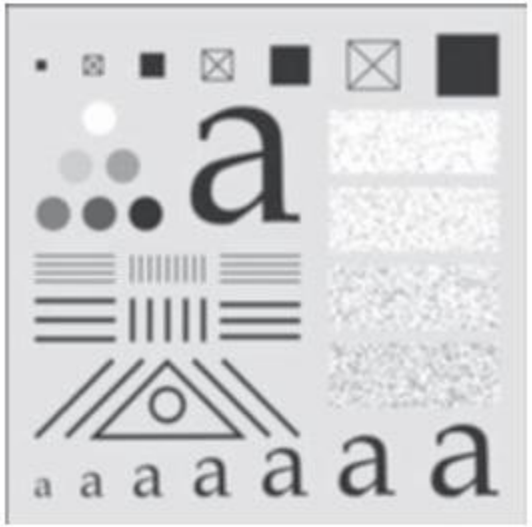
Weighted Average
Five more (less) weight to pixels near (away from) the output location

Smoothing Spatial Filters

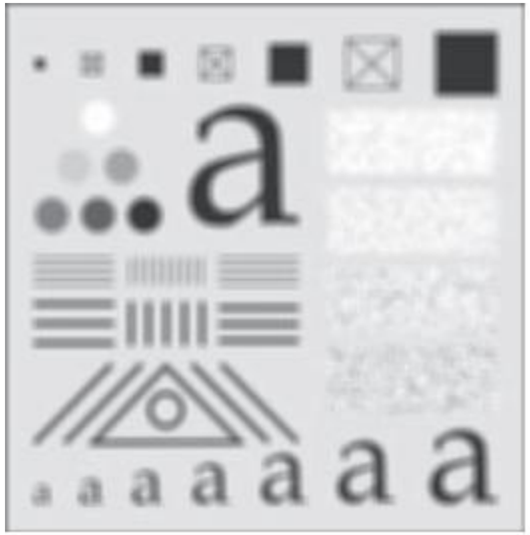
Example of box kernel

Original
Image

3 x 3 box
kernel result



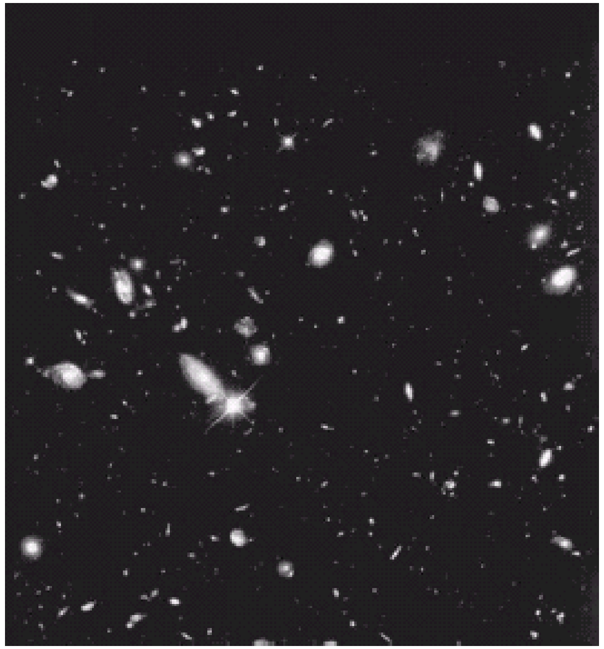
11 x 11 box
kernel result



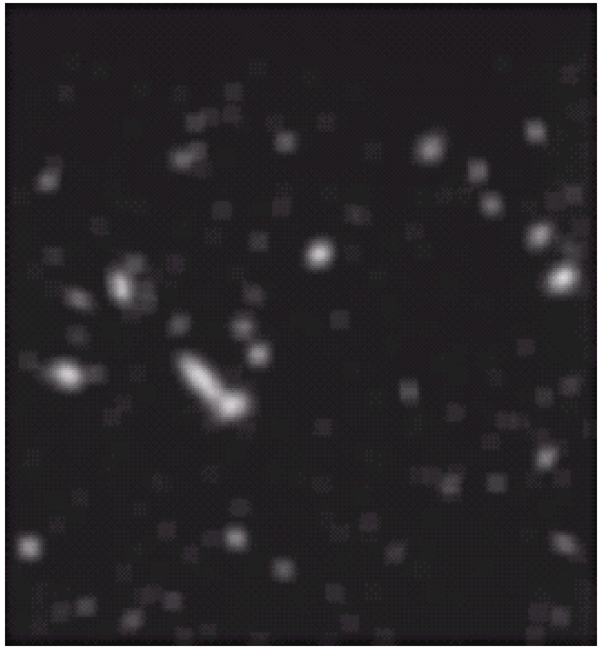
21 x 21 box
kernel result

Smoothing Spatial Filters

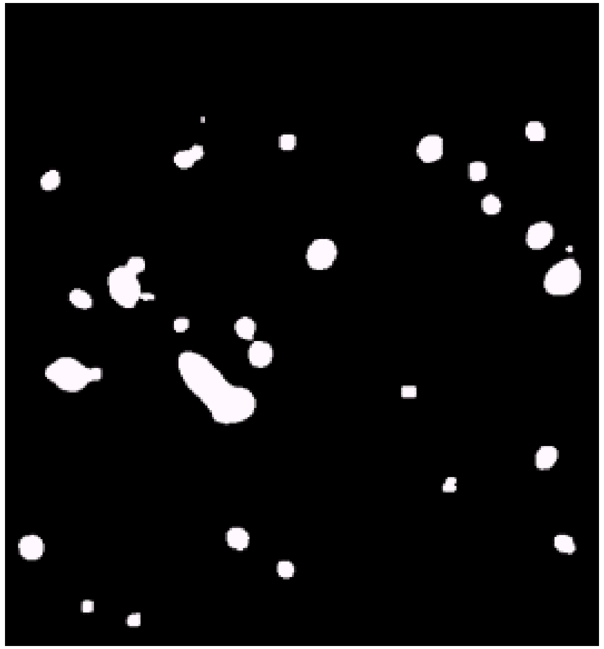
Example of Weighted Average Filters



Original Image



15 x 15 Averaging kernel



Result of Thresholding

Smoothing Spatial Filters

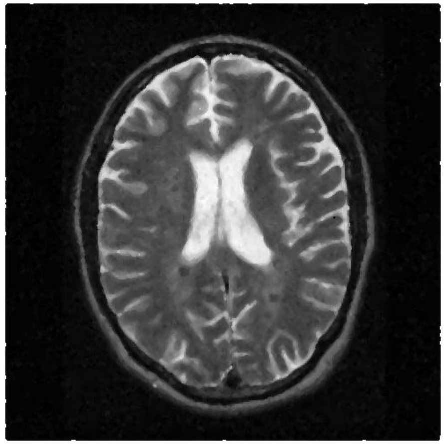
Order-statistic (Nonlinear) Filters

- Nonlinear
- Based on ordering (ranking) the pixels contained in the filter mask
- Replacing the value of the center pixel with the value determined by the ranking result
- E.g., median filter, max filter, min filter

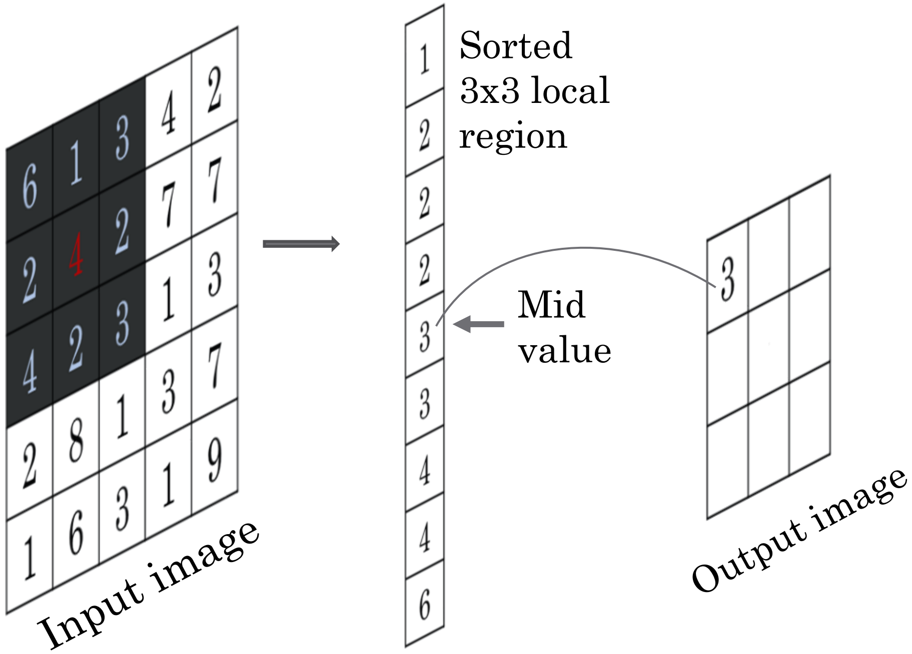
Median Filtering

- Assigns the mid value of all the gray levels in the mask to the center of mask
- Useful in removing impulse noise (also known as salt-and-pepper-noise).

Noisy Image

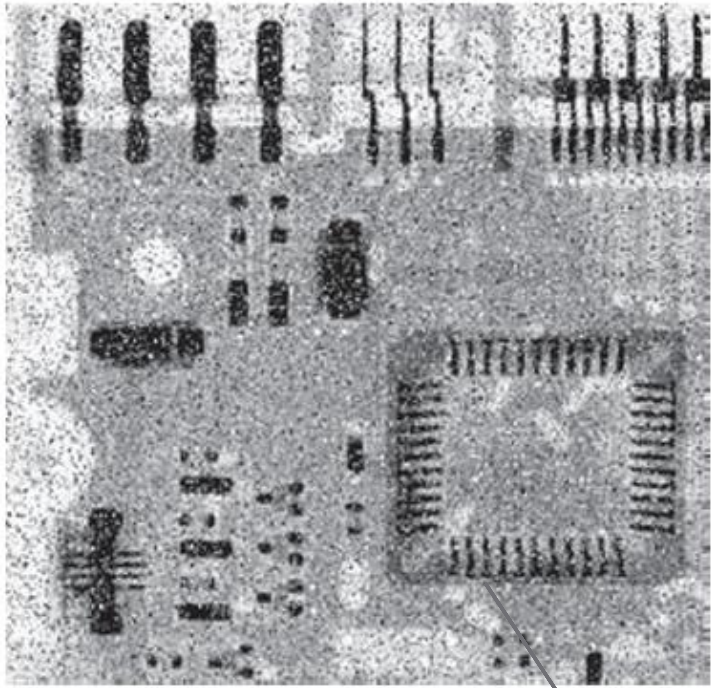


3x3 Median filtering



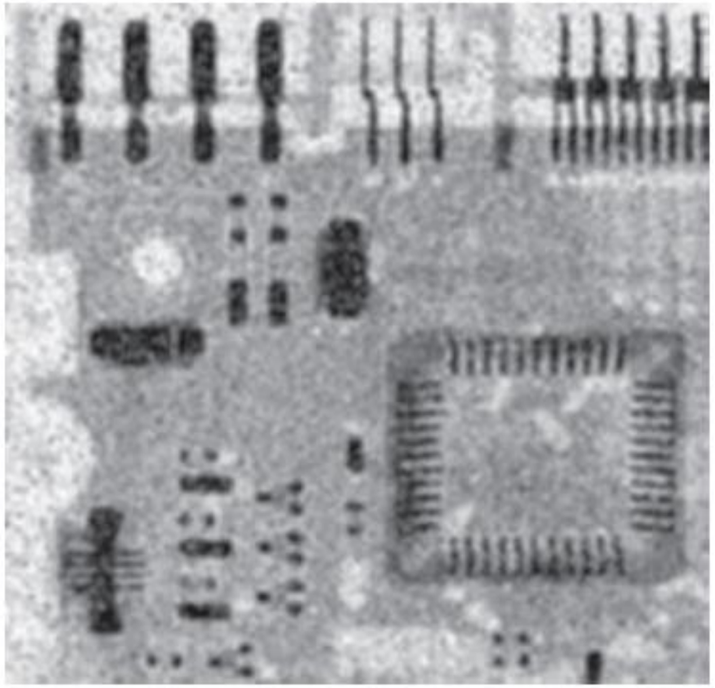
Smoothing Spatial Filters

Median Filtering

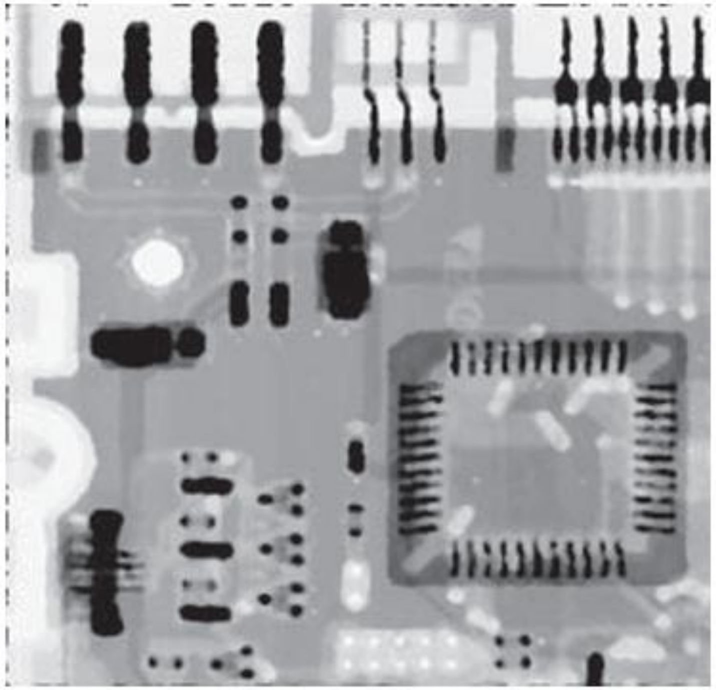


X-ray image of a circuit board

Corrupted by salt-and-pepper noise



Result 3x3 averaging filer



Using 3x3 median filter

Sharpening Spatial Filters

Purpose:

Sharpening highlights intensity transitions, enhancing edges and fine details in images.

Method:

Achieved by spatial differentiation, which emphasizes intensity changes (edges) and reduces areas with gradual intensity variations.

Key Concept:

Sharpening is often referred to as *highpass filtering*, where high frequencies (fine details) are enhanced, and low frequencies are suppressed.

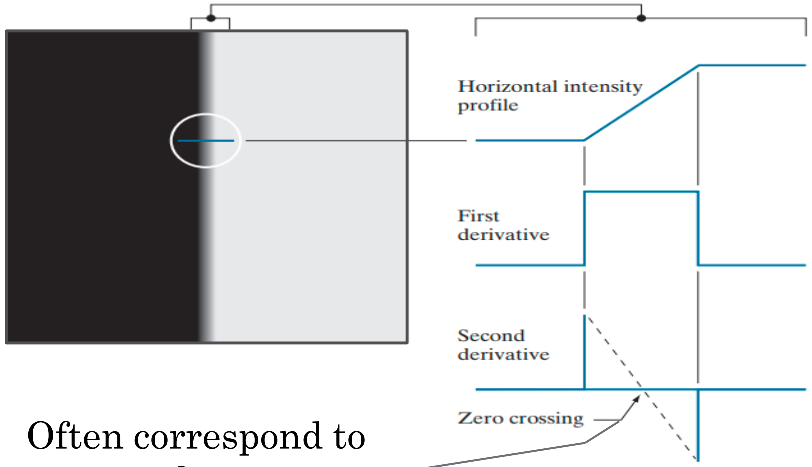
Applications:

Used in fields like electronic printing, medical imaging, industrial inspection, and military systems.

Original Image

Sharpened Image

edges



Often correspond to edges

Sharpening via Spatial Differentiation

First-Order Derivatives (Gradient-based sharpening):

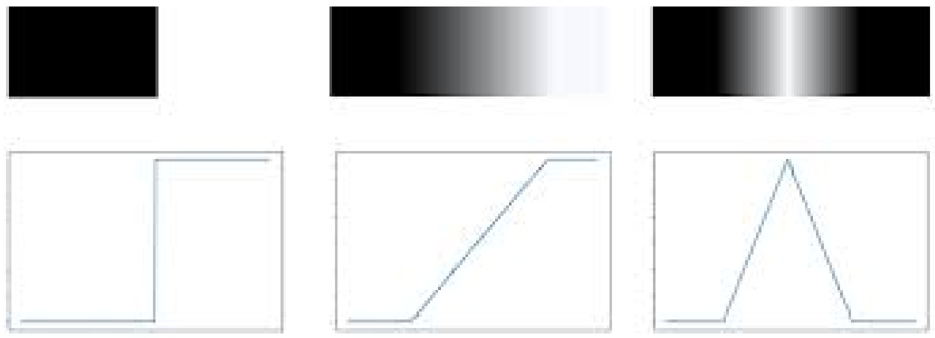
- Measures the rate of change of pixel intensity.
- Useful for detecting edges

$$\frac{\partial f}{\partial x} = f(x+1, y) - f(x, y) \qquad \text{X-direction}$$

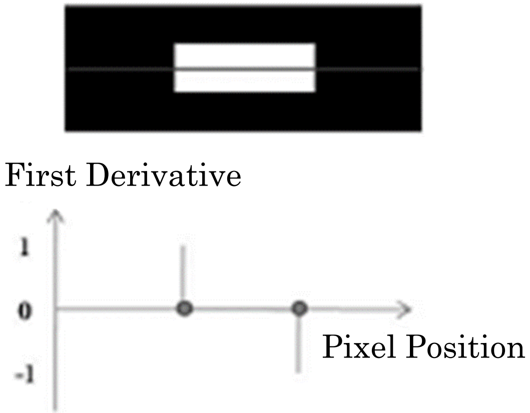
$$\frac{\partial f}{\partial y} = f(x, y+1) - f(x, y) \qquad \text{Y-direction}$$

The first derivative must be:

1. Zero in areas of constant intensity, (along flat segments)
2. Nonzero at the onset of an intensity step or ramp.
3. Nonzero along intensity ramps.



Step Edges: Abrupt change | Ramp Edges: Gradual change | Roof Edges: Peak change.



1	-1

First-order derivative kernel used for edge detection

Sharpening via Spatial Differentiation

Second order derivatives of digital functions

$$\frac{\partial^2 f}{\partial x^2} = f'(x+1) - f'(x)$$

$$= f(x+2) - f(x+1) + f(x+1) + f(x)$$

$$= [f(x+2) - f(x+1)] - [f(x+1) - f(x)]$$

f(x)	f(x+1)

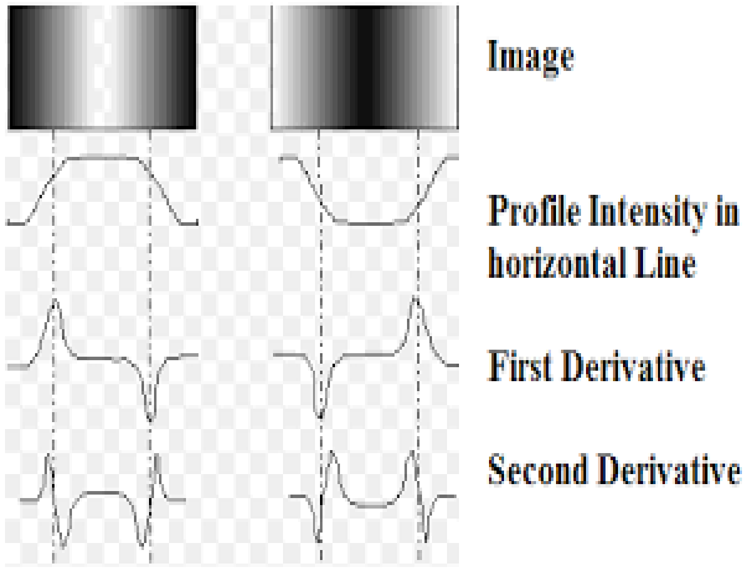
Position for the output pixel

$$\frac{\partial^2 f}{\partial x^2} = f(x+2) - 2f(x+1) + f(x)$$

f(x)	f(x+1)	f(x+2)

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$$

f(x-1)	f(x)	f(x+1)



The second derivative must be:

- Zero in areas of constant intensity.
- Nonzero at the onset and end of an intensity step or ramp.
- Zero along intensity ramps.

Y-direction kernel

1
-2
1

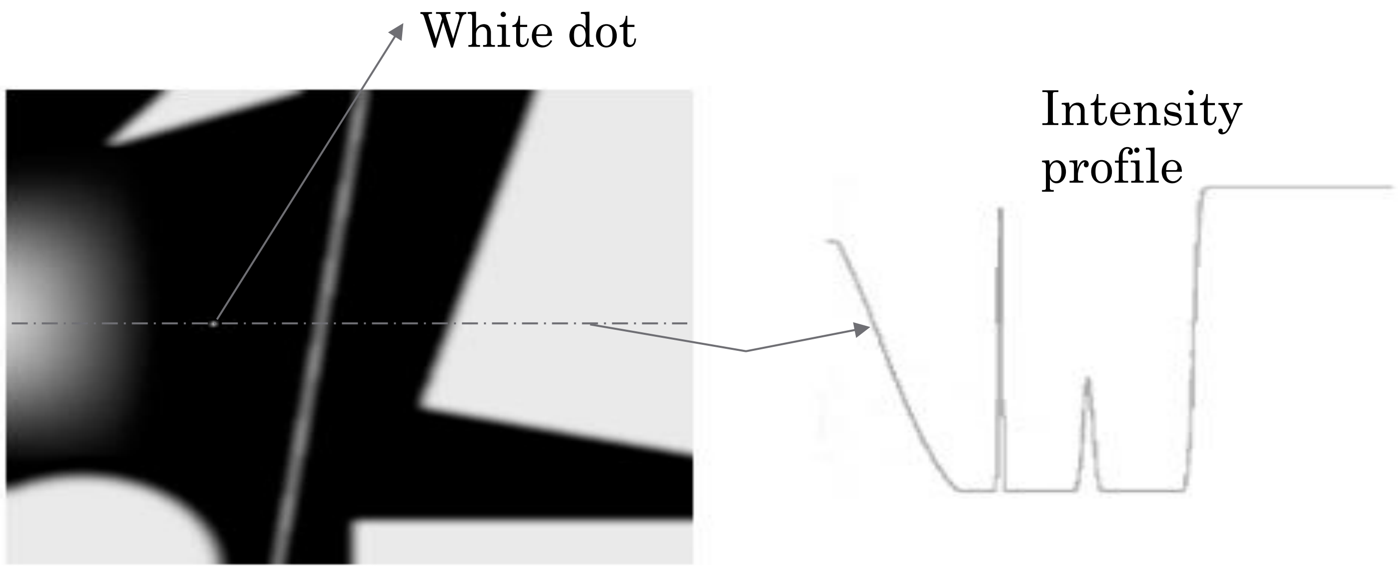
1	-2	1

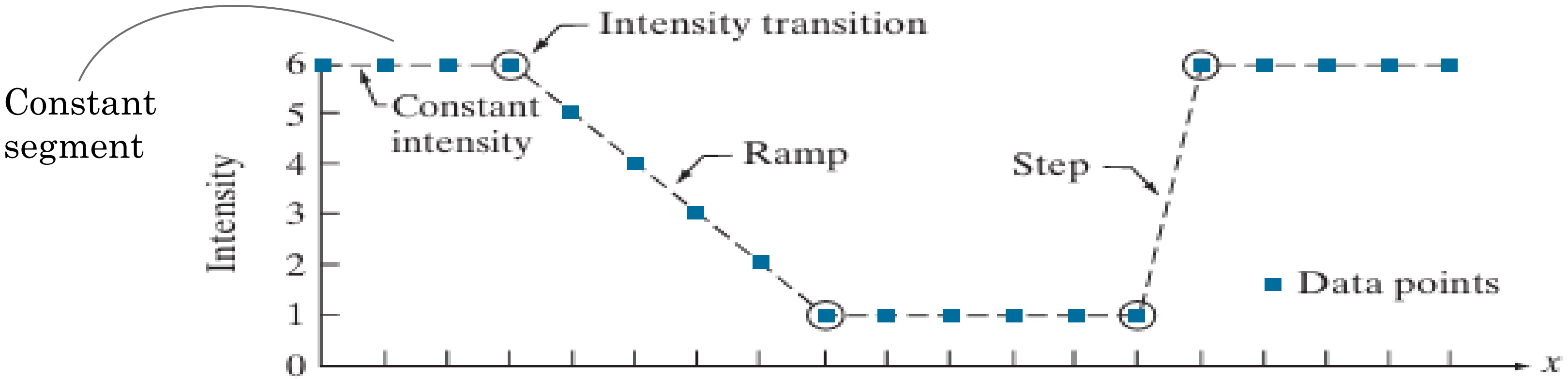
X-direction kernel

Example of derivatives

- 1^{st} derivative detect thick edges while 2^{nd} derivative detect thin edges.

- 2^{nd} derivative has mush stronger response at gray-level step than 1^{st} derivative.





Values of scan line	6	6	6	6	5	4	3	2	1	1	1	1	1	1	6	6	6	6	6
1st derivative		0	0	−1	−1	−1	−1	−1	0	0	0	0	0	5	0	0	0	0	
2nd derivative		0	0	−1	0	0	0	0	1	0	0	0	0	5	−5	0	0	0	

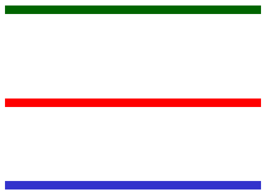
Various situations encountered for derivatives

$$f' = \frac{\partial f}{\partial x} \qquad f'' = \frac{\partial^2 f}{\partial x^2}$$

- Ramps or steps in the 1D profile normally characterize the edges in an image
- f″ is nonzero at the onset and end of the ramp: produce thin (double) edges
- f ′ is nonzero along the entire ramp produce thick edges

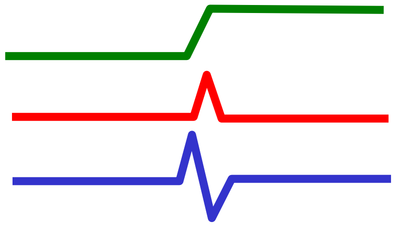
•Flat segment→ (*f* ′)=0; (*f* ″)=0

f	0	0	0	0	0	
f′		0	0	0	0	
f″		0	0	0		



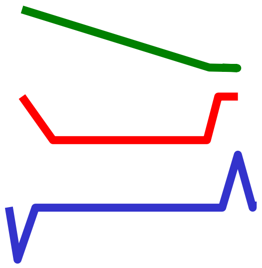
•Step→ (*f* ′):{0,+,0}; (*f* ″):{0,+,–,0}

f	0	0	0	7	7	7	7	
f		0	0	7	0	0	0	0
f″		0	7	–7	0	0	0	



•Ramp→ (*f* ′)≈constant; (*f* ″)=0

f	5	4	3	2	1	0	0	
f′	0	–1	–1	–1	–1	–1	0	0
f″	–1	0	0	0	0	1	0	



The Laplacian Filter

- 2D second-order derivative operator used for image sharpening.
- Highlights sharp intensity transitions and de-emphasizes regions with slow intensity changes.
- Produces grayish edge lines and discontinuities on a dark background.

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$$\therefore \frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

$$\therefore \frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y)$$

$$\nabla^2 f(x, y) = f(x+1, y) + f(x-1, y) + f(x, y+1) \\ + f(x, y-1) - 4f(x, y)$$

- We can apply these kernels using convolution operations.

	f(x-1, y)	
f(x, y-1)	f(x, y)	f(x,y+1)
	f(x+1,y)	

Rotation invariant at 90° ++

0	1	0
1	-4	1
0	1	0

0	-1	0
-1	4	-1
0	-1	0

Laplacian kernel, Equation

Rotation invariant at 45° ++

Includes the diagonal terms

1	1	1
1	-8	1
1	1	1

-1	-1	-1
-1	8	-1
-1	-1	-1

Laplacian for Image Enhancement

To obtain the enhance image

$$g(x,y) = \begin{cases} f(x,y) - \nabla^2 f(x,y), & w_5 < 0 \\ f(x,y) + \nabla^2 f(x,y), & w_5 > 0 \end{cases}$$

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

- f(x, y) → Original Image
- $\nabla^2 f(x,y)$ → Laplacian of original image
- In this way, background tonality can be perfectly preserved while details are enhanced.

Laplacian Kernel, Highlight areas of sharp intensity

-1	-1	-1
-1	8	-1
-1	-1	-1

+

0	0	0
0	1	0
0	0	0

Identity kernel, doesn't modify the pixel values

=

Resultant Kernel, produce a sharper image by preserving the original intensity values

-1	-1	-1
-1	9	-1
-1	-1	-1

Laplacian for Image Enhancement (Example)

$$g(x,y) = \begin{cases} f(x,y) - \nabla^2 f(x,y), & w_5 < 0 \\ f(x,y) + \nabla^2 f(x,y), & w_5 > 0 \end{cases}$$

Blurred image of the North Pole of the moon.

Laplacian image obtained using the 90° isotropic kernel

Image sharpened using equation above

Image sharpened using the same procedure, but with 45° isotropic kernel.

Thank You