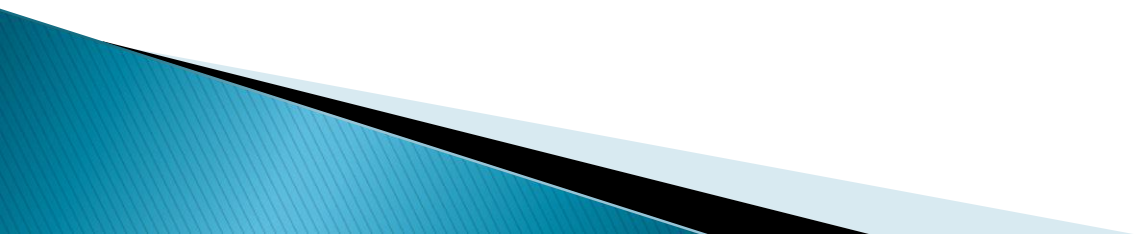
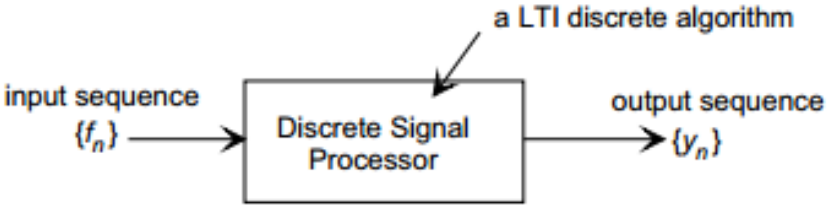
Filtering and Convolution

Linear Systems



Review

- Discrete-time signal processing of data sequences $\{f_n\}$ that might be samples of real continuous experimental data.



- Properties of LTI Continuous Filter

$$y_3(t) = ay_1(t) + by_2(t).$$

Systems and Filters

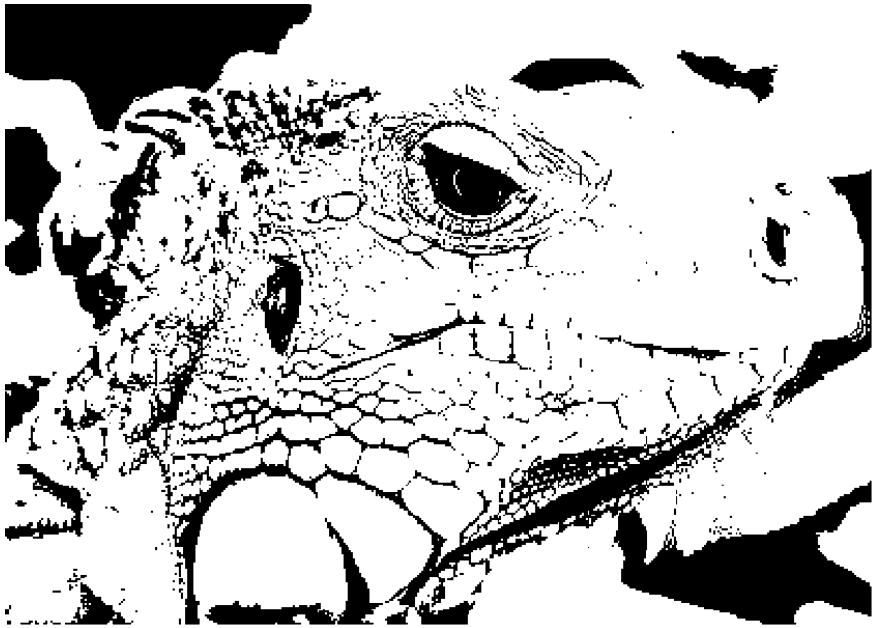
- **Filtering**
  - Forming a new image whose pixel values are transformed from original pixel values

- **Goals**
  - Goal is to extract useful information from images, or transform images into another domain where we can modify/enhance image properties
    - Features (edges, corners, blobs…)
    - super-resolution; in-painting; de-noising

Juan Carlos

Intuition behind linear systems

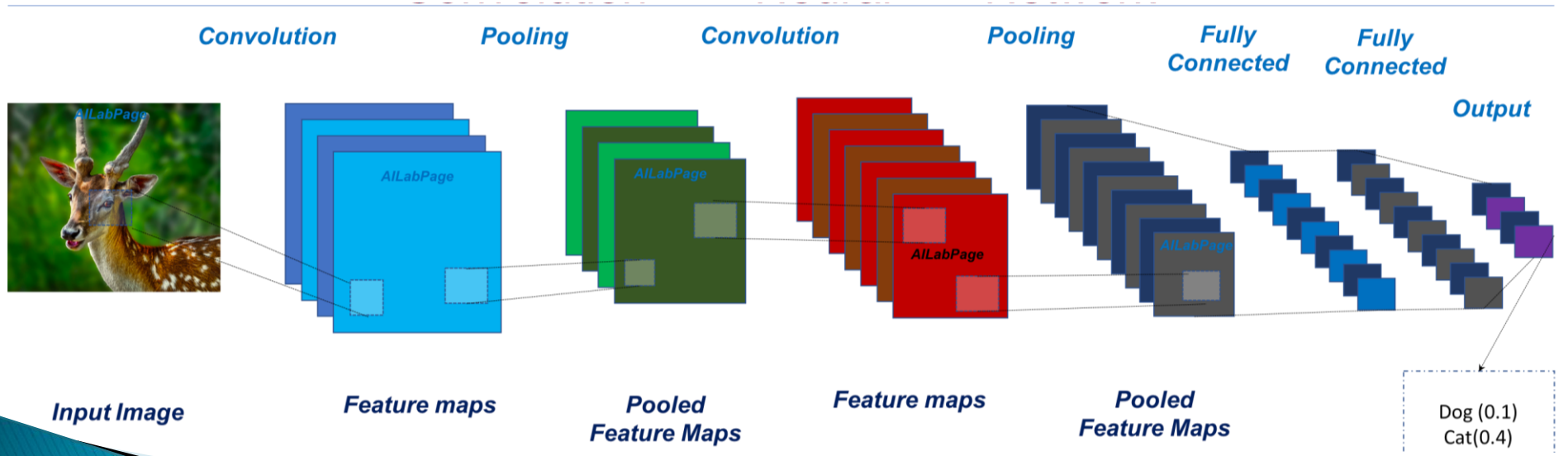
- We will view linear systems are a type of function that operates over images



Juan Carlos

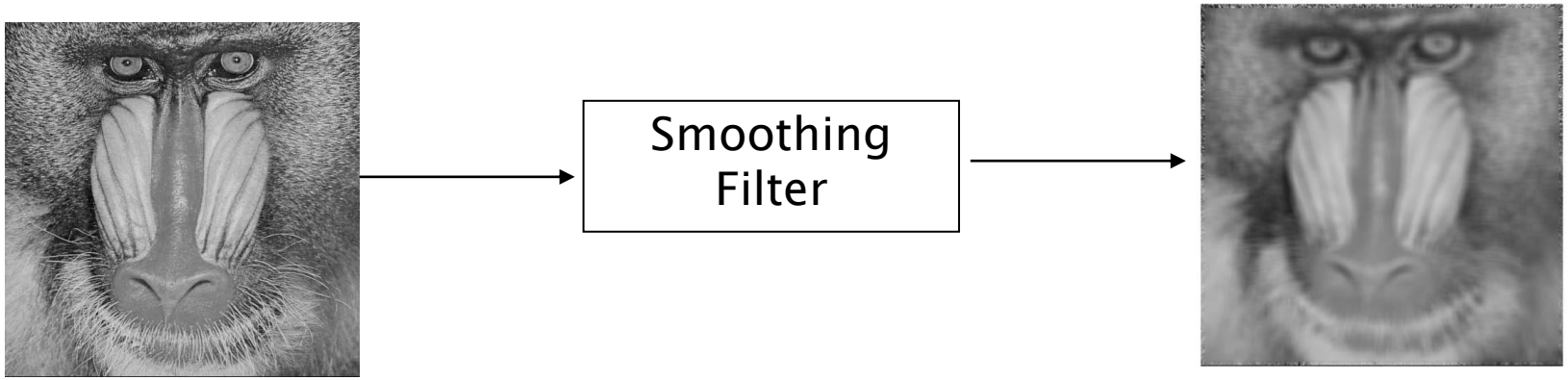
Aside

- Neural networks and specifically convolutional neural networks are a type of system or non-linear system that contains multiple individual linear sub-systems.



Juan Carlos

Filtering



Filter example #1: Moving Average

- 2D moving average over a 3 × 3 window of neighborhood

$$g[n,m] = \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{l=m-1}^{m+1} f[k,l]$$

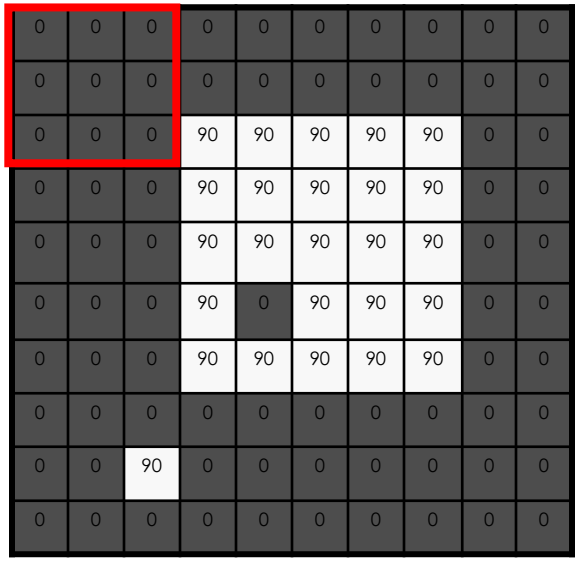
$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} f[n-k, m-l]$$

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

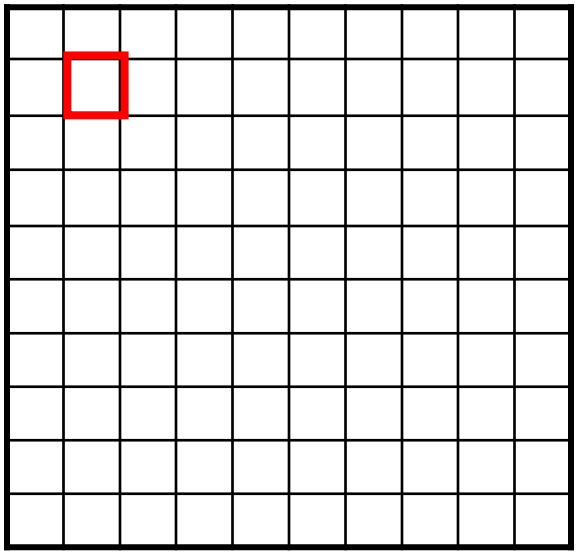
Juan Carlos

Filter example #1: Moving Average

$f[n,m]$



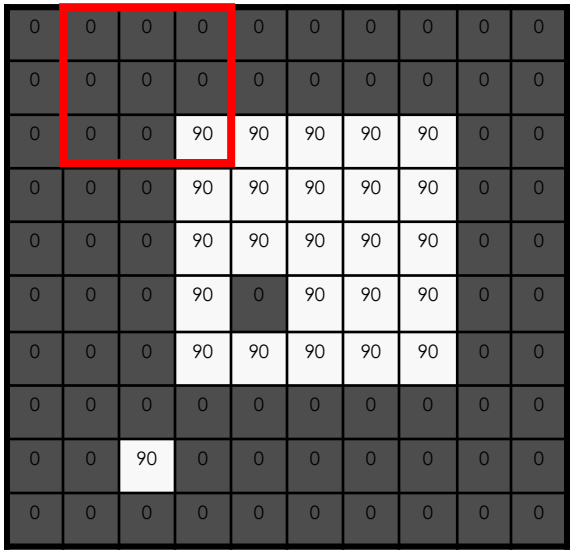
$g[n,m]$



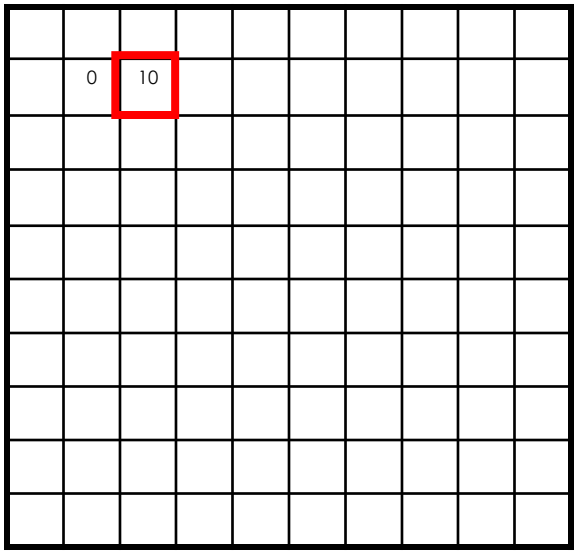
Courtesy of S. Seitz

Filter example #1: Moving Average

$f[n,m]$

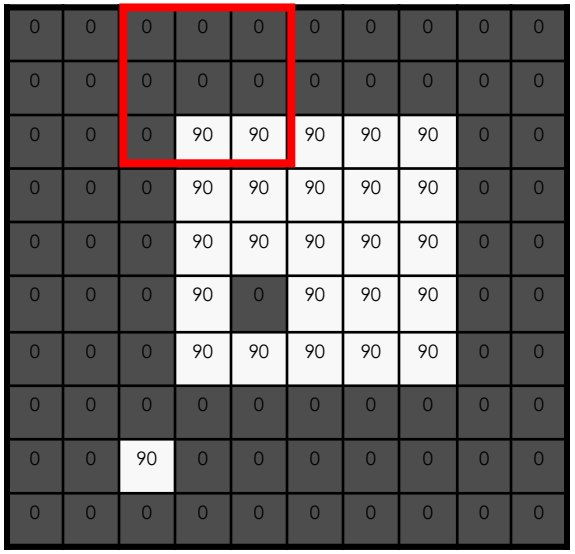


$g[n,m]$

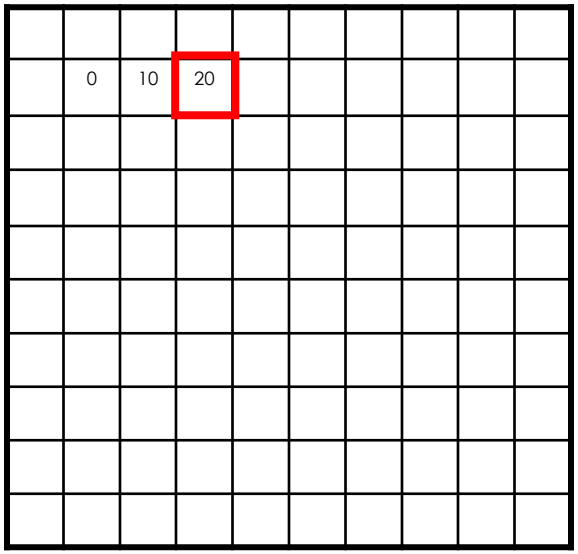


Filter example #1: Moving Average

$f[n,m]$

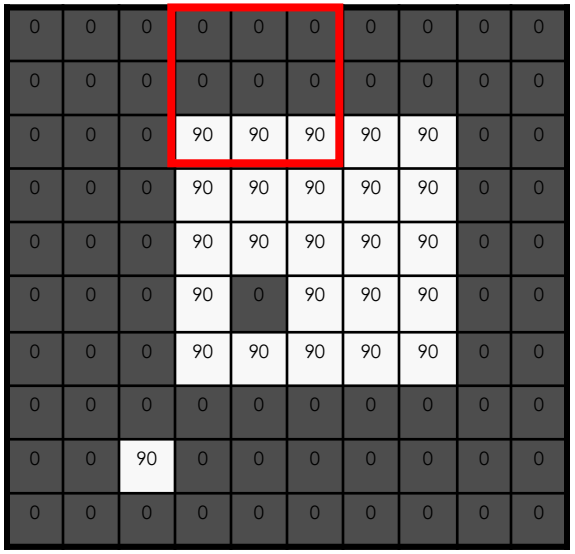


$g[n,m]$

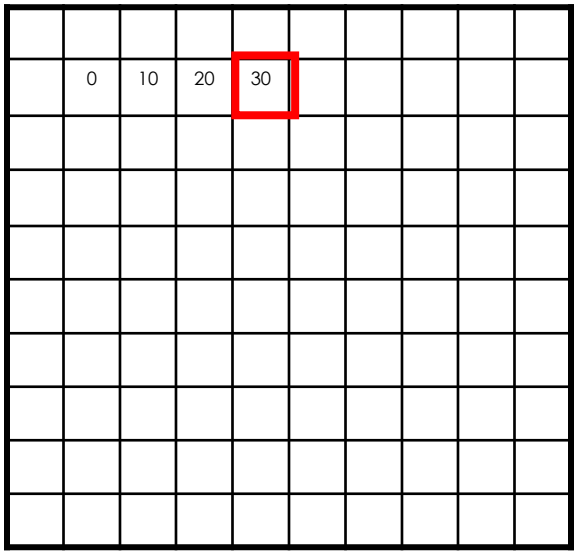


Filter example #1: Moving Average

$f[n,m]$

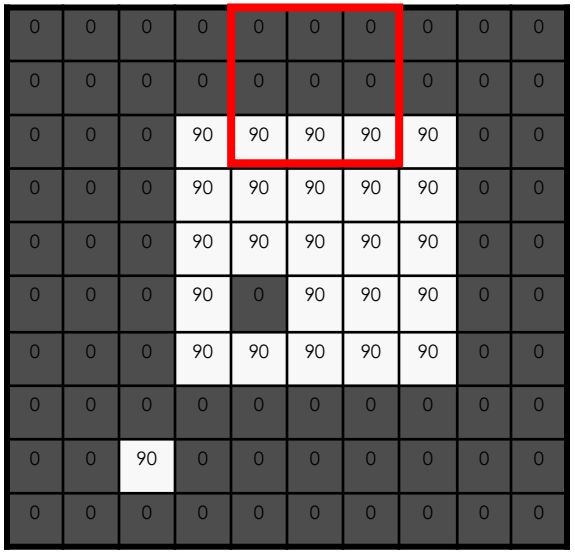


$g[n,m]$

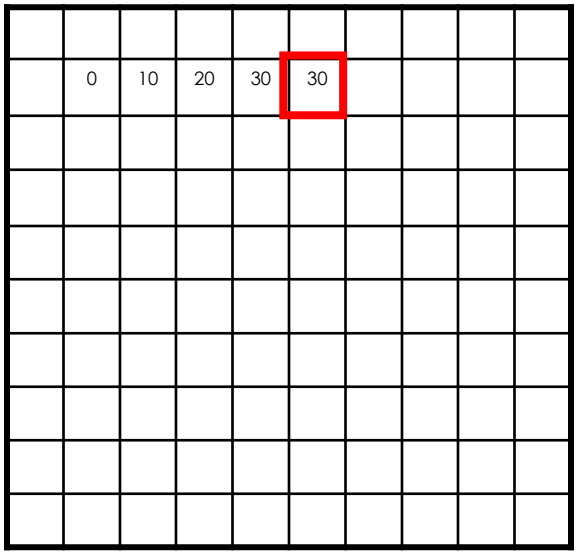


Filter example #1: Moving Average

$f[n,m]$



$g[n,m]$



Filter example #1: Moving Average

$f[n,m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$g[n,m]$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

Filter example #1: Moving Average

Original image

Smoothed image

Properties of Linear Systems

- Shift-Invariant
  - $g(t) = T\{f(t)\} \longleftrightarrow g(t-t_0) = T\{f(t-t_0)\}$
- Additivity $S[f_i[n,m] + f_j[n,m]] = S[f_i[n,m]] + S[f_j[n,m]]$
- Homogeneity $S[\alpha f_i[n,m]] = \alpha S[f_i[n,m]]]$
- Superposition

$$S[\alpha f_i[n,m] + \beta f_j[n,m]] = \alpha S[f_i[n,m]] + \beta S[f_j[n,m]]$$

Is the moving average filter is shift-invariant?

$$g[n,m] = \frac{1}{9}\sum_{k=n-1}^{n+1}\sum_{l=m-1}^{m+1} f[k,l] \quad = \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1} f[n-k,m-l]$$

$$\mathrm{g}[n-n_0,m-m_0] = \tfrac{1}{9}\textstyle\sum_{k=-1}^{1}\sum_{l=-1}^{1} f[n-n_0-k,m-m_0-l]$$

Let $f[n-n_0,m-m_0]$ be a shifted input of $f[n,m]$

Now let's pass $f[n-n_0,m-m_0]$ through the system:

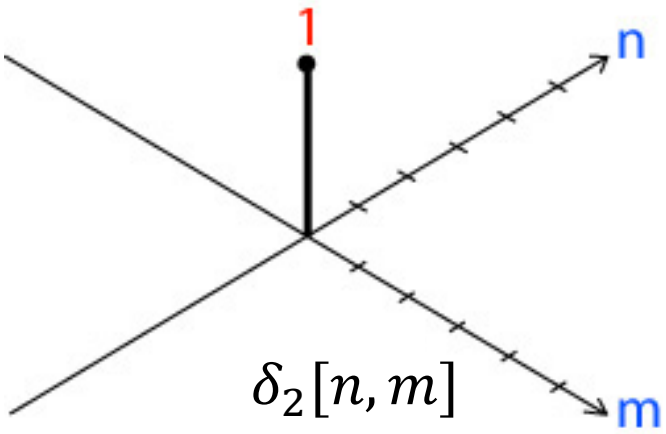
$$f[n-n_0,m-m_0] \xrightarrow{s} \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1} f[n-n_0-k,m-m_0-l]$$

Yes!

Juan Carlos

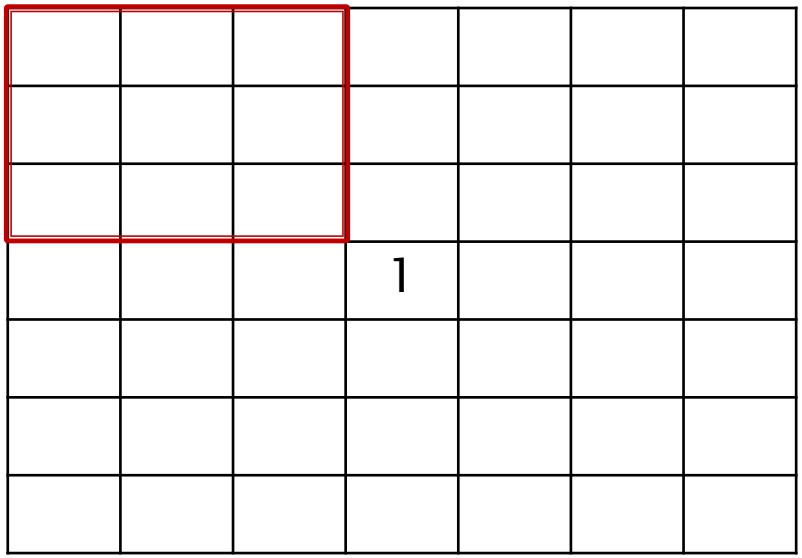
2D impulse function

- $\delta_2[n,m]$ is 1 at (0,0).
- Otherwise, $\delta_2[n,m]$ is 0

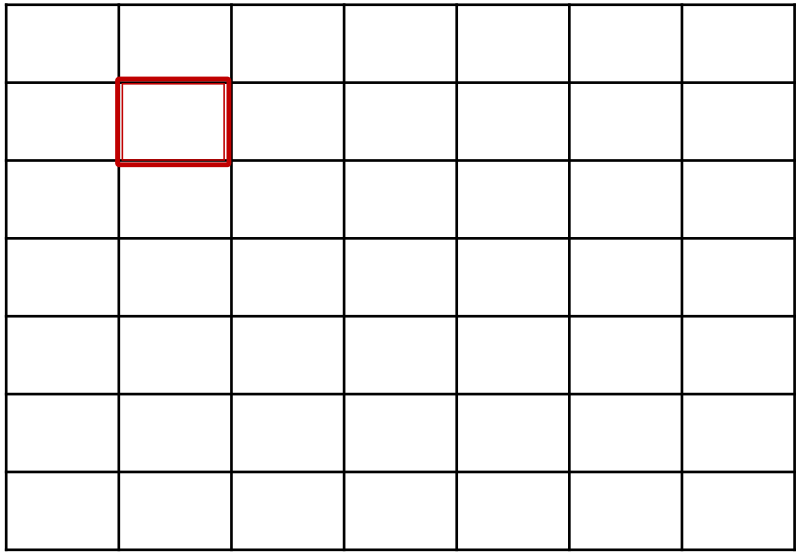


Impulse response to the moving average filter

$$g[n,m] = \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{l=m-1}^{m+1} f[k,l]$$



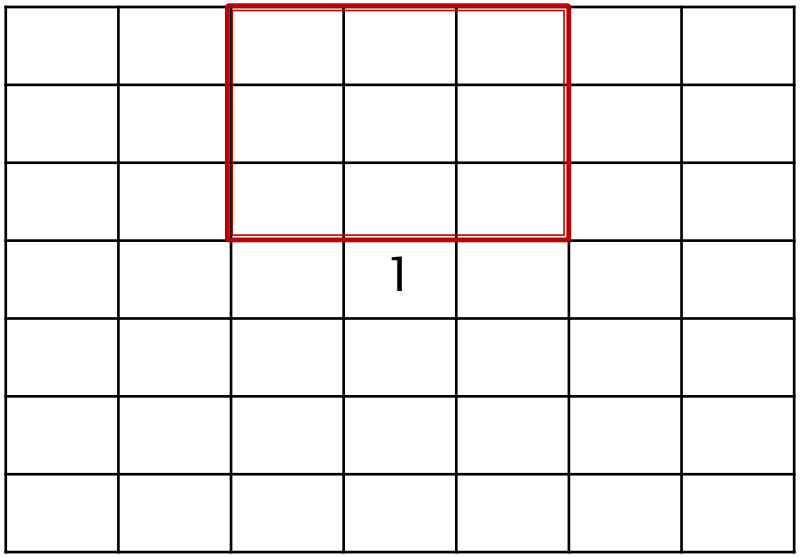
Input (2D delta)



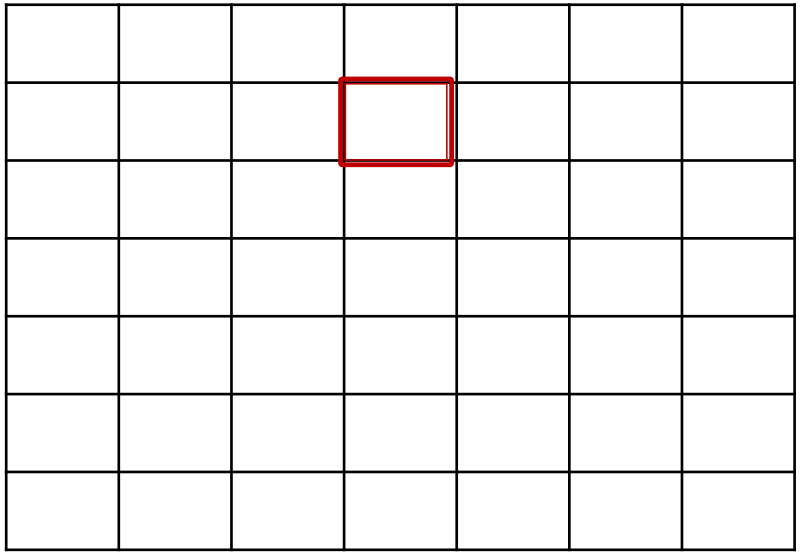
Output

Impulse response to the moving average filter

$$g[n,m] = \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{l=m-1}^{m+1} f[k,l]$$



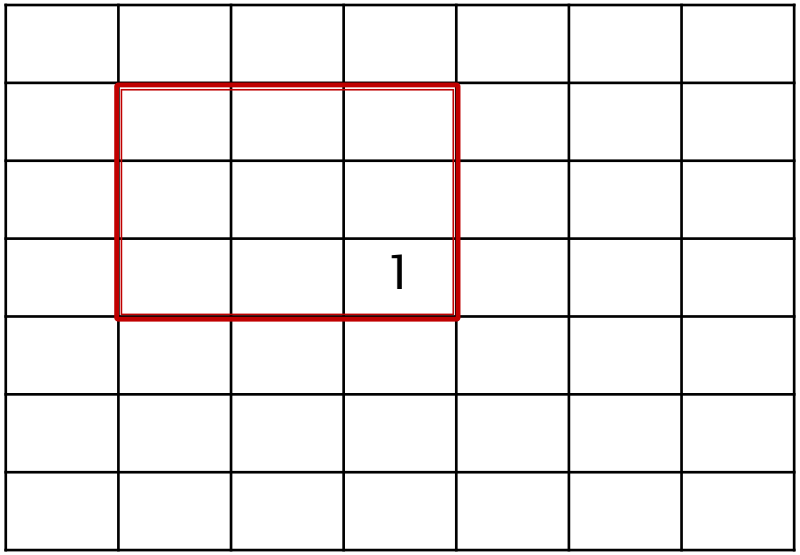
Input (2D delta)



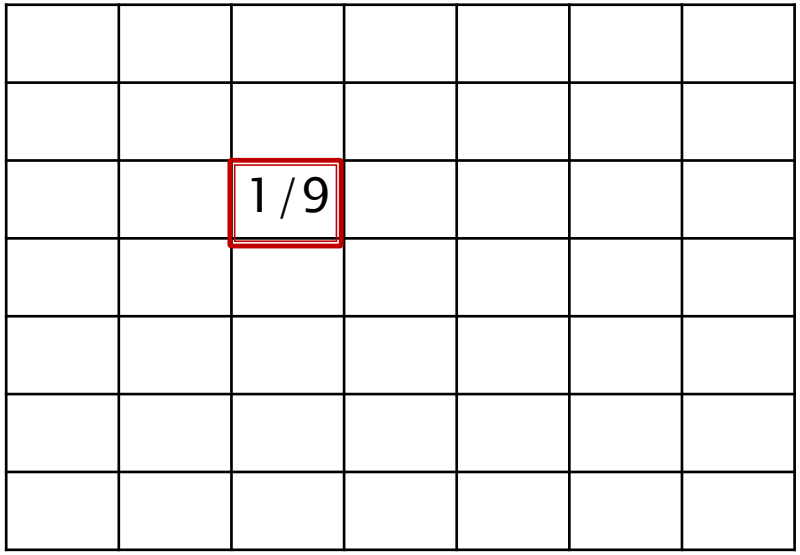
Output

Impulse response to the moving average filter

$$g[n,m] = \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{l=m-1}^{m+1} f[k,l]$$



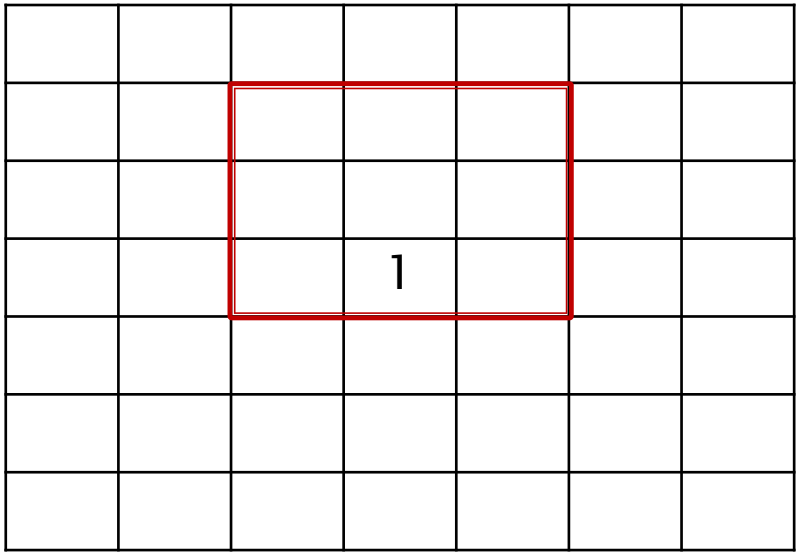
Input (2D delta)



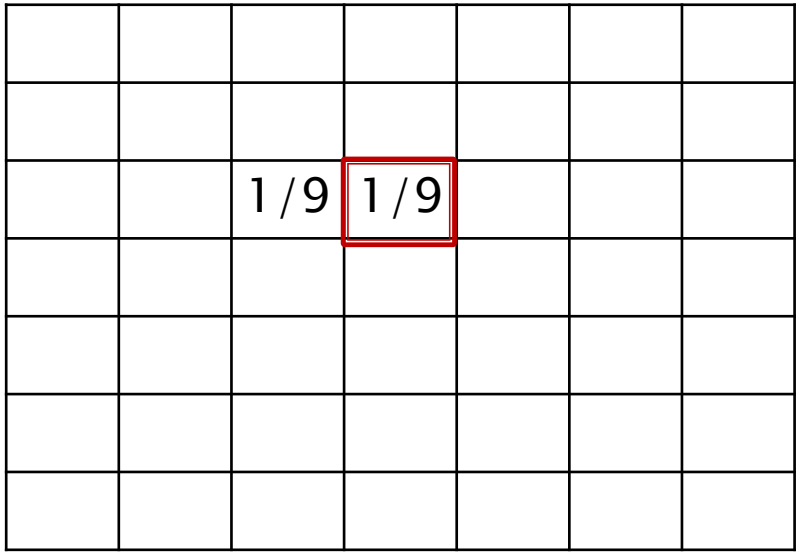
Output

Impulse response to the moving average filter

$$g[n,m] = \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{l=m-1}^{m+1} f[k,l]$$



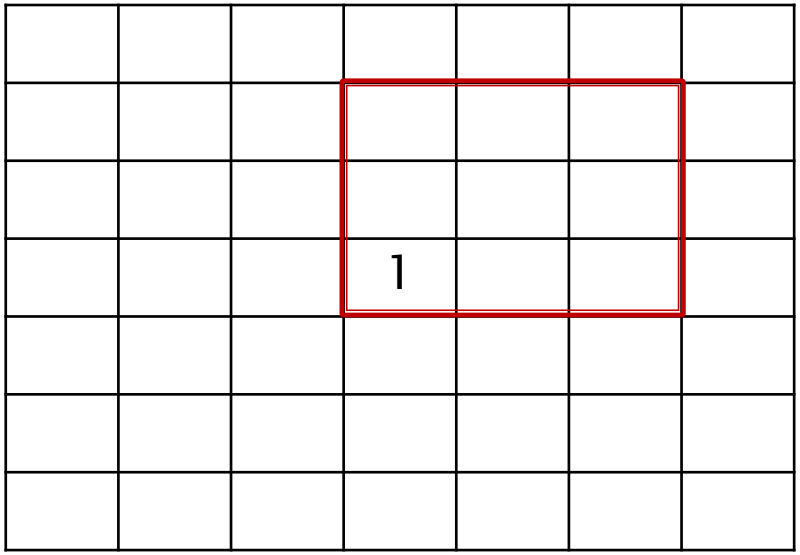
Input (2D delta)



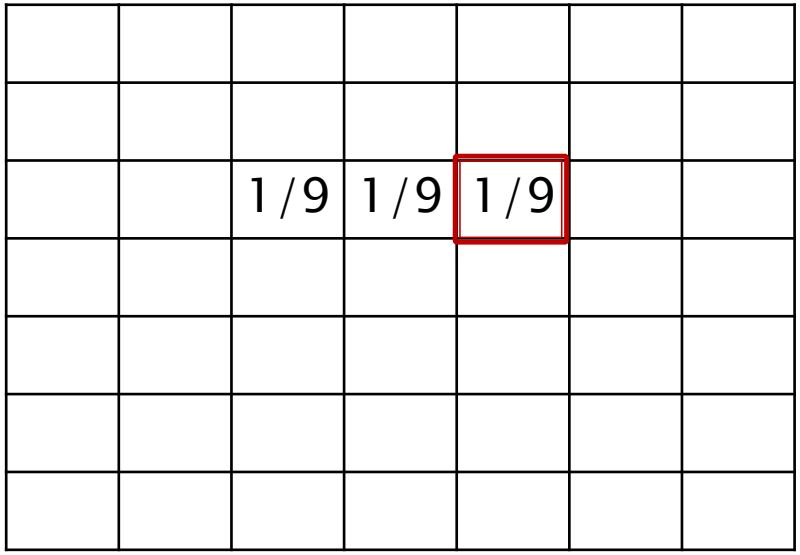
Output

Impulse response to the moving average filter

$$g[n,m] = \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{l=m-1}^{m+1} f[k,l]$$



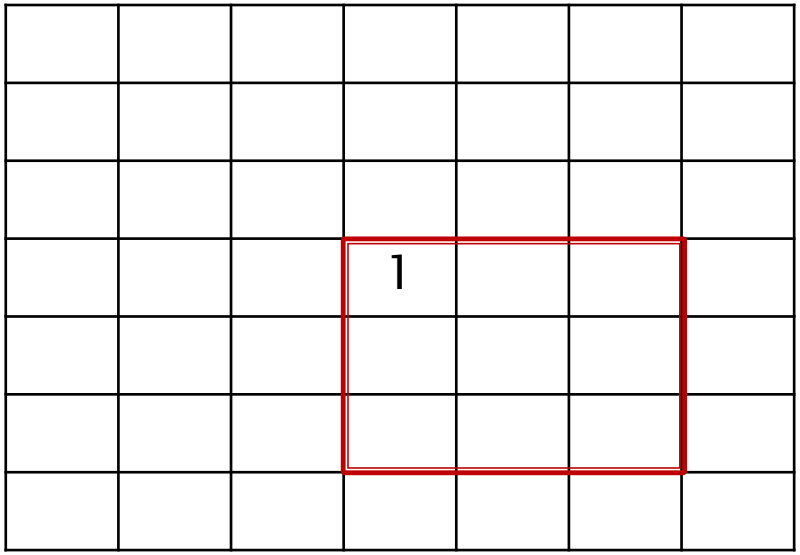
Input (2D delta)



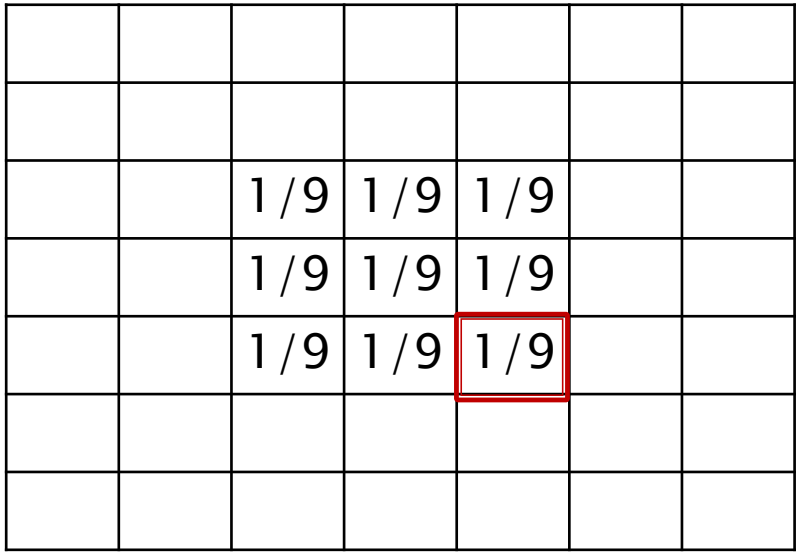
Output

Impulse response to the moving average filter

$$g[n,m] = \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{l=m-1}^{m+1} f[k,l]$$



Input (2D delta)



Output

Impulse response to the moving average filter

$$h[n,m] = \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \delta_2[n-k, m-l]$$

$$= \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$$

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

Juan Carlos

Generalization

- Let's say our input f is a 3x3 image:

f[0,0]	f[0,1]	f[1,1]
f[1,0]	f[1,1]	f[1,2]
f[2,0]	f[2,1]	f[2,2]

$$f[n,m] = f[0,0] \times \delta_2[n-0, m-0] + f[0,1] \times \delta_2[n-0, m-1] + \ldots + f[n,m] \times \delta_2[0,0] + \ldots$$

- We can rewrite the input f as follows.

$$f[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=\infty}^{\infty} f[k,l] \times \delta_2[n-k, m-l]$$

LSI system

- Let S be a LSI filter.

$$S[f] = S\left[\sum_{k=-\infty}^{\infty}\sum_{l=\infty}^{\infty} f[k,l] \times \delta_2[n-k, m-l]\right]$$

$$= \sum_{k=-\infty}^{\infty}\sum_{l=\infty}^{\infty} f[k,l] \times S[\delta_2[n-k, m-l]]$$

$$= \sum_{k=-\infty}^{\infty}\sum_{l=\infty}^{\infty} f[k,l] \times h[n-k, m-l]$$

- Convolution
  - $S[f] = f[n,m] * h[n,m]$

Convolution

2D Convolution

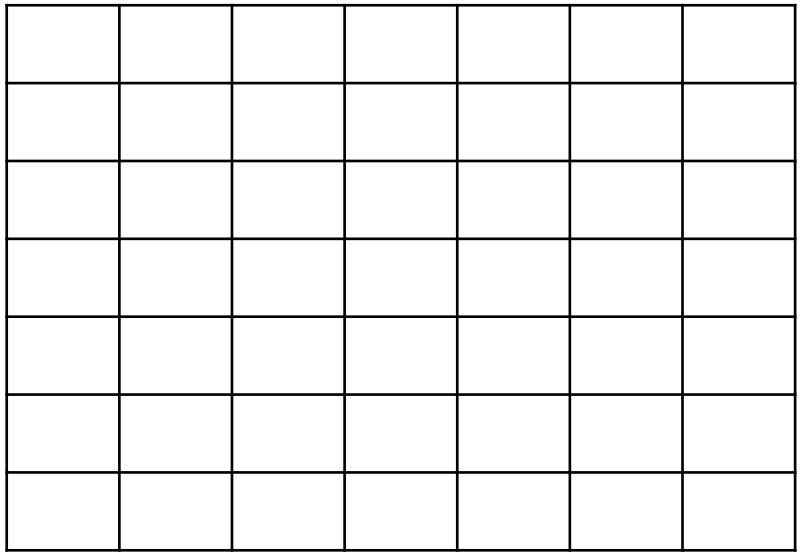
$$g(m,n) = f(m,n) * h(m,n)$$
$$= \sum_{k=-\infty}^{\infty} \sum_{l=\infty}^{\infty} f[k,l] \times h[n-k, m-l]$$

- Example

$$g(0,0) = f(0,0) * h(0,0)$$
$$= \sum_{k=-\infty}^{\infty} \sum_{l=\infty}^{\infty} f[k,l] \times h[0-k, 0-l]$$

2D Convolution Example

$$g(0,0) = f(0,0) * h(0,0)$$
$$= \sum_{k=-\infty}^{\infty} \sum_{l=\infty}^{\infty} f[k,l] \times h[0-k, 0-l]$$



$f[n,m]$

[0,0]

-1	-2	-3
4	0	5
1	2	3

$h[n,m]$

2D Convolution Example

$$g(0,0) = f(0,0) * h(0,0)$$
$$= \sum_{k=-\infty}^{\infty} \sum_{l=\infty}^{\infty} f[k,l] \times h[0-k, 0-l]$$

$h[-n,-m]$

3	2	1					
5	0	4					
−3	−2	−1					

$f[n,m]$

[0,0]

−1	−2	−3
4	0	5
1	2	3

$h[n,m]$

Convolution in 2D – examples

0	0	0
0	1	0
0	0	0

$h[n, m]$

0	0	0
1	0	0
0	0	0

$h[n, m]$

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

$h[n, m]$

Kernel Size

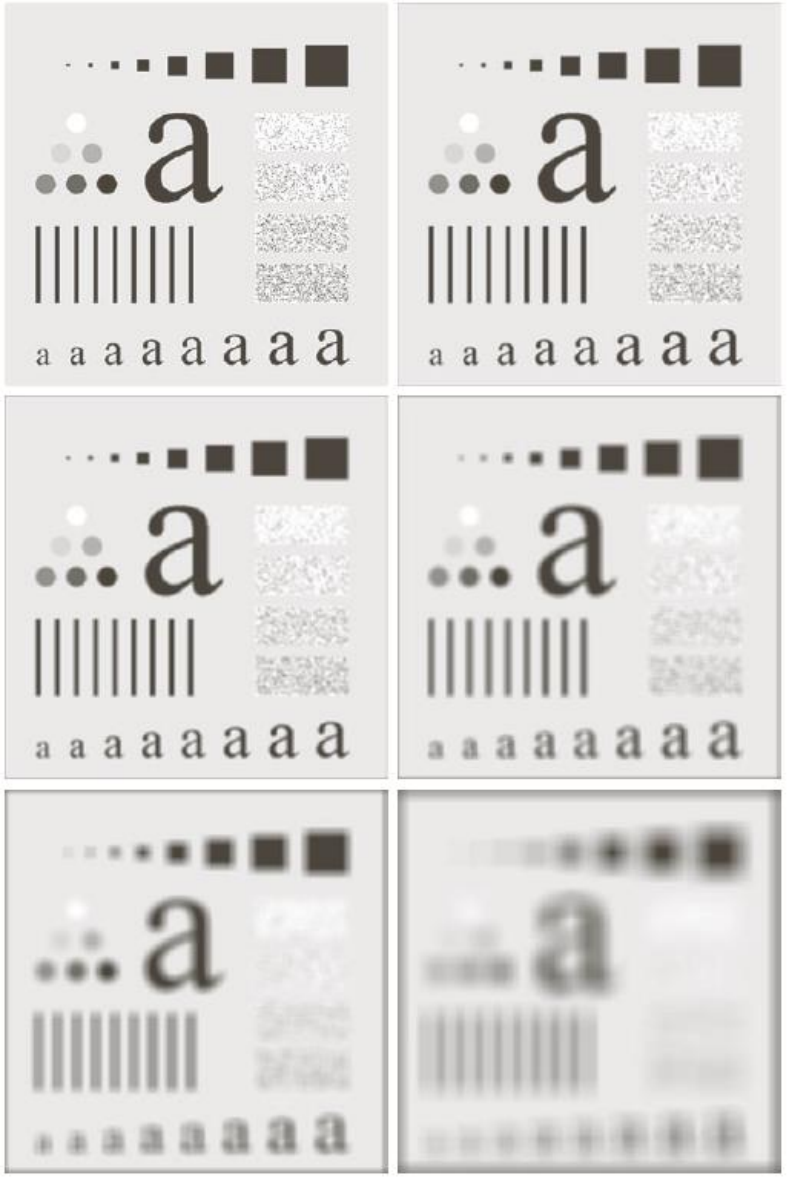
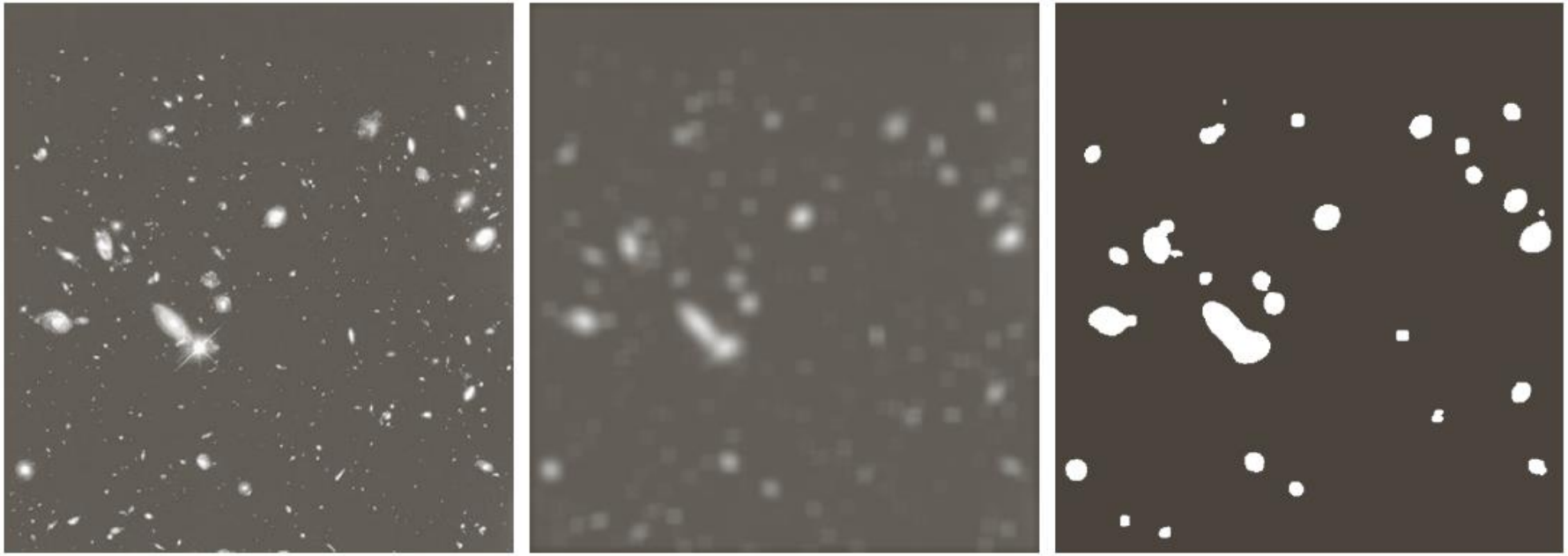


FIGURE 3.33 (a) Original image, of size 500 × 500 pixels. (b)–(f) Results of smoothing with square averaging filter masks of sizes $m = 3, 5, 9, 15,$ and 35, respectively. The black squares at the top are of sizes 3, 5, 9, 15, 25, 35, 45, and 55 pixels, respectively; their borders are 25 pixels apart. The letters at the bottom range in size from 10 to 24 points, in increments of 2 points; the large letter at the top is 60 points. The vertical bars are 5 pixels wide and 100 pixels high; their separation is 20 pixels. The diameter of the circles is 25 pixels, and their borders are 15 pixels apart; their intensity levels range from 0% to 100% black in increments of 20%. The background of the image is 10% black. The noisy rectangles are of size 50 × 120 pixels.

Kernel Size



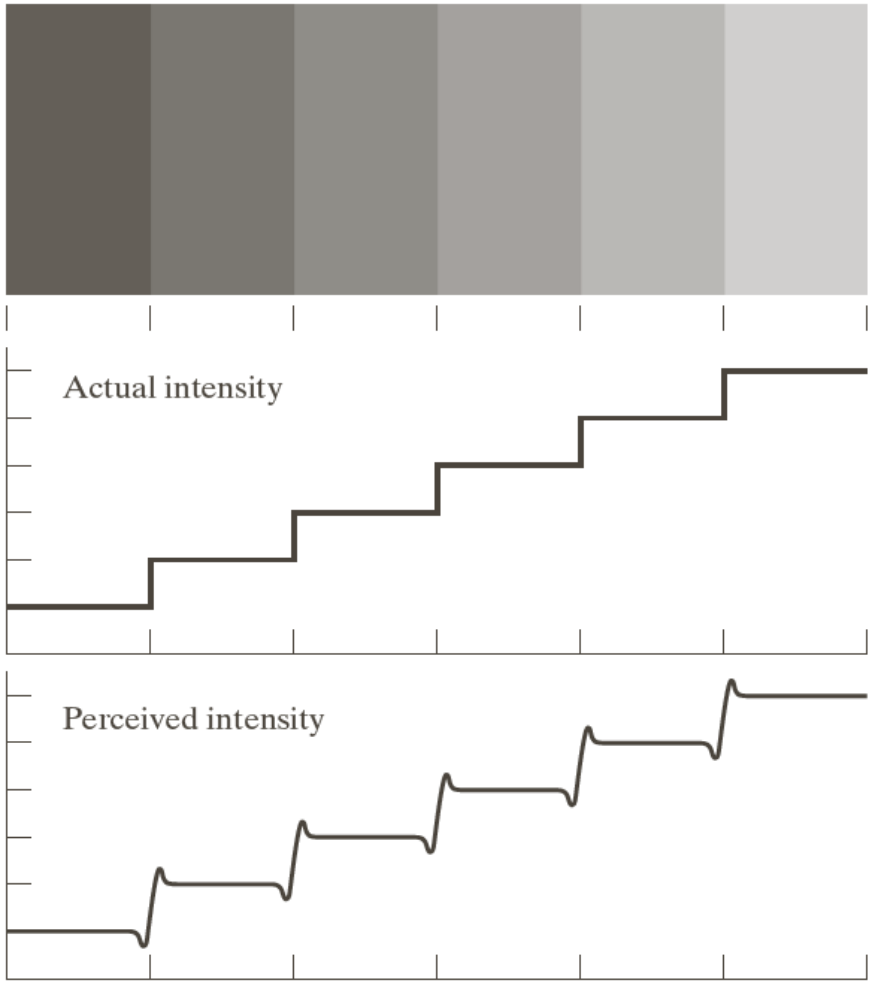
a b c

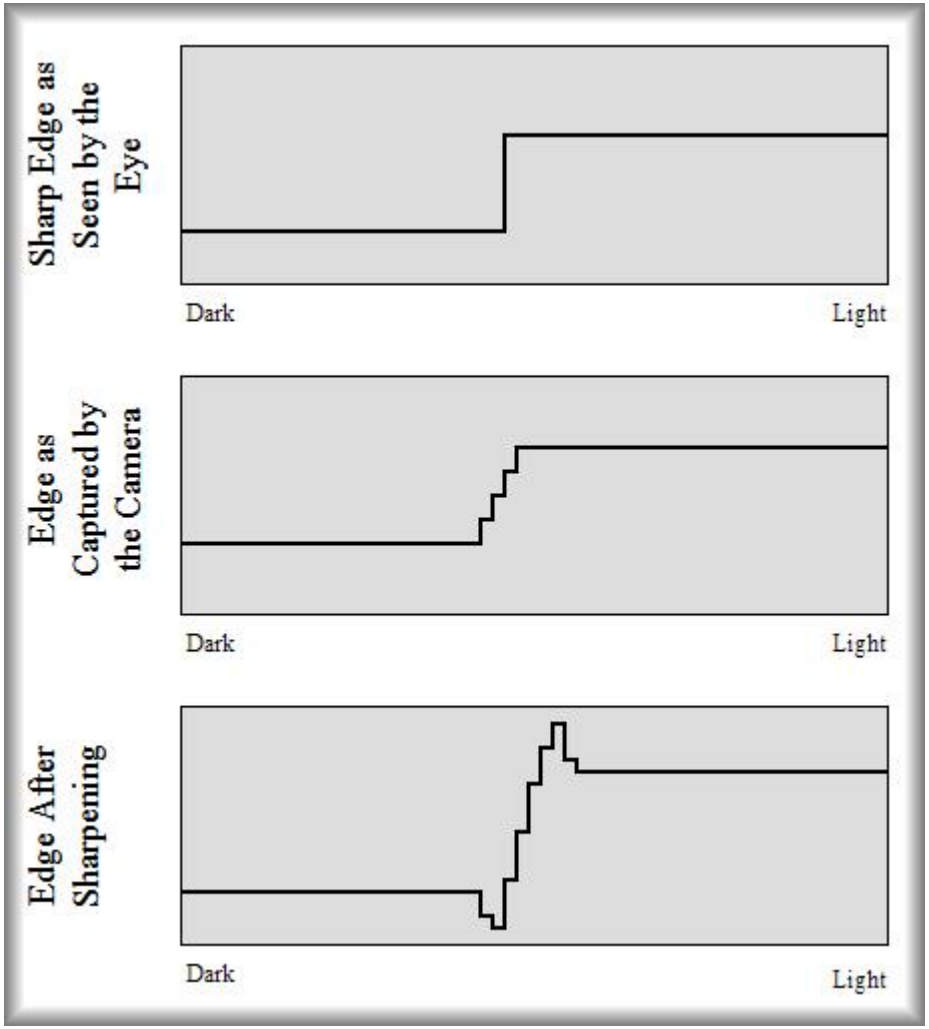
FIGURE 3.34 (a) Image of size 528 × 485 pixels from the Hubble Space Telescope. (b) Image filtered with a 15 × 15 averaging mask. (c) Result of thresholding (b). (Original image courtesy of NASA.)

Sharpening Spatial Filters

- The principal objective of sharpening is to highlight transitions in intensity.

How can we Sharpen images?





Foundation (Background)

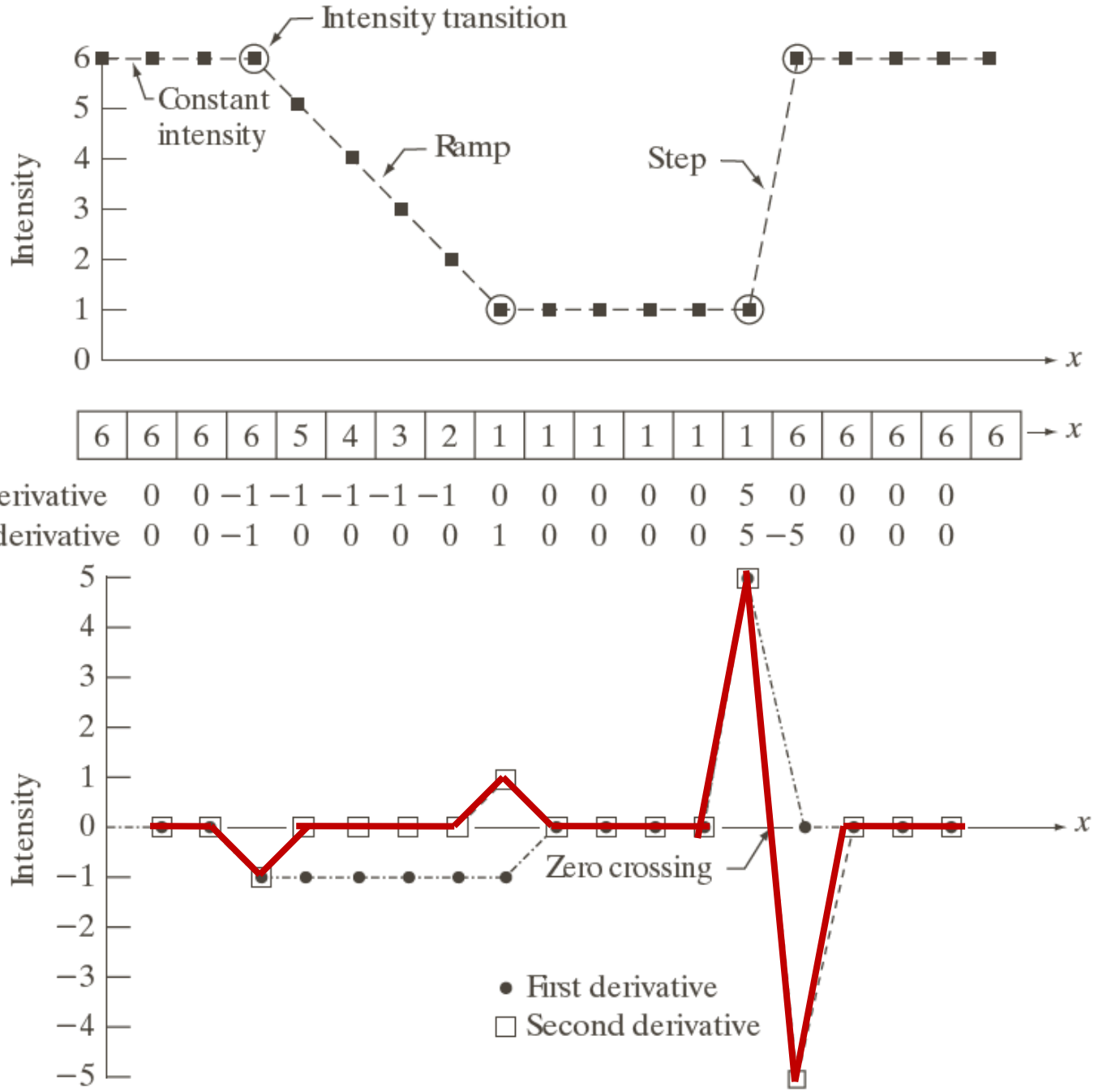
- The first-order derivative of a one-dimensional function f(x) is the difference

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

- The second-order derivative of f(x) as the difference

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

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



a
b
c

FIGURE 3.36 Illustration of the first and second derivatives of a 1-D digital function representing a section of a horizontal intensity profile from an image. In (a) and (c) data points are joined by dashed lines as a visualization aid.

Laplacian

- The second-order isotropic derivative operator is the Laplacian for a function (image) f(x,y)

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

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

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

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

Laplacian

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

x-1,y-1	x,y-1	x+1,y-1
x-1,y	x,y	x+1,y
x-1,y+1	x,y+1	x+1,y+1

0	1	0
1	4	1
0	1	0

Various Implementations of Laplacian

0	1	0
1	−4	1
0	1	0

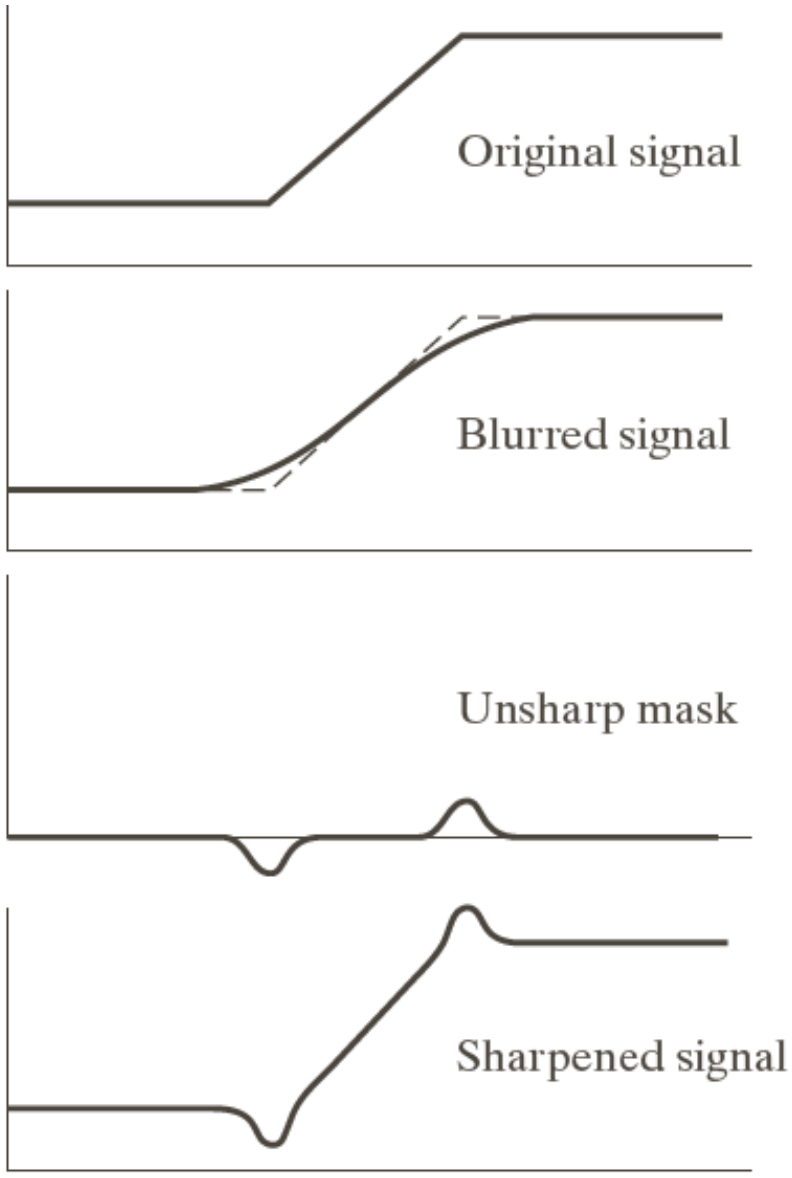
1	1	1
1	−8	1
1	1	1

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

a b
c d

FIGURE 3.37
(a) Filter mask used to implement Eq. (3.6-6). (b) Mask used to implement an extension of this equation that includes the diagonal terms. (c) and (d) Two other implementations of the Laplacian found frequently in practice.

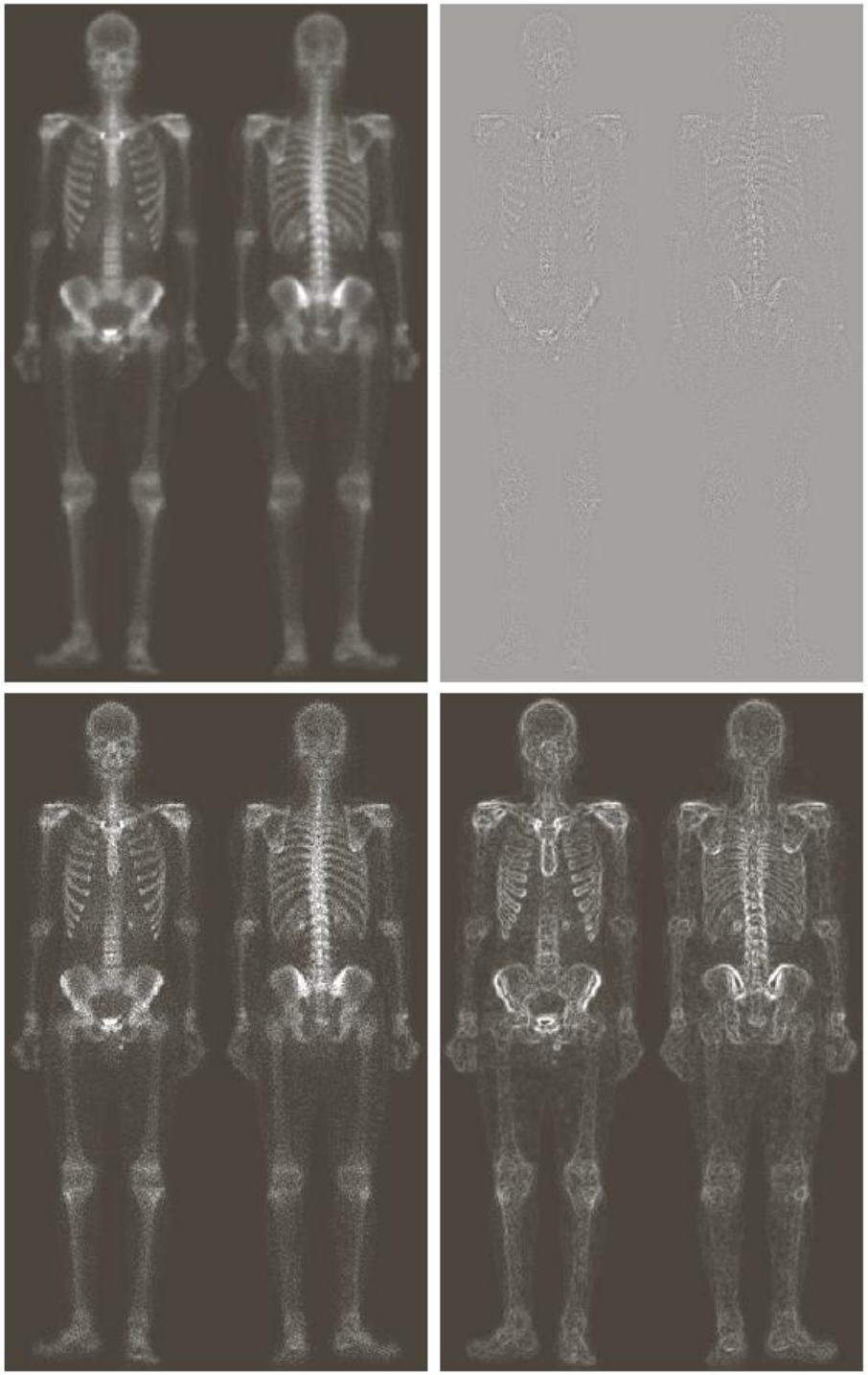


a
b
c
d

FIGURE 3.39 1-D illustration of the mechanics of unsharp masking. (a) Original signal. (b) Blurred signal with original shown dashed for reference. (c) Unsharp mask. (d) Sharpened signal, obtained by adding (c) to (a).

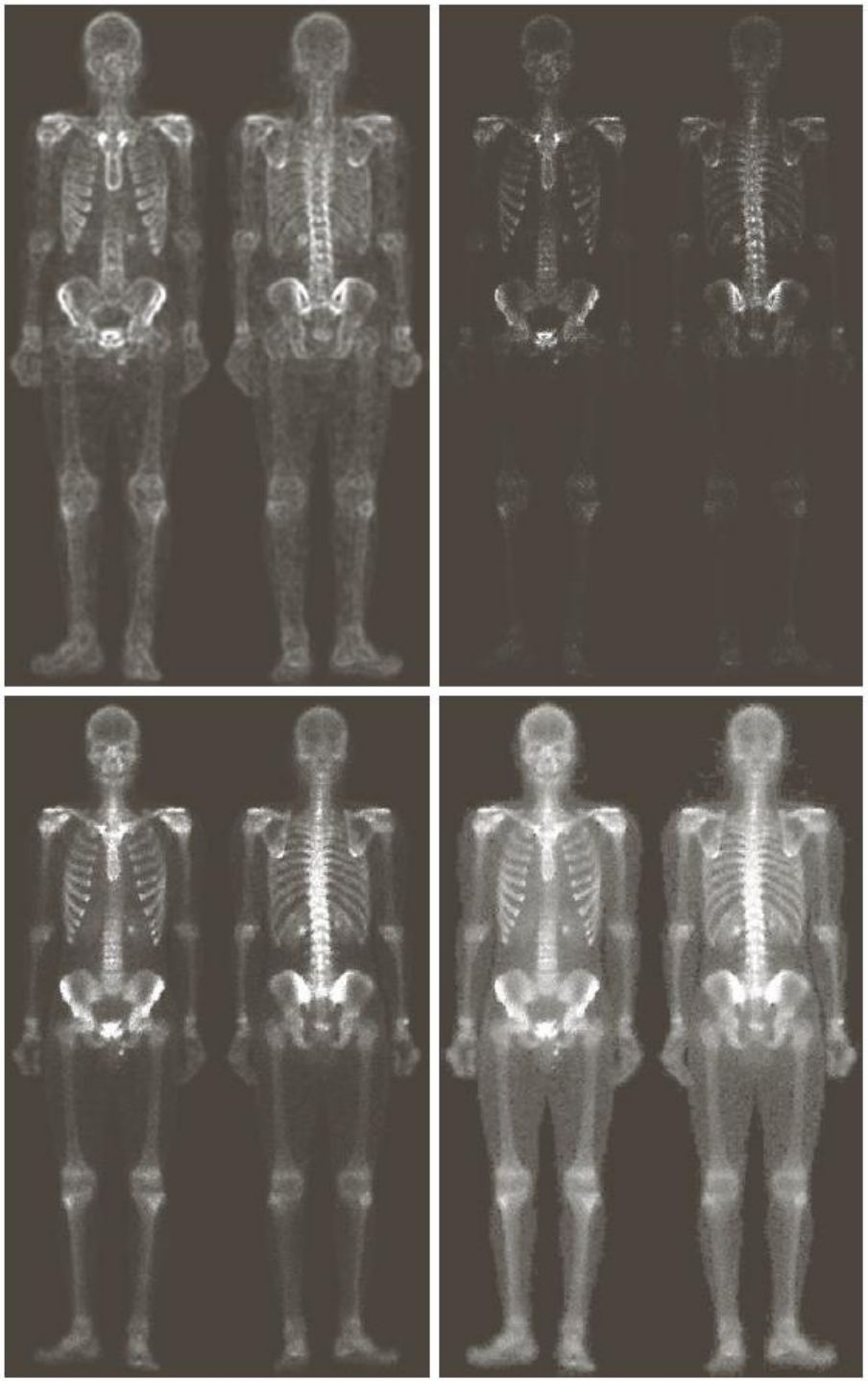
a
b
c
d
e

FIGURE 3.40
(a) Original image. (b) Result of blurring with a Gaussian filter. (c) Unsharp mask. (d) Result of using unsharp masking. (e) Result of using highboost filtering.



a b
c d

FIGURE 3.43
(a) Image of whole body bone scan.
(b) Laplacian of (a). (c) Sharpened image obtained by adding (a) and (b).
(d) Sobel gradient of (a).



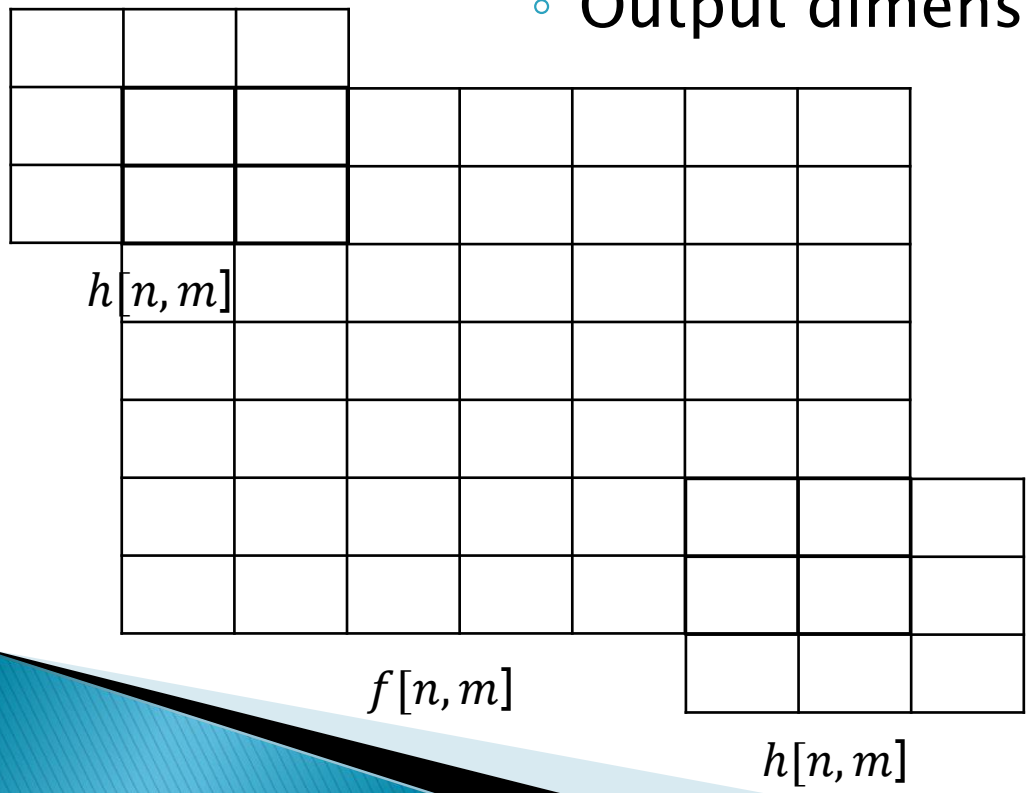
e f
g h

FIGURE 3.43
(Continued)
(e) Sobel image smoothed with a 5×5 averaging filter. (f) Mask image formed by the product of (c) and (e). (g) Sharpened image obtained by the sum of (a) and (f). (h) Final result obtained by applying a power-law transformation to (g). Compare (g) and (h) with (a). (Original image courtesy of G.E. Medical Systems.)

Boundary (Edge)

- Input and output dimension
  - Input dimension: (N_1, M_1)
  - Kernel dimension: $h(N_2, M_2)$
  - Output dimension:

$$(N_1+N_2-1, M_1+M_2-1)$$



Boundary (Edge)

- How can we handle “no pixel” area?
  - Zero padding
  - Boundary value repetition
  - Mirroring
  - ……