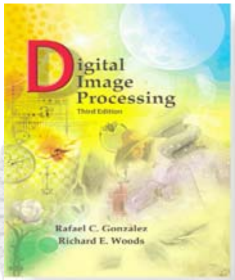


Lecture #6

- Neighborhood (spatial) transformations
- Convolution - moving window, shift-multiply-add
  - Correlation
  - Padding
  - Impulses
- Filters
  - Blurring
  - Median
  - Derivative — gradient, Laplacian, Sobel
- Image Sharpening, Unsharp masking

Spatial Neighborhoods

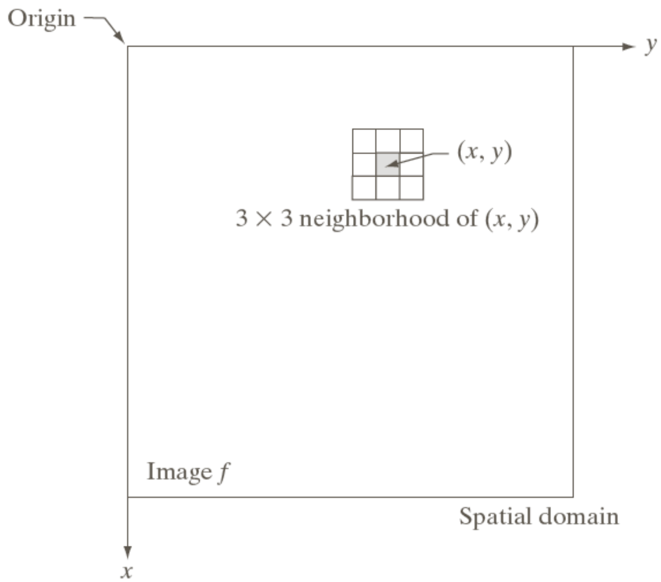


FIGURE 3.1 A 3 × 3 neighborhood about a point (x, y) in an image in the spatial domain. The neighborhood is moved from pixel to pixel in the image to generate an output image.

Spatial Filtering

Let I and J be images such that $J = T[I]$.

$T[\cdot]$ represents a transformation, such that,

$$J(r,c) = T[I](r,c) = f\left(\left\{ I(u,v) \;\middle|\; u \in \{r-s,\ldots,r,\ldots r+s\},\, v \in \{c-d,\ldots,c,\ldots c+d\} \right\}\right)$$

That is, the value of the transformed image, J, at pixel location (r,c) is a function of the values of the original image, I, in a $2s+1 \times 2d+1$ rectangular neighborhood centered on pixel location (r,c).

Moving Windows

- The value, $J(r,c) = \mathrm{T}[I](r,c)$, is a function of a rectangular neighborhood centered on pixel location (r,c) in I.
- There is a different neighborhood for each pixel location, but if the dimensions of the neighbor-hood are the same for each location, then trans-form T is sometimes called a *moving window transform*.

Moving-Window Transformations

Neutral Buoyancy Facility at NASA Johnson Space Center

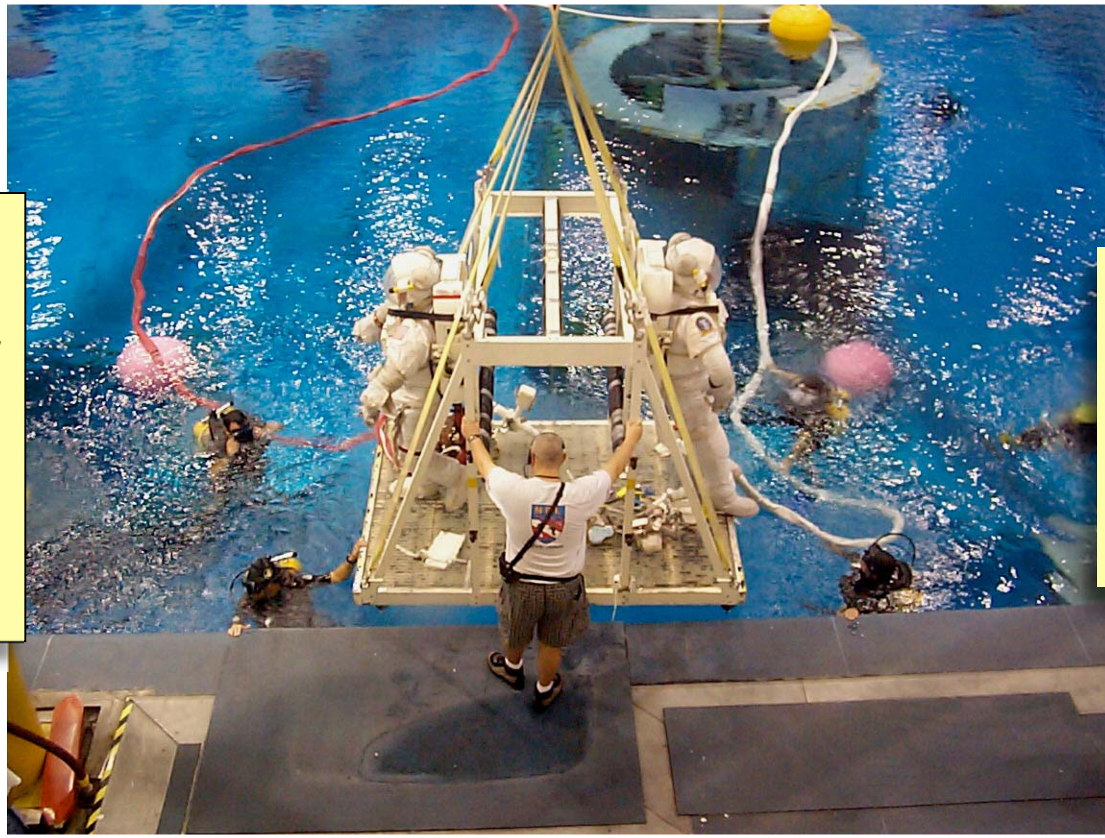
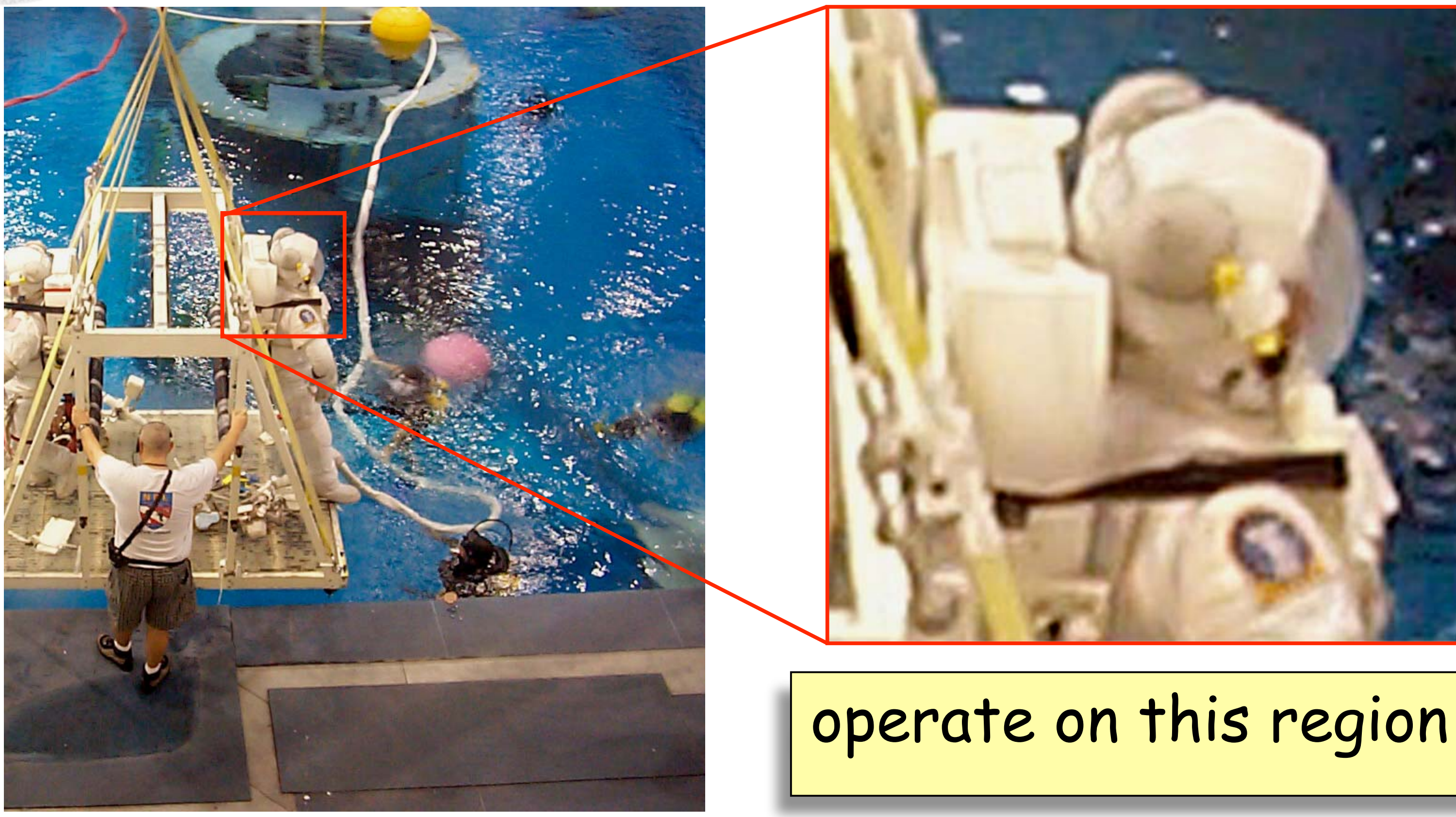


photo: R.A.Peters II, 1999

We'll take a section of this image to demonstrate the MWT

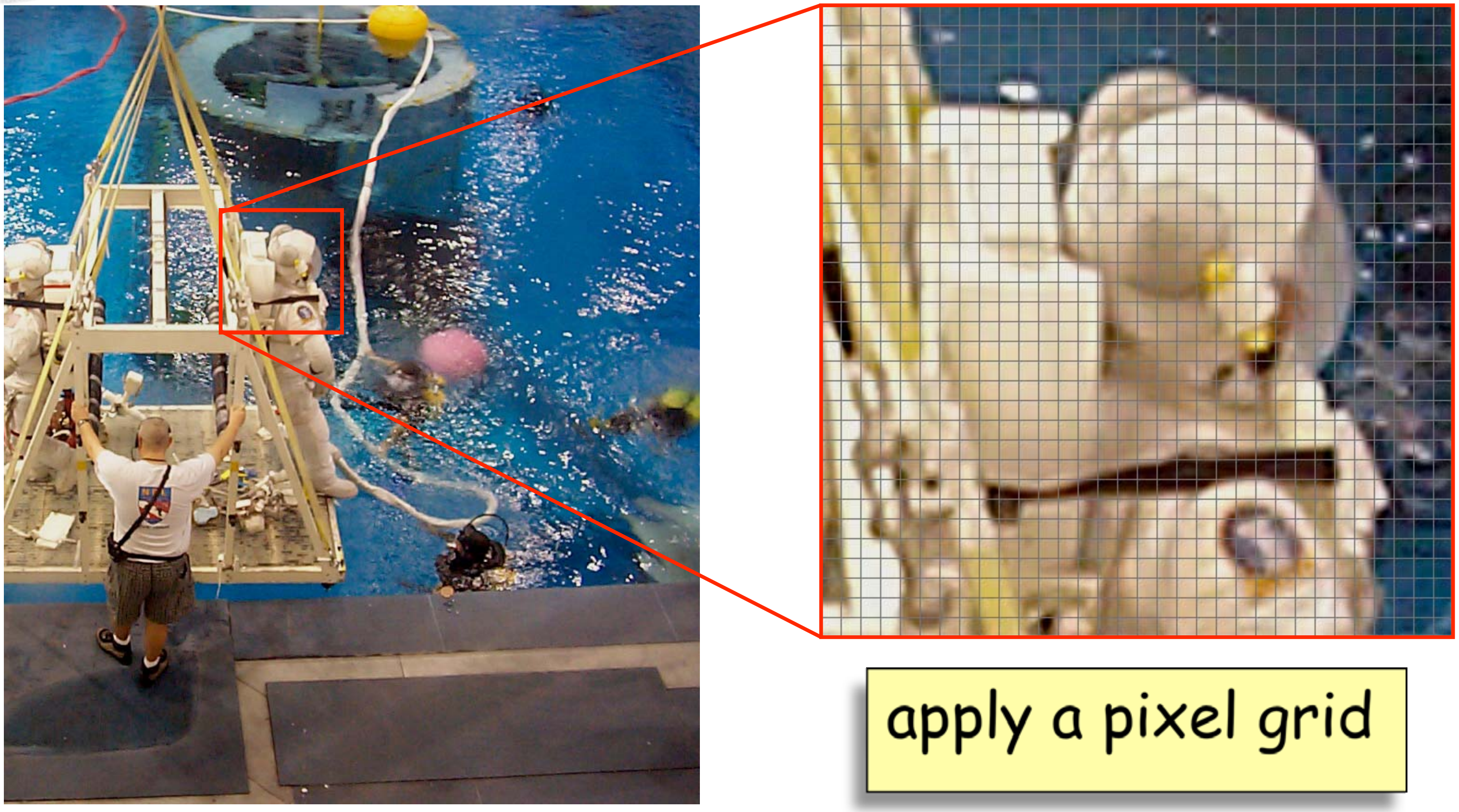
Moving-Window Transformations



operate on this region

Pixelize the section to better see the effects.

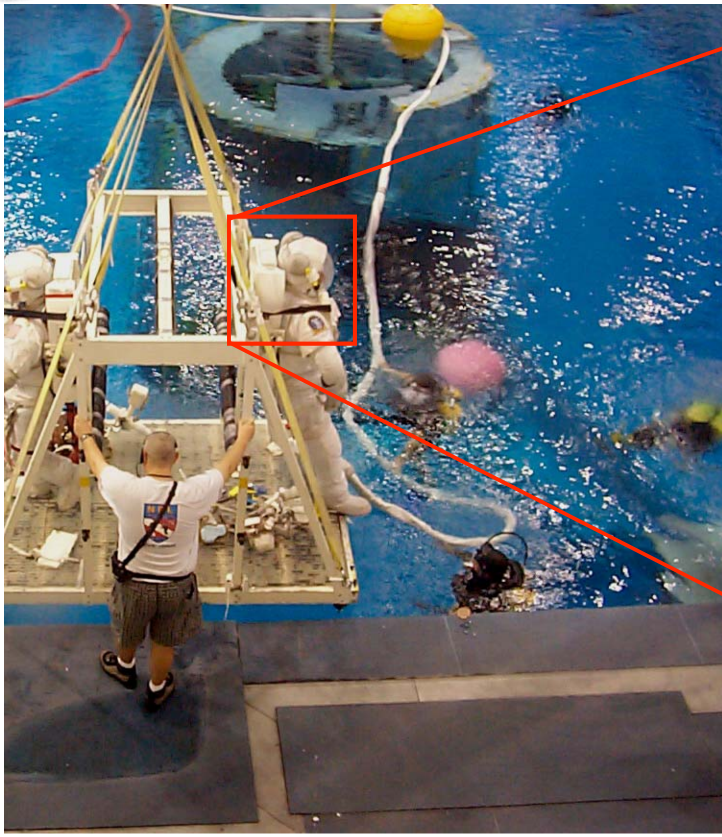
Moving-Window Transformations

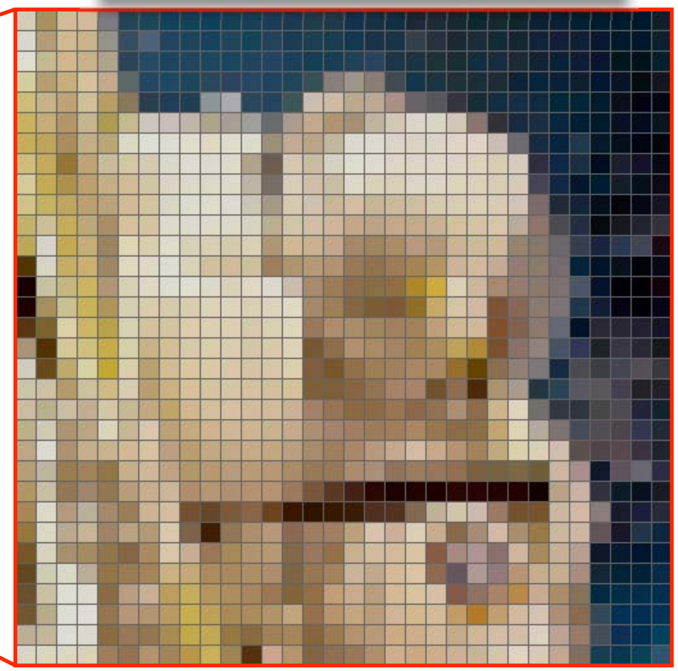


apply a pixel grid

Moving-Window Transformations

Pixelize the section to better see the effects.





sample (average in the squares).

Spatial Filtering

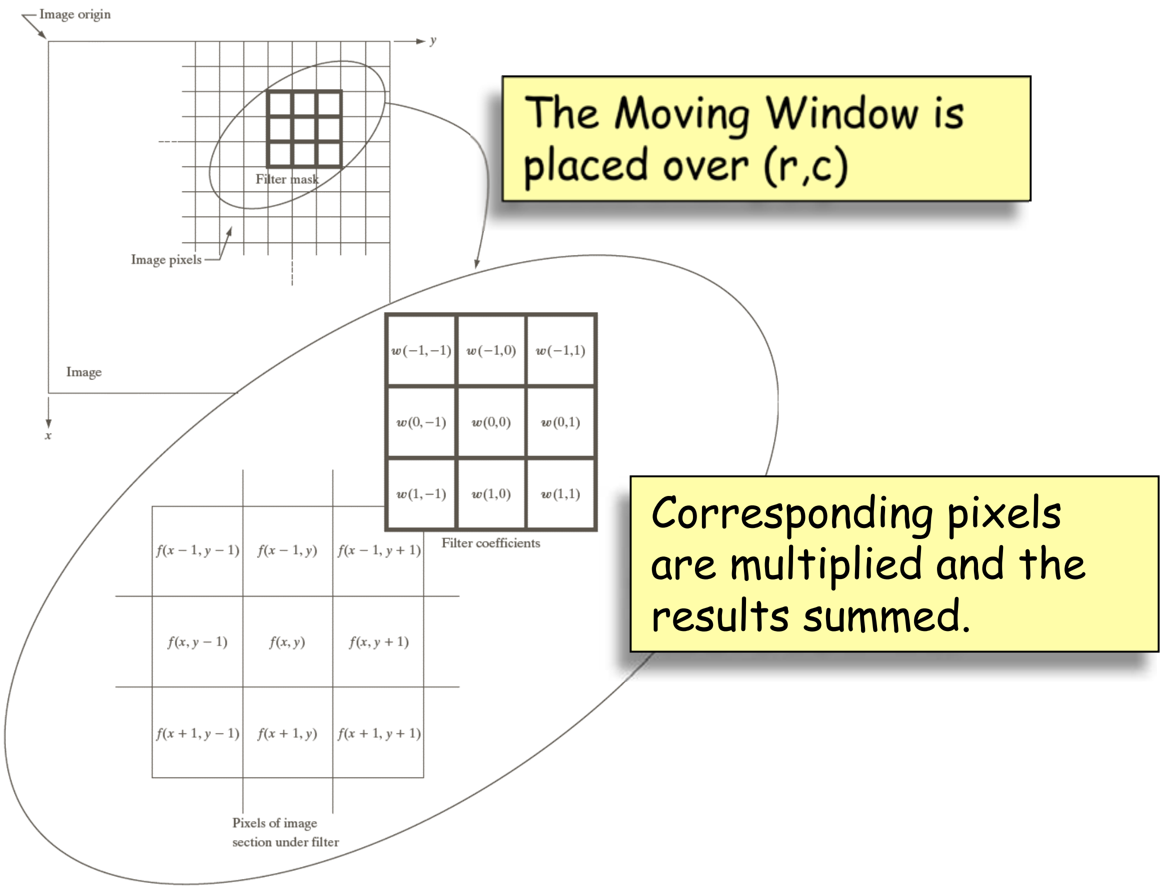


FIGURE 3.28 The mechanics of linear spatial filtering using a 3×3 filter mask. The form chosen to denote the coordinates of the filter mask coefficients simplifies writing expressions for linear filtering.

$\frac{1}{9} \times$

1	1	1
1	1	1
1	1	1

$\frac{1}{16} \times$

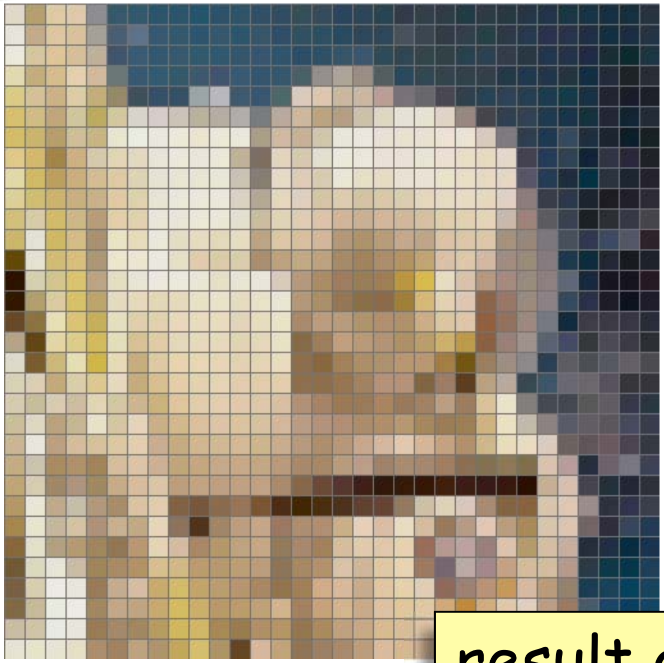
1	2	1
2	4	2
1	2	1

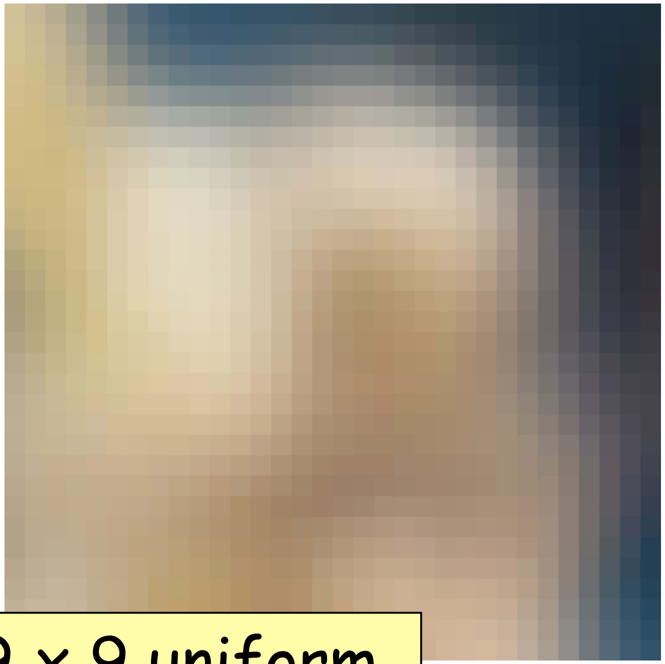
a b

FIGURE 3.32 Two 3×3 smoothing (averaging) filter masks. The constant multiplier in front of each mask is equal to 1 divided by the sum of the values of its coefficients, as is required to compute an average.

A normalization constant is necessary to prevent shifting image values out of range.

Moving-Window Transformations





result of a 9 x 9 uniform averaging window

Convolution: Mathematical Representation

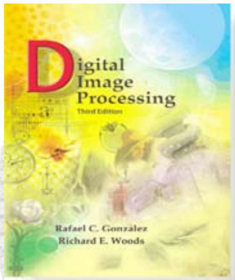
If a MW transformation is *linear* then it is a *convolution*:

$$J(r,c) = [I * h](r,c) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(r-\rho, c-\kappa)\, h(\rho,\kappa)\, d\rho\, d\kappa,$$

for an ideal, Euclidean image, or for a digital image:

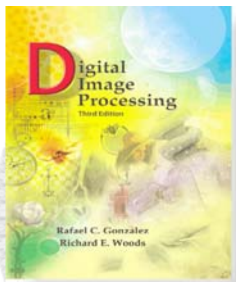
kernel

$$J(r,c) = [I * h](r,c) = \sum_{\rho=-\infty}^{\infty}\sum_{\kappa=-\infty}^{\infty} I(r-\rho, c-\kappa)\, h(\rho,\kappa)$$

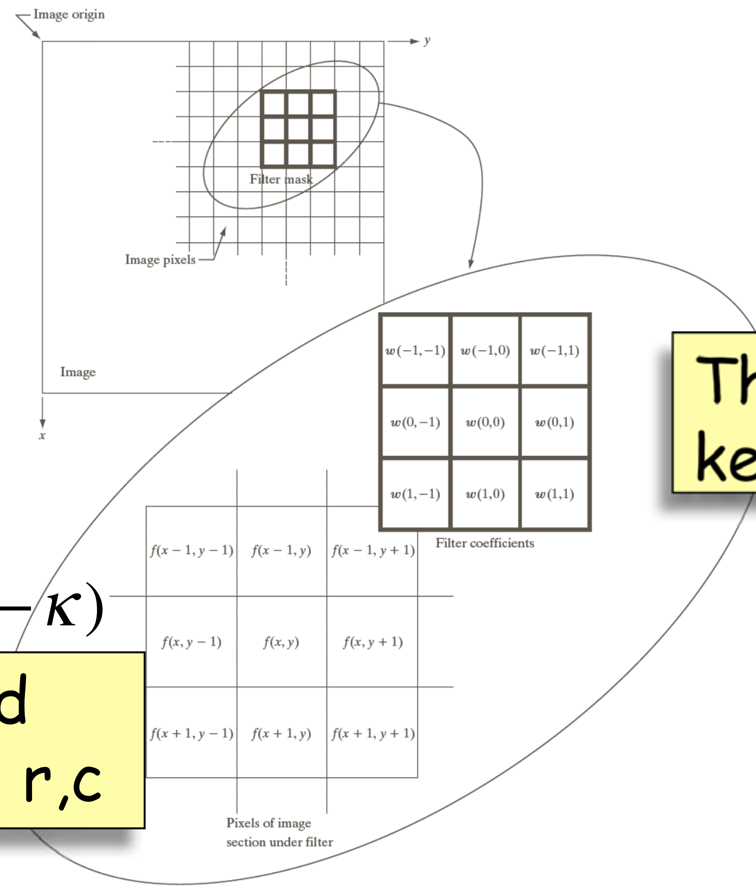


Convolution Mask (Weight Matrix)

- The object, $h(\rho,\kappa)$, in the equation is a weighting function, or in the discrete case, a rectangular matrix of numbers.
- The matrix is the moving window.
- Pixel (r,c) in the output image is the weighted sum of pixels from the original image in the neighborhood of (r,c) traced by the matrix.
- Each pixel in the neighborhood of (r,c) is multiplied by the corresponding matrix value — <u>after the matrix is rotated by 180º</u>.
- The sum of those products is the value of pixel (r,c) in the output image



Spatial Filtering



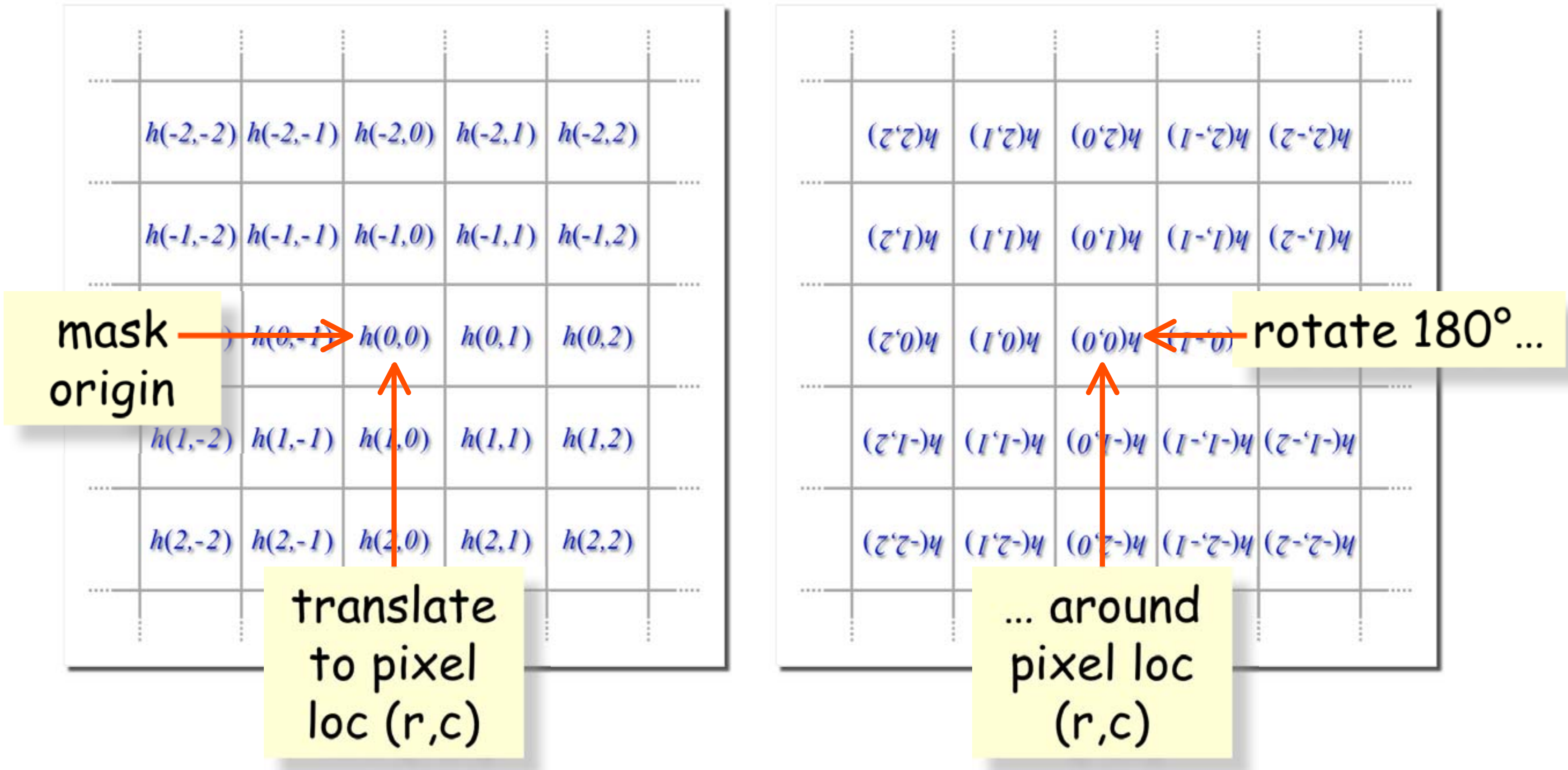
The mask is the kernel h(ρ,κ)

$$I(r-\rho,c-\kappa)$$

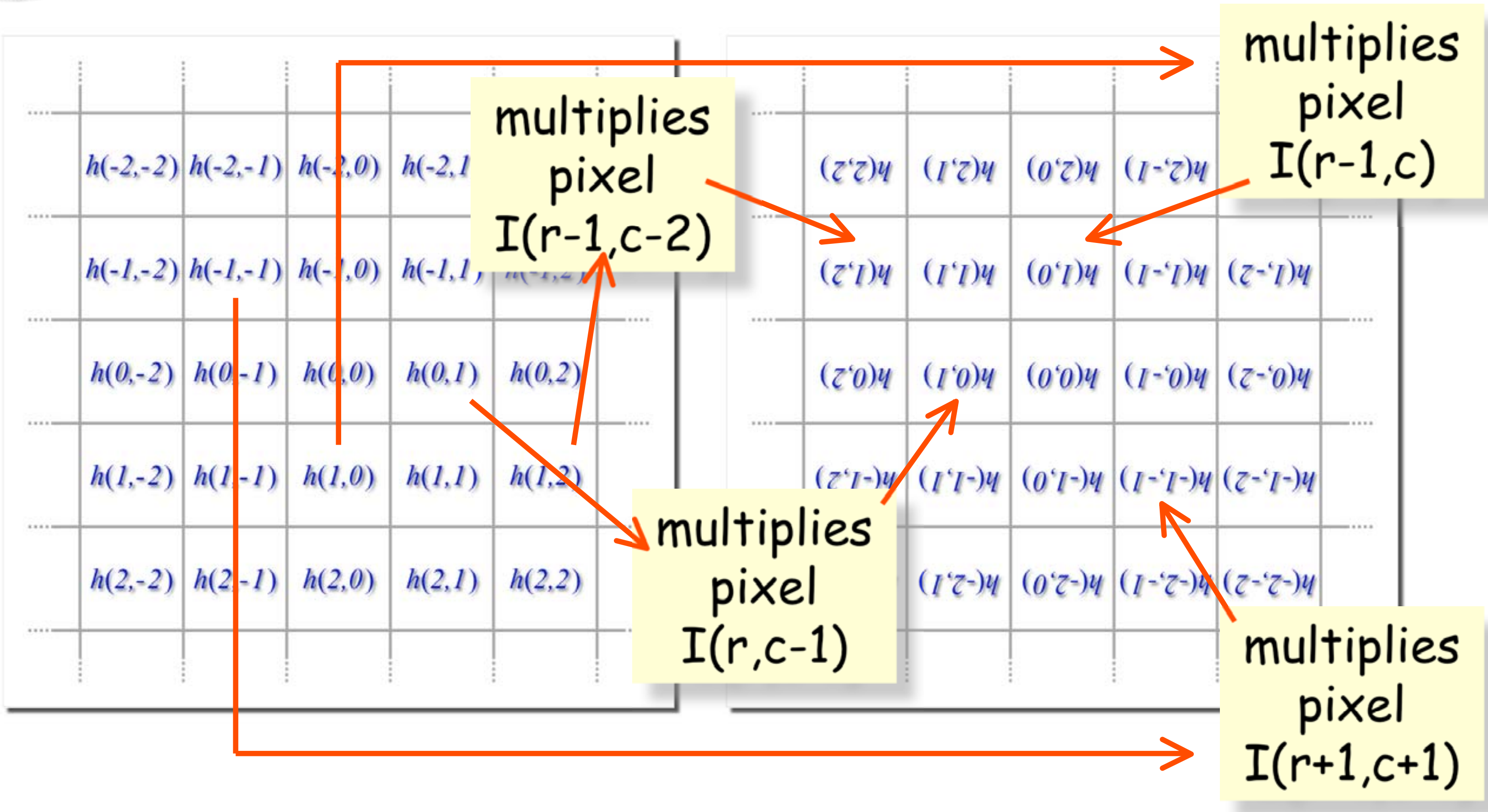
The neighborhood I(r-ρ,c-κ) around r,c

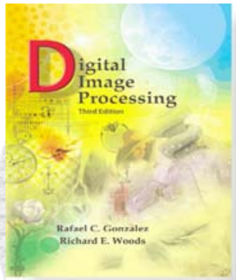
FIGURE 3.28 The mechanics of linear spatial filtering using a 3 × 3 filter mask. The form chosen to denote the coordinates of the filter mask coefficients simplifies writing expressions for linear filtering.

Convolution Masks: Moving Window

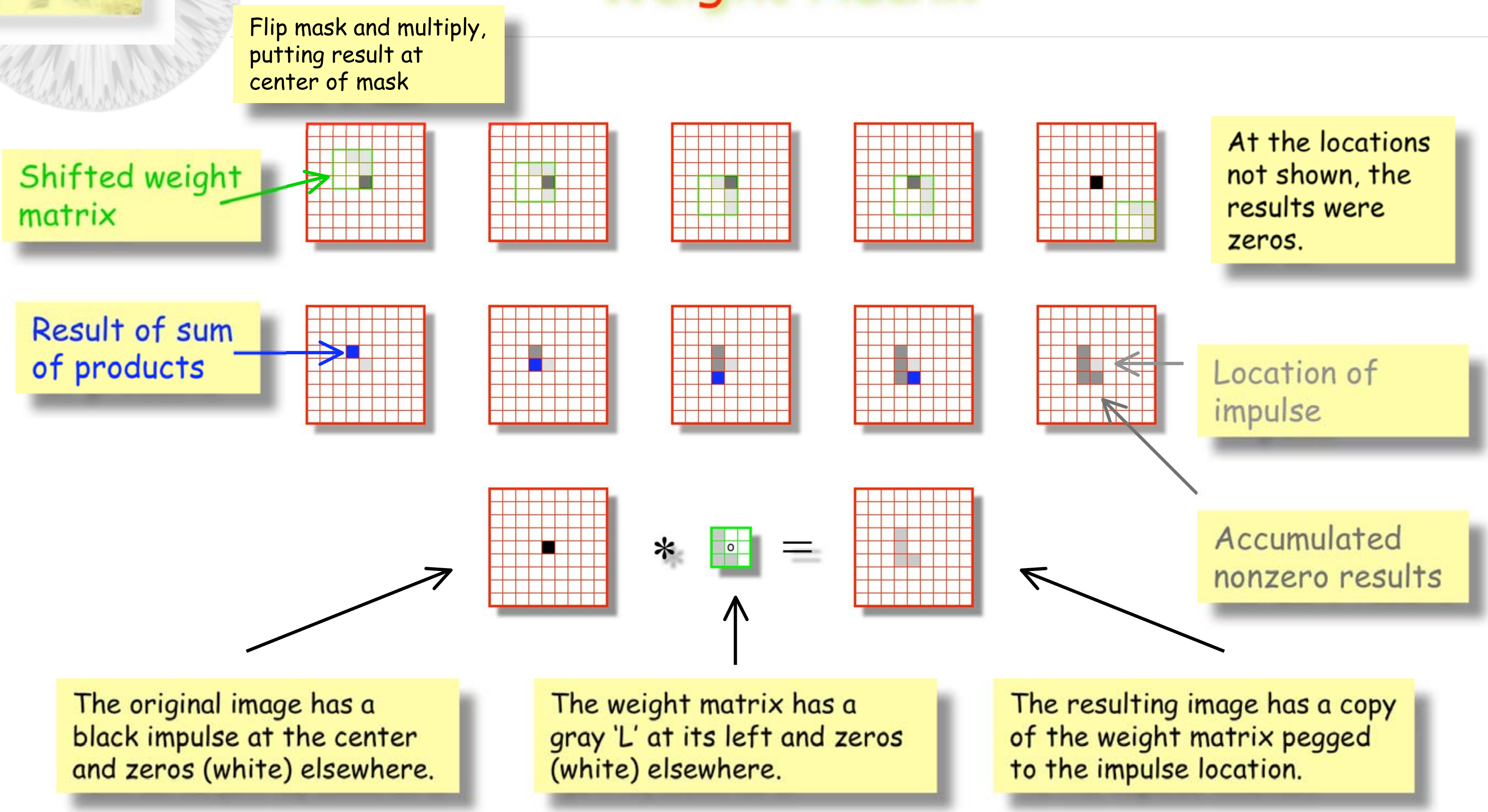


Convolution Masks: Moving Window





Convolution by Rotating and Shifting the Weight Matrix



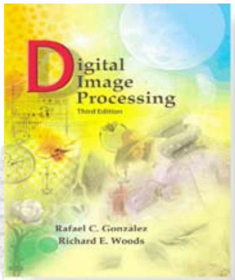
Correlation: Mathematical Representation

A *correlation* looks almost identical to a *convolution*:

$$J(r,c) = [I \star h](r,c) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(r+\rho, c+\kappa)h(\rho,\kappa)\,d\rho\, d\kappa\,,$$

for an ideal, Euclidean image, or for a digital image:

$$J(r,c) = [I \star h](r,c) = \sum_{\rho=-\infty}^{\infty}\sum_{\kappa=-\infty}^{\infty} I(r+\rho, c+\kappa)h(\rho,\kappa)$$



1-D Comparison of Correlation & Convolution

Correlation — not flipped

(a) Origin, f: 0 0 0 1 0 0 0 0 w: 1 2 3 2 8

(b) 0 0 0 1 0 0 0 0
1 2 3 2 8
Starting position alignment

Pad image with zeros so mask is multiplying something.

(c) Zero padding
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 2 3 2 8

(d) 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 2 3 2 8
Position after one shift

(e) 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 2 3 2 8
Position after four shifts

(f) 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 2 3 2 8
Final position

(g) Full correlation result
0 0 0 8 2 3 2 1 0 0 0 0

(h) Cropped correlation result
0 8 2 3 2 1 0 0

Crop to remove padding.

Convolution — flipped

(i) Origin, f: 0 0 0 1 0 0 0 0 w rotated 180°: 8 2 3 2 1

(j) 0 0 0 1 0 0 0 0
8 2 3 2 1

(k) 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
8 2 3 2 1

(l) 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
8 2 3 2 1

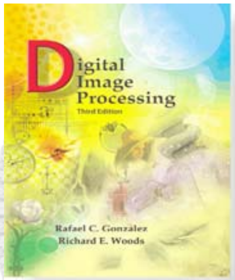
(m) 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
8 2 3 2 1

(n) 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
8 2 3 2 1

(o) Full convolution result
0 0 0 1 2 3 2 8 0 0 0 0

(p) Cropped convolution result
0 1 2 3 2 8 0 0

FIGURE 3.29 Illustration of 1-D correlation and convolution of a filter with a discrete unit impulse. Note that correlation and convolution are functions of *displacement*.



2-D Comparison of Correlation & Convolution

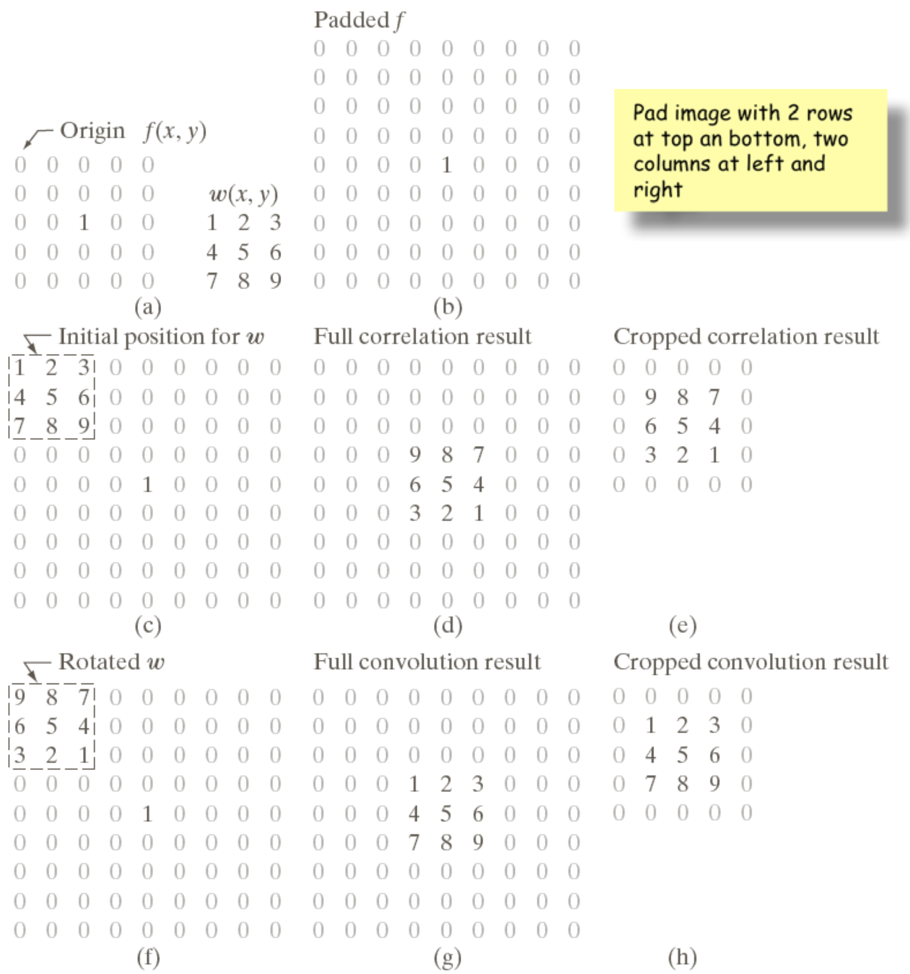


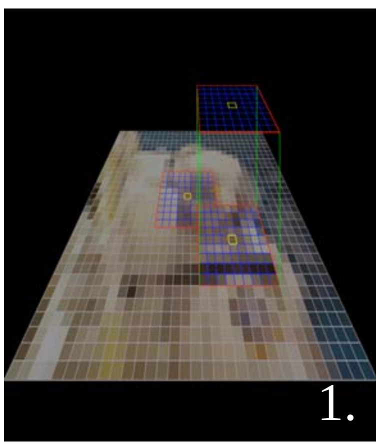
FIGURE 3.30 Correlation (middle row) and convolution (last row) of a 2-D filter with a 2-D discrete, unit impulse. The 0s are shown in gray to simplify visual analysis.

Convolution and Correlation

- Convolution is associated with filtering
  - –The masks look like impulse responses
- Correlation is associated with pattern recognition
  - –The masks look like objects to be found

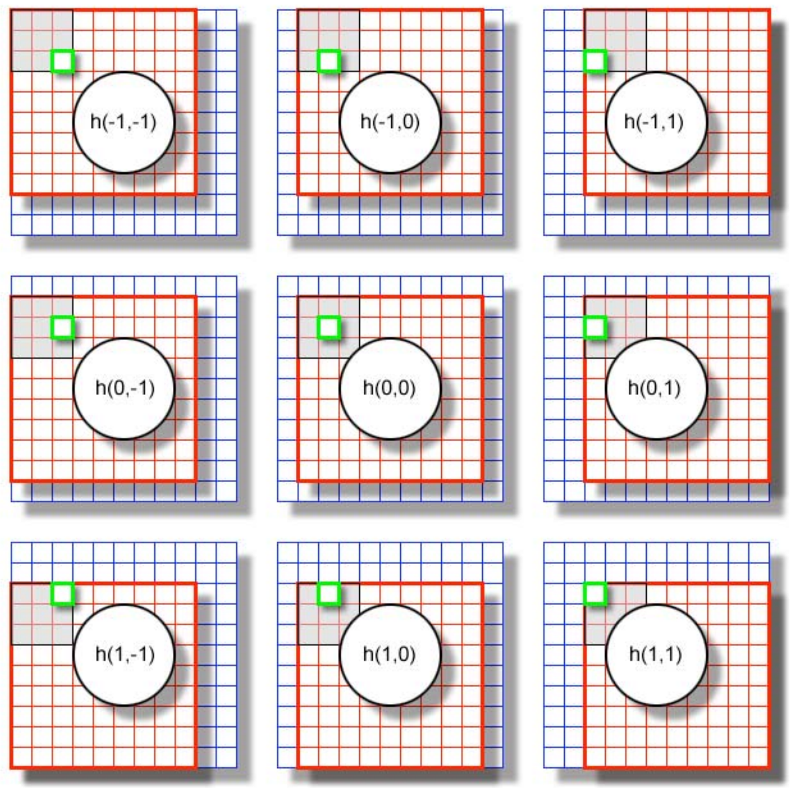
Three ways to compute a convolution

1. Moving window transform as just described.
2. Shift multiply add.
3. Fourier transform.



Convolution by Copying, Multiplying, and Shifting the Image

For each element $h(r_h,c_h)$ in weight matrix, h, image I is copied into a zero-padded image, P, starting at (r_h,c_h). Each P is multiplied by the corresponding weight, $h(r_h,c_h)$. All the P images are summed pixel-wise then divided by the sum of the elements of h. The result is cropped out of the center of the accumulated P's.



original image, I

padded image, P

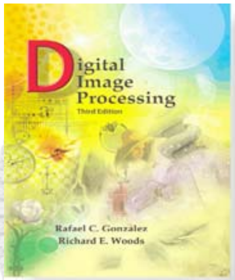
effective neighborhood

h(-1,-1)	h(-1,0)	h(-1,1)
h(0,-1)	h(0,0)	h(0,1)
h(1,-1)	h(1,0)	h(1,1)

weight matrix

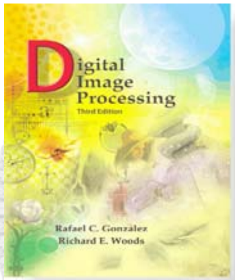
aligned pixels to be summed

weight for image ○



Shift-Multiply-Add Approach

- The image is copied 1 time for each element in the convolution mask.
- Each copy is shifted relative to the original by the displacement of its associated mask element.
- Each copy is multiplied by the value of its associated mask element.
- The set of shifted and multiplied images is summed pixel wise.



Convolution by an Impulse

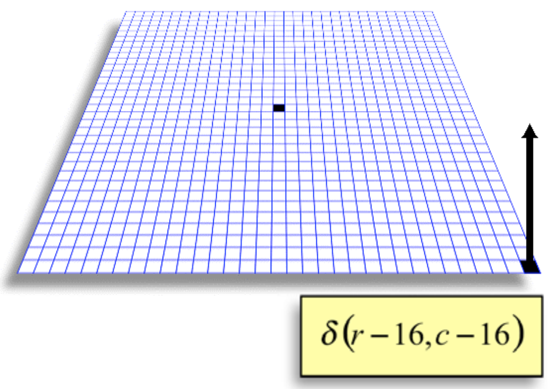
An *impulse* is a digital image, that has a single pixel with value 1; all others have value zero. An impulse at location (ρ, χ) is represented by:

$$\delta(r-\rho, c-\chi) = \begin{cases} 1, & \text{if } \; r=\rho \; \text{ and } \; c=\chi \\ 0, & \text{otherwise} \end{cases}$$

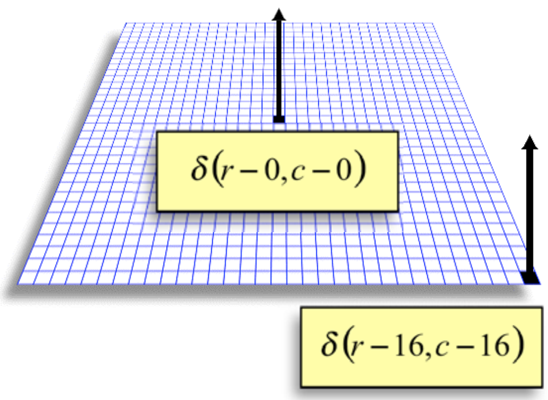
If an image is convolved with an impulse at location (ρ, χ), the image is shifted in location down by r pixels and to the right by χ pixels.

$$[I * \delta(r-\rho, c-\chi)](r,c) = I(r-\rho, c-\chi)$$

Convolution by an Impulse

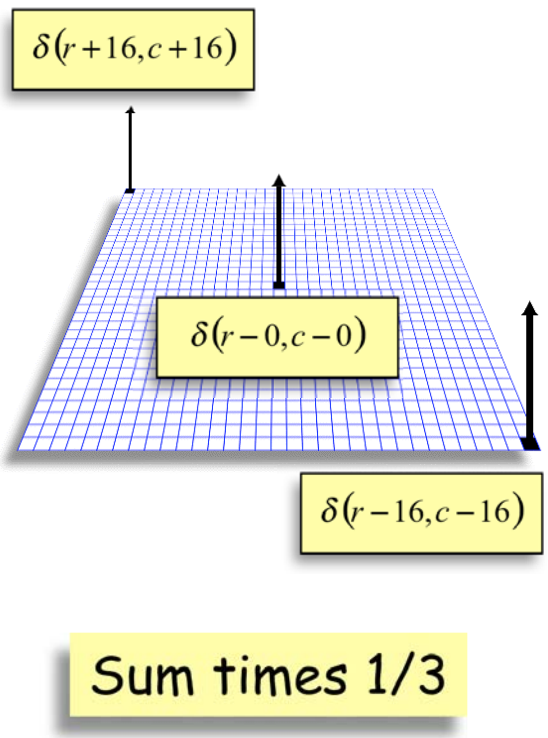


Convolution by Two Impulses

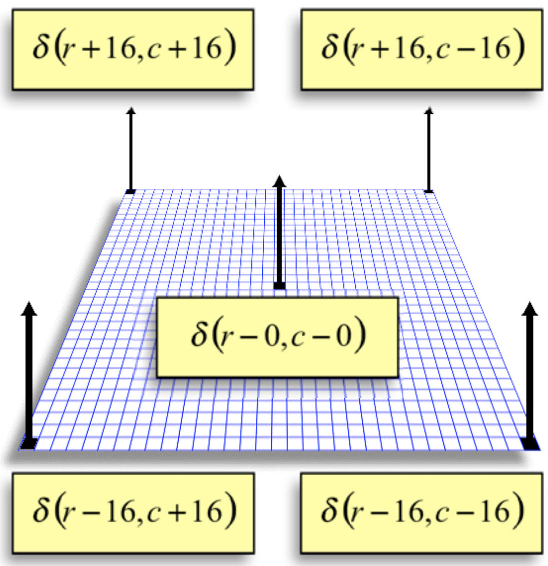


Sum times 1/2

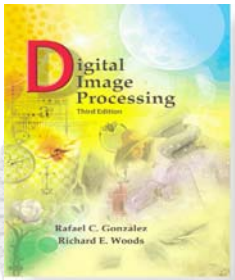
Convolution by Three Impulses



Convolution by Five Impulses



Sum times 1/5



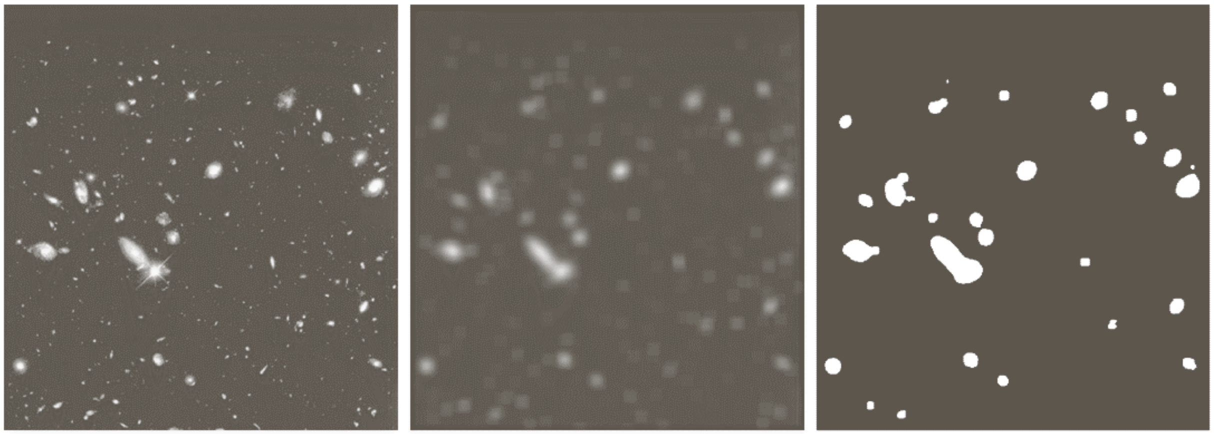
Effects of Spatial Averaging

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.

a b
c d
e f

Averaging filters are also called blur filters

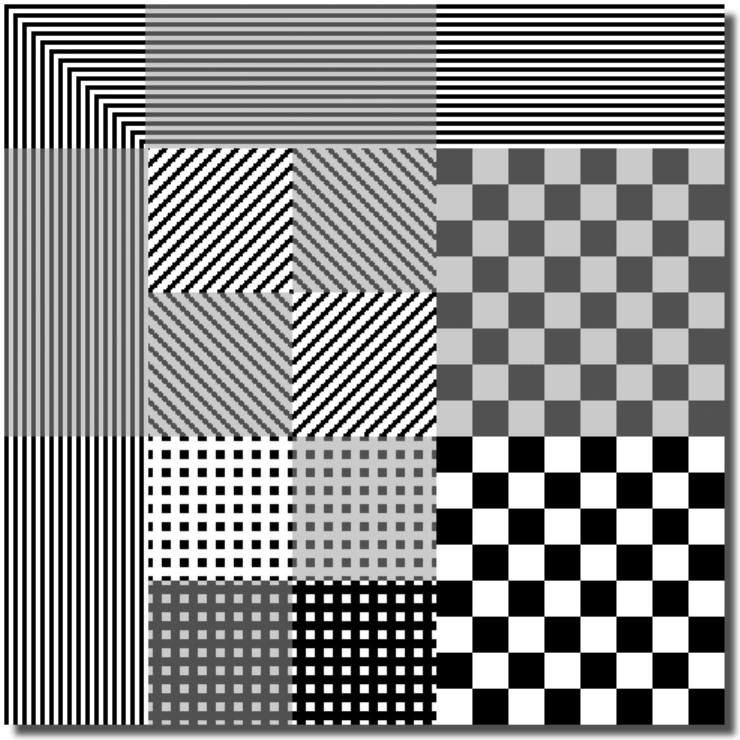
Averaging and Thresholding



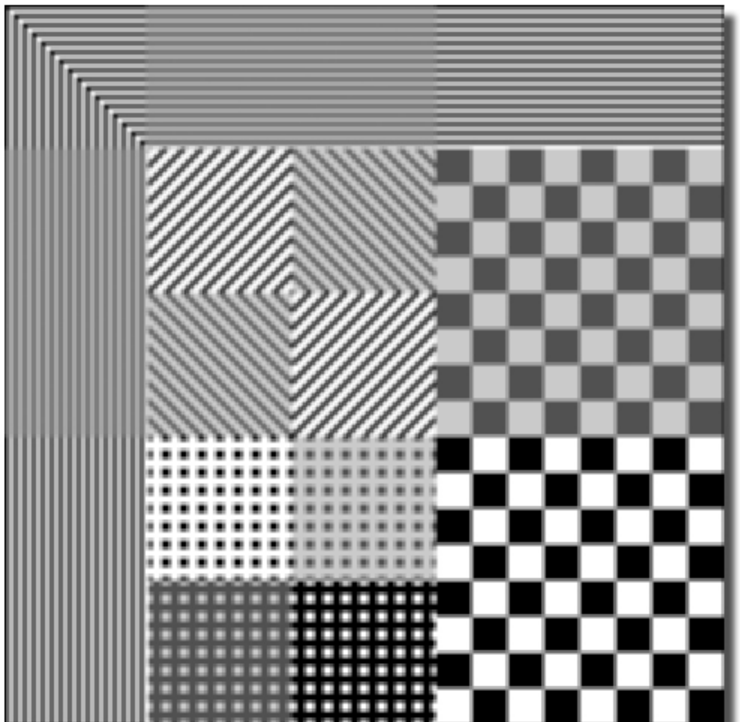
a b c

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.)

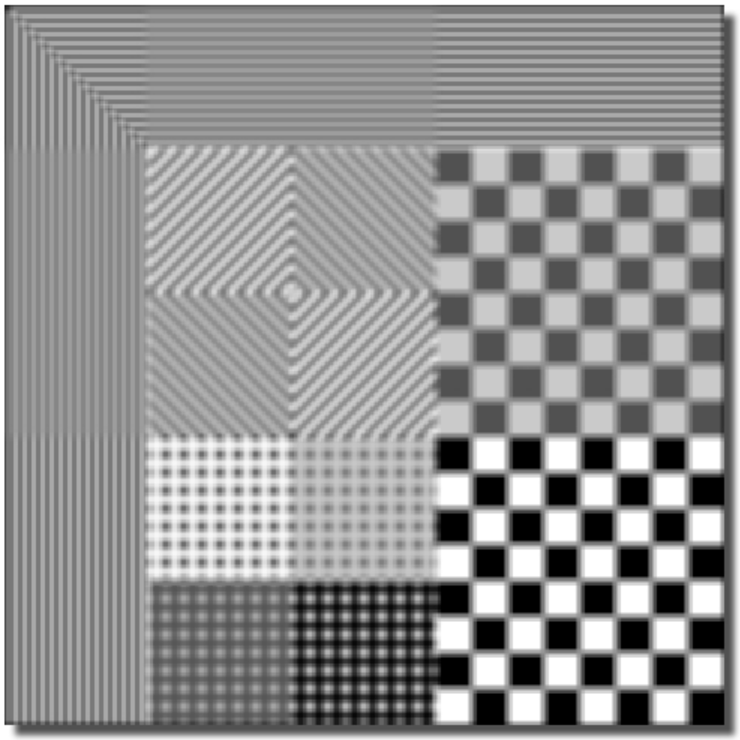
Convolution Examples: Original Images

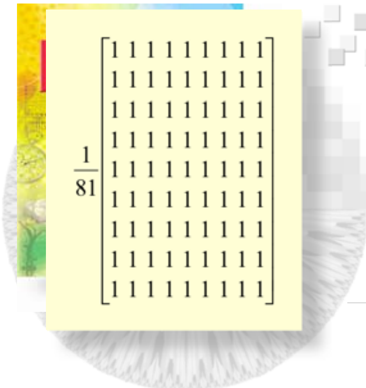


Convolution Examples: 3×3 Blur

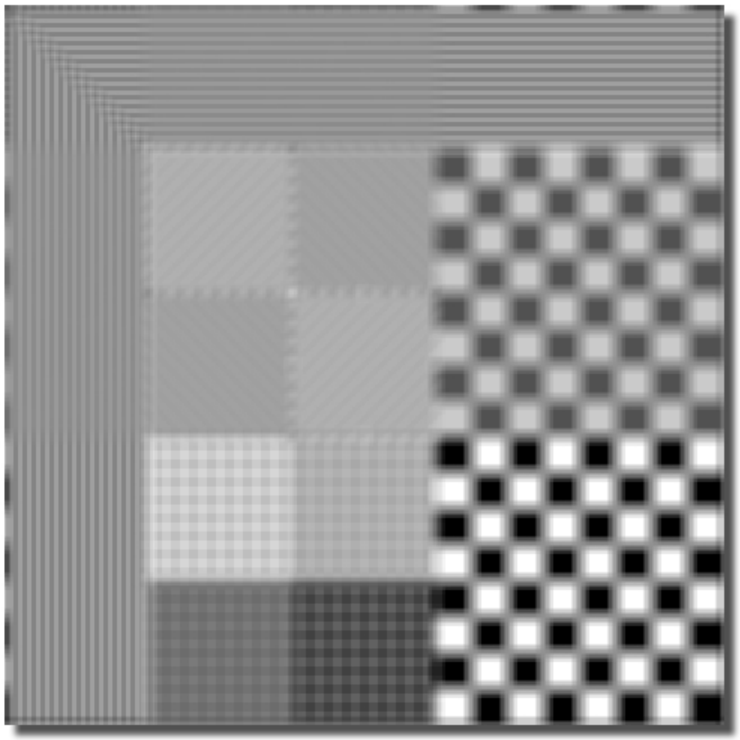


Convolution Examples: 5×5 Blur

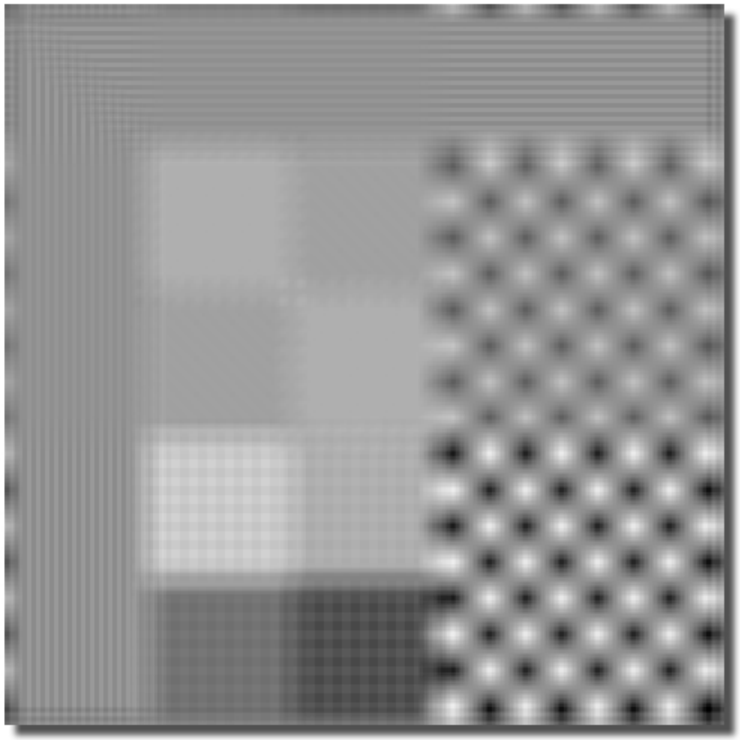




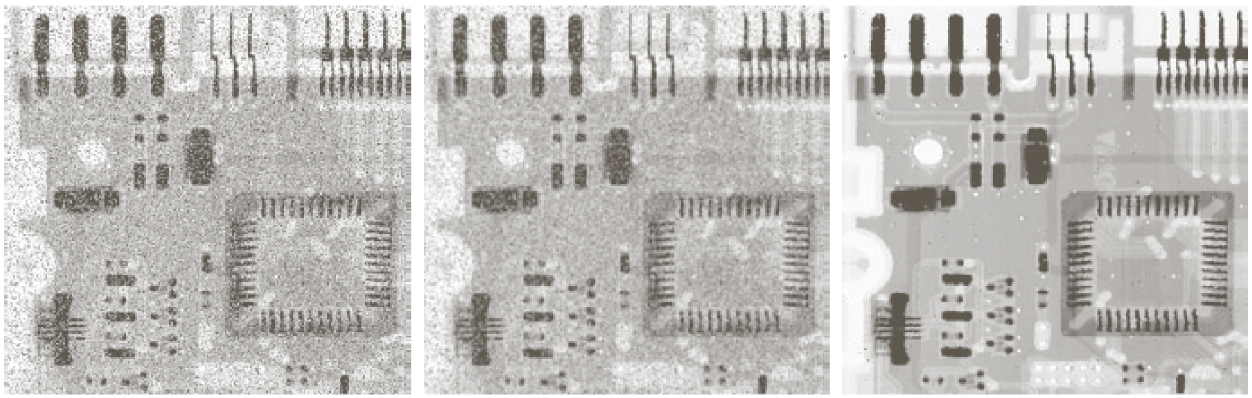
Convolution Examples: 9×9 Blur



Convolution Examples: 17×17 Blur



Noise: average v. median



a b c

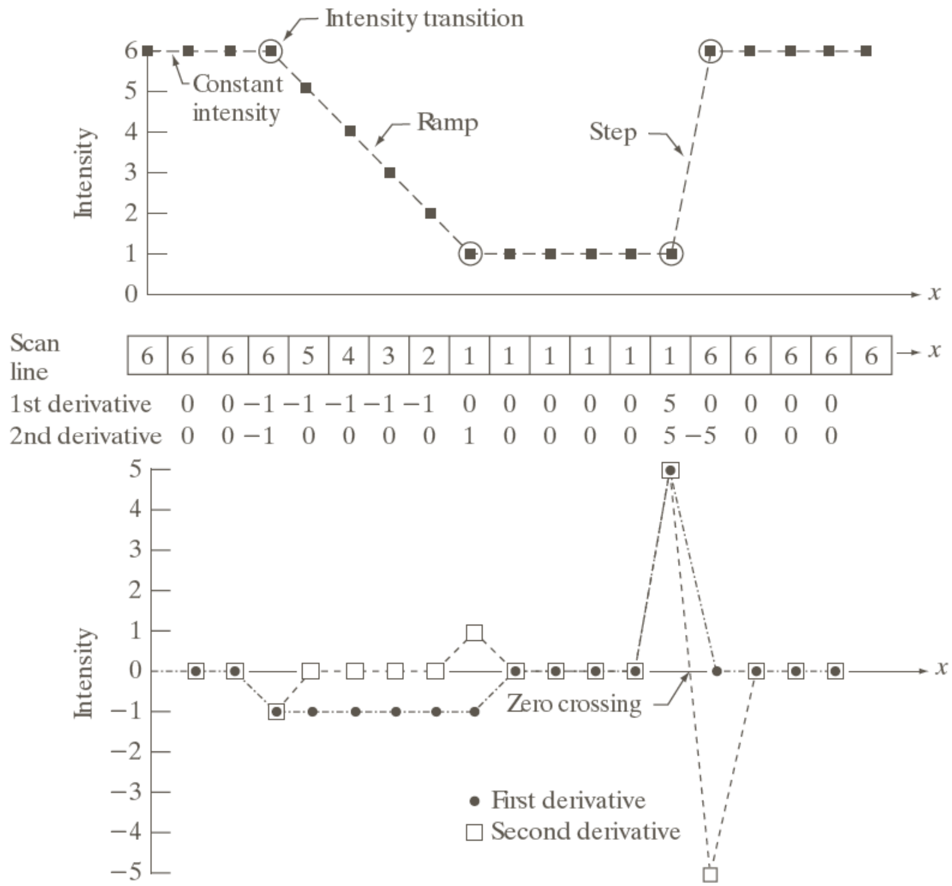
FIGURE 3.35 (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a 3×3 averaging mask. (c) Noise reduction with a 3×3 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

The median selects the middle value of a distribution and is good for distributions with outliers

The derivative in Image Processing

$$\frac{df}{dx} = \frac{f(x+\Delta x) - f(x)}{\Delta x} = f(x+1) - f(x)$$

$$\frac{d^2 f}{dx^2} = \frac{\frac{df(x+1)}{dx} - \frac{df(x-1)}{dx}}{1}$$
$$= f(x+1) - f(x) - f(x) + f(x-1)$$
$$= f(x+1) - 2f(x) + f(x-1)$$



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.

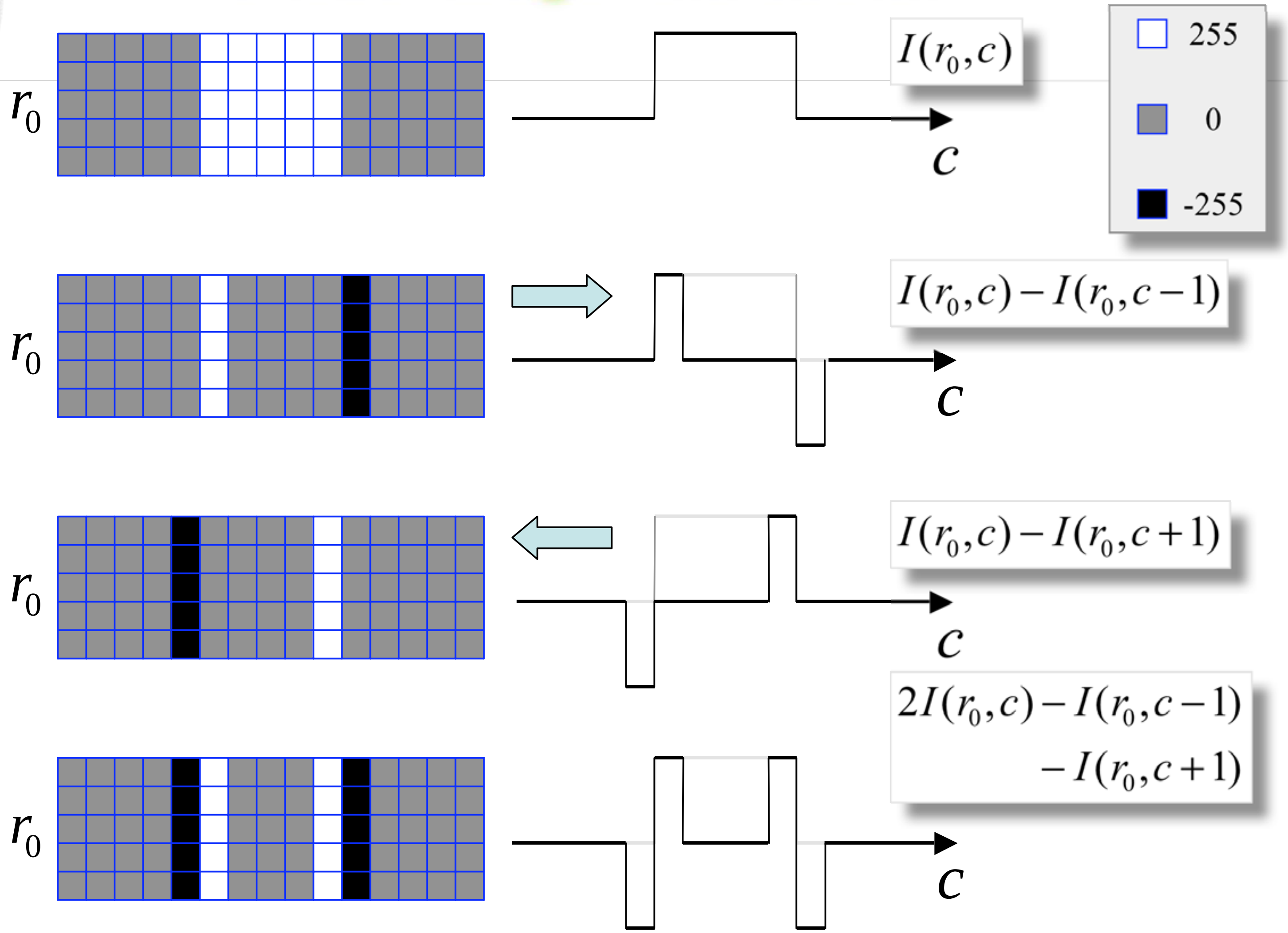
Vertical Edge Detection

Image

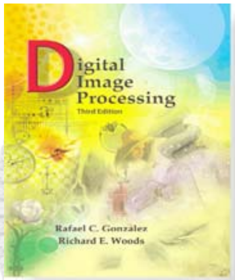
Backward Difference

Forward Difference

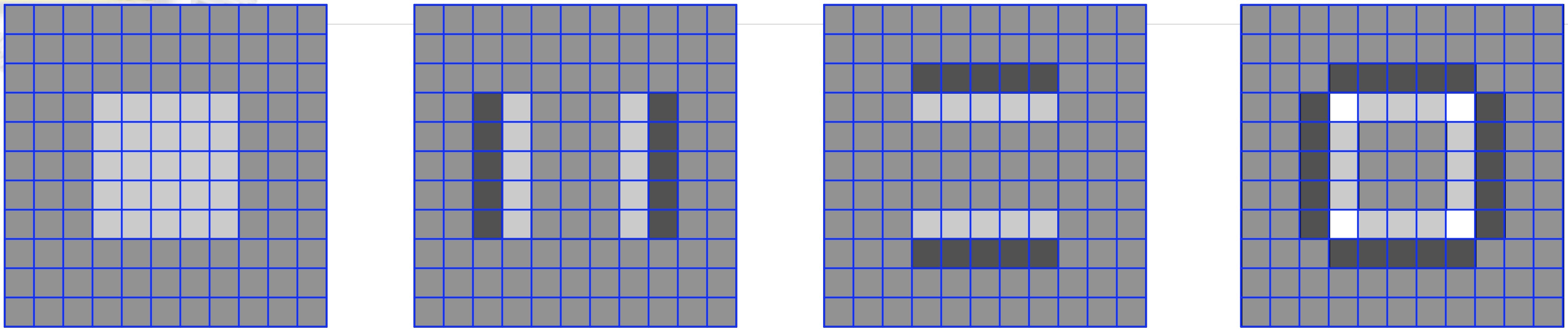
Sum of Differences



This is effectively a 2nd derivative



Symmetric Edge Detection



$$I(r,c)$$

$$2I(r,c) - I(r,c-1) - I(r,c+1)$$

$$2I(r,c) - I(r-1,c) - I(r+1,c)$$

$$4I(r,c) - I(r-1,c) - I(r+1,c) - I(r,c-1) - I(r,c+1)$$

510
255
0
-255

-1	2	-1

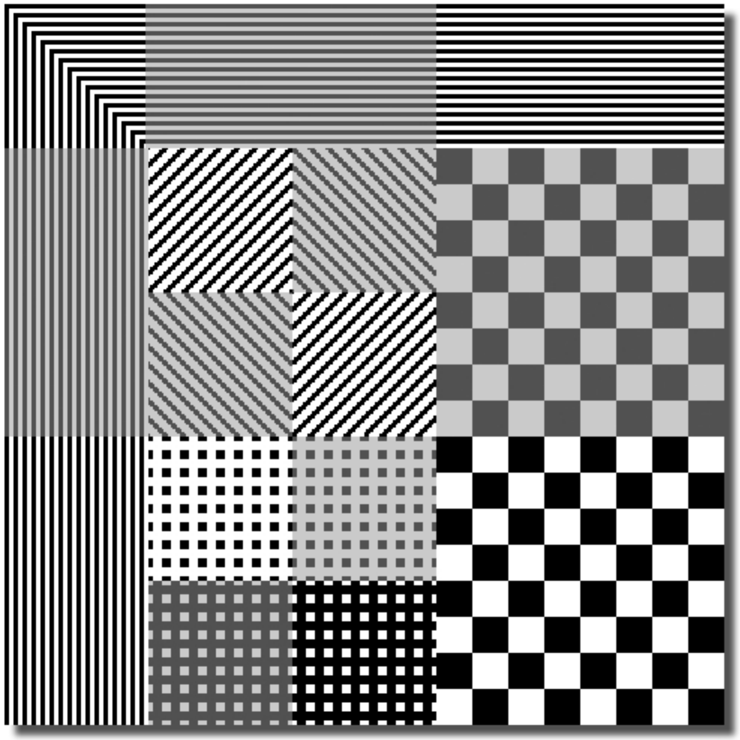
	-1	
	2	
	-1	

	-1	
-1	4	-1
	-1	

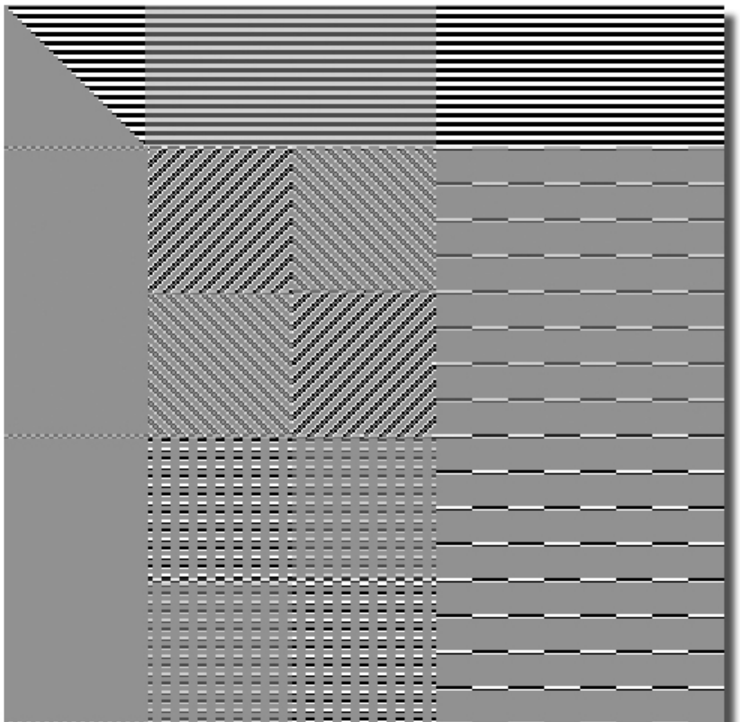
Express 2^{nd} derivatives in x and y as a mask

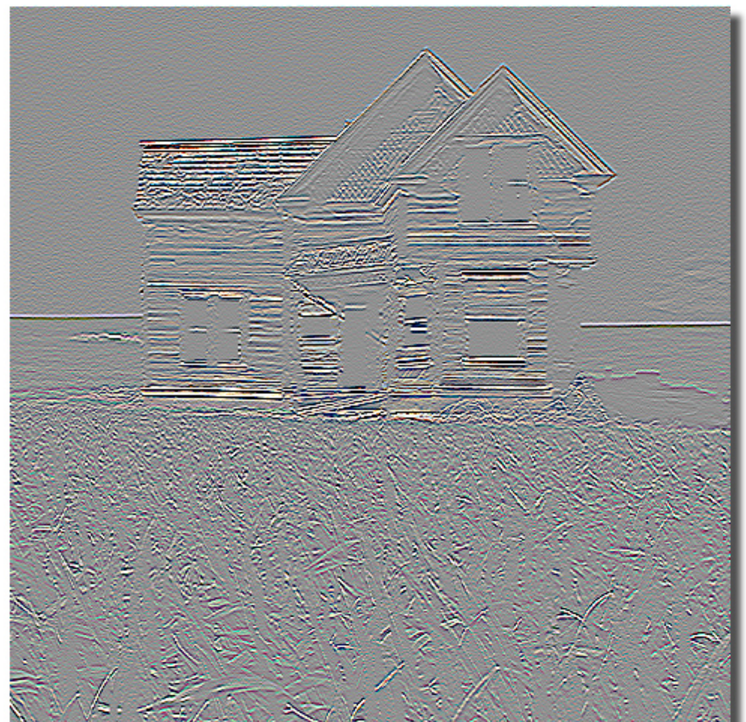
Combine x and y derivates

Convolution Examples: Original Images



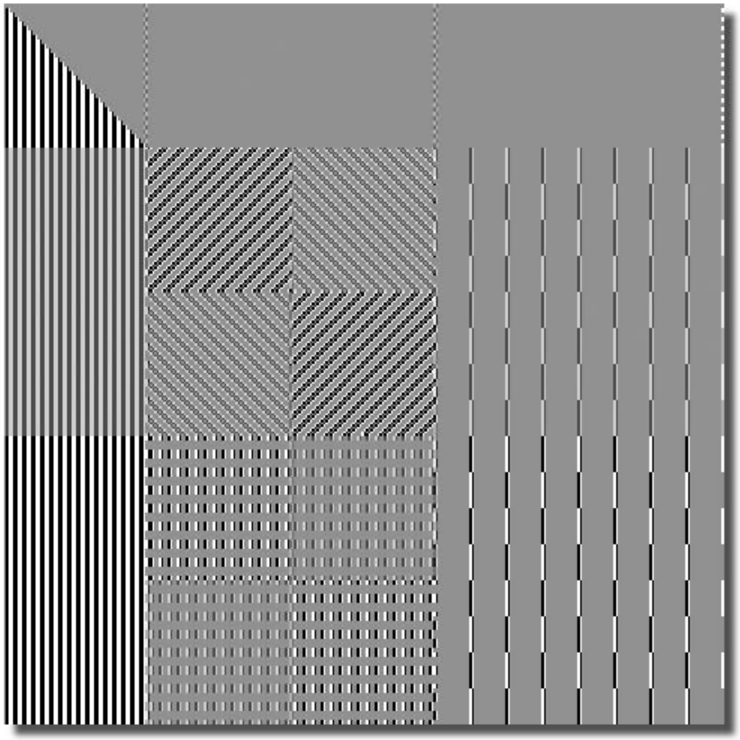
Convolution Examples: Vertical Difference



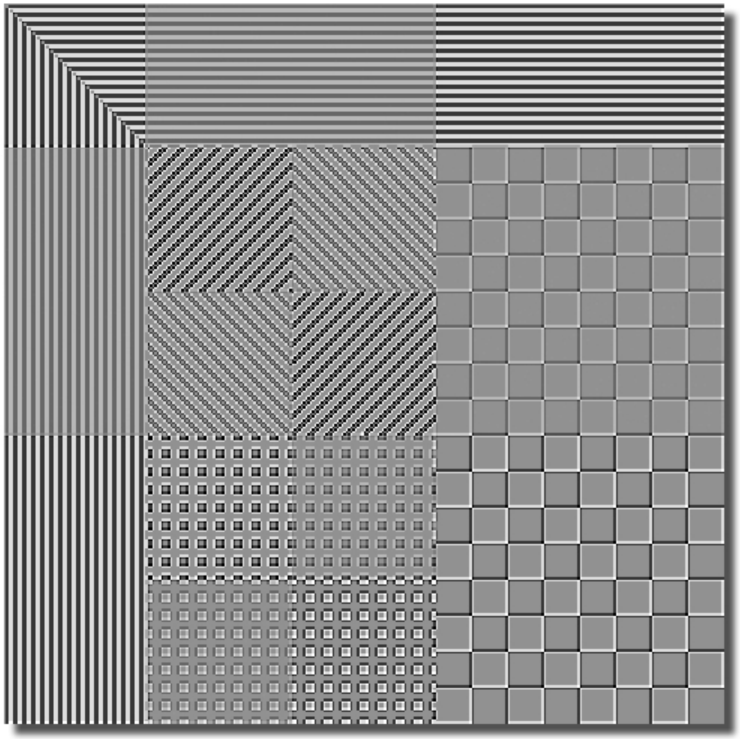


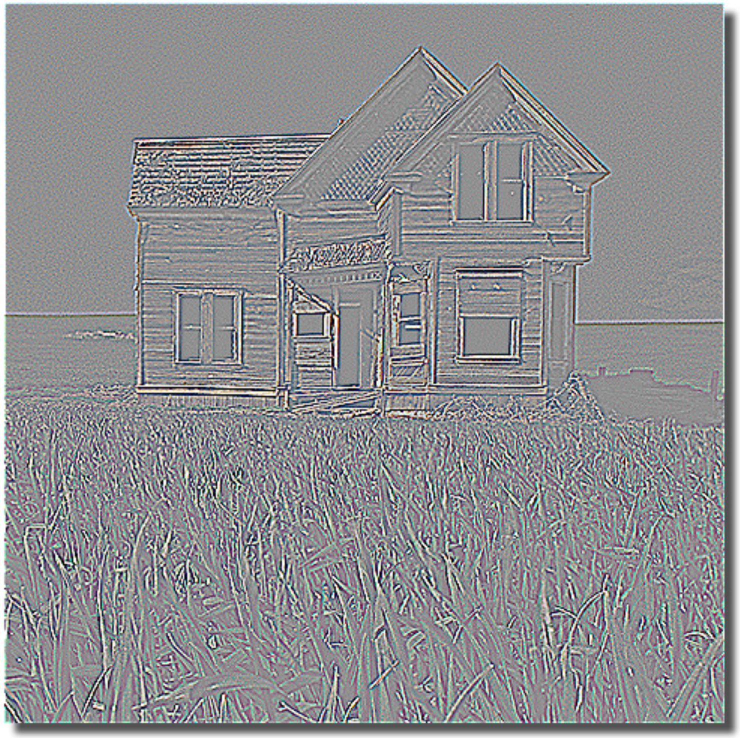
$[-1 \quad 2 \quad -1]$

Convolution Examples: Horizontal Difference

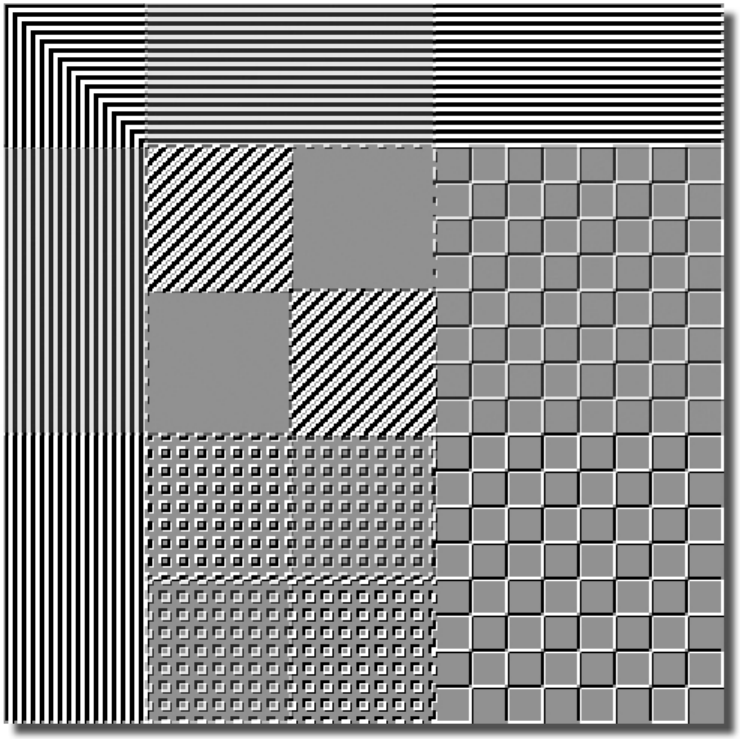


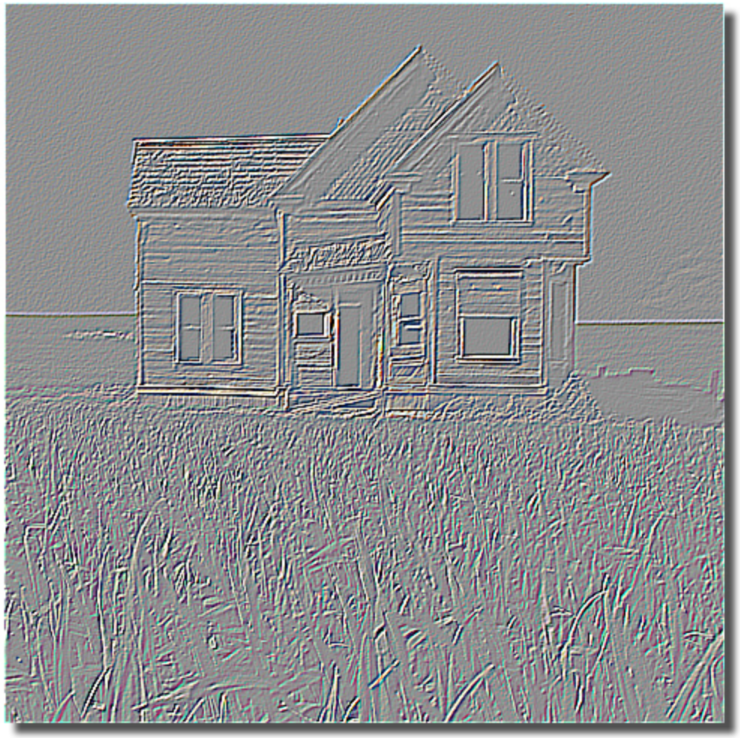
Convolution Examples: H + V Diff.



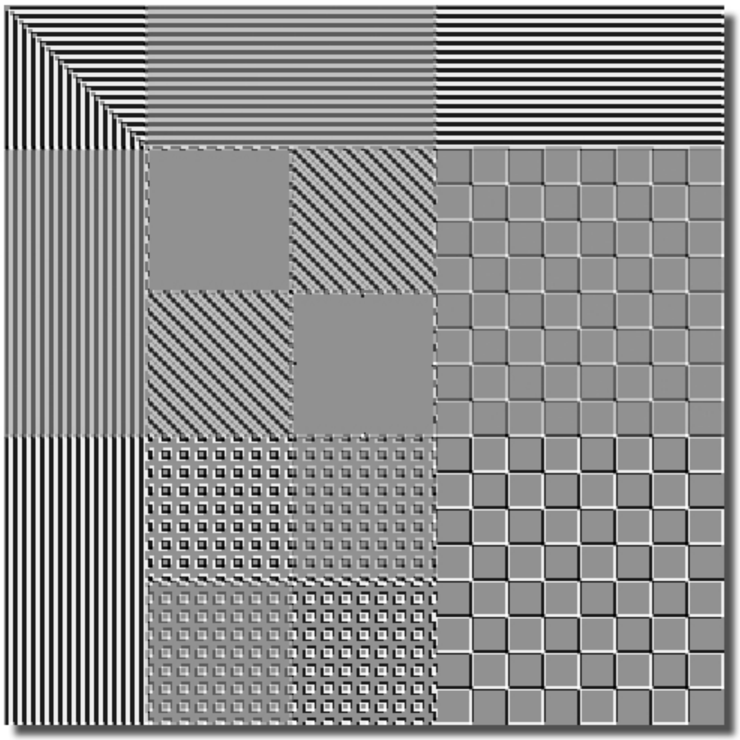


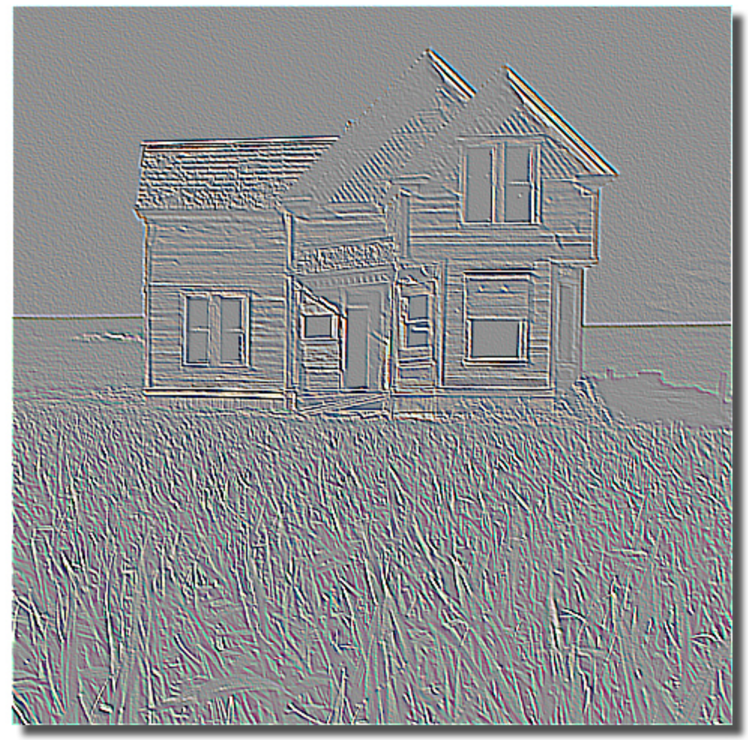
Convolution Examples: Diagonal Difference



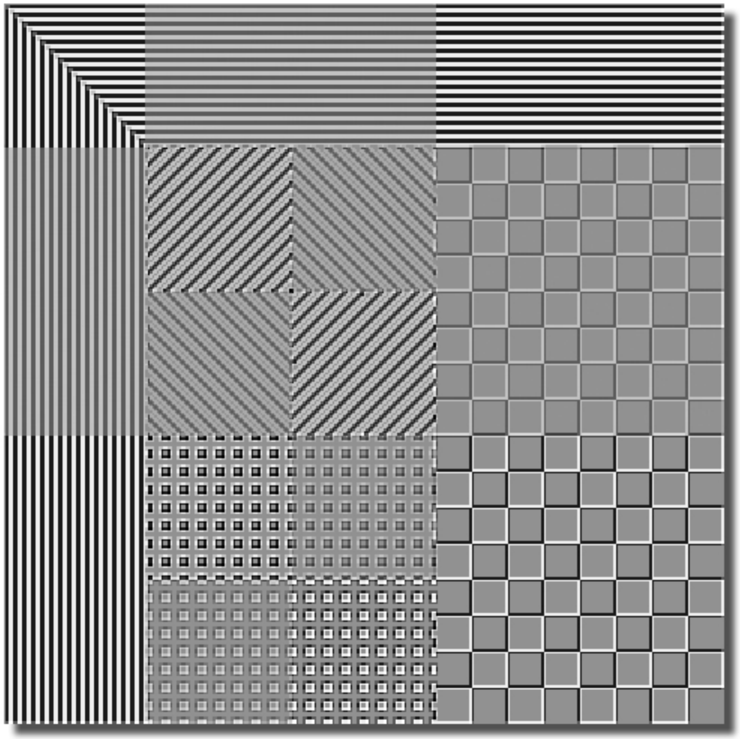


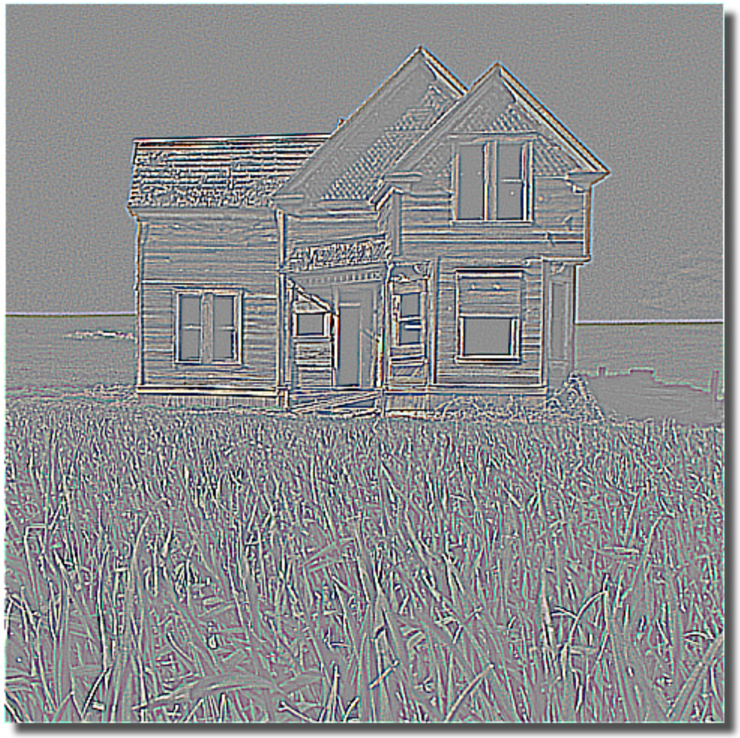
Convolution Examples: Diagonal Difference



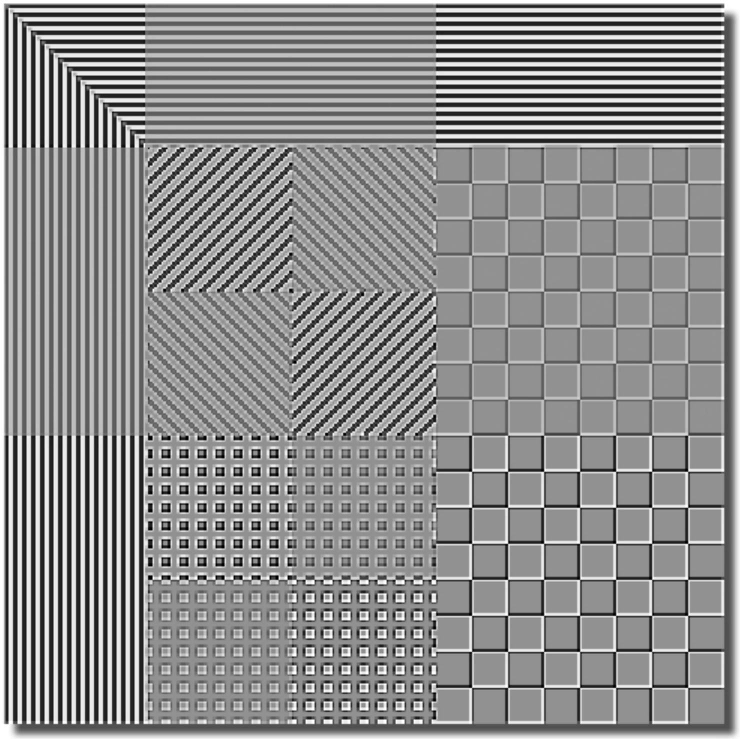


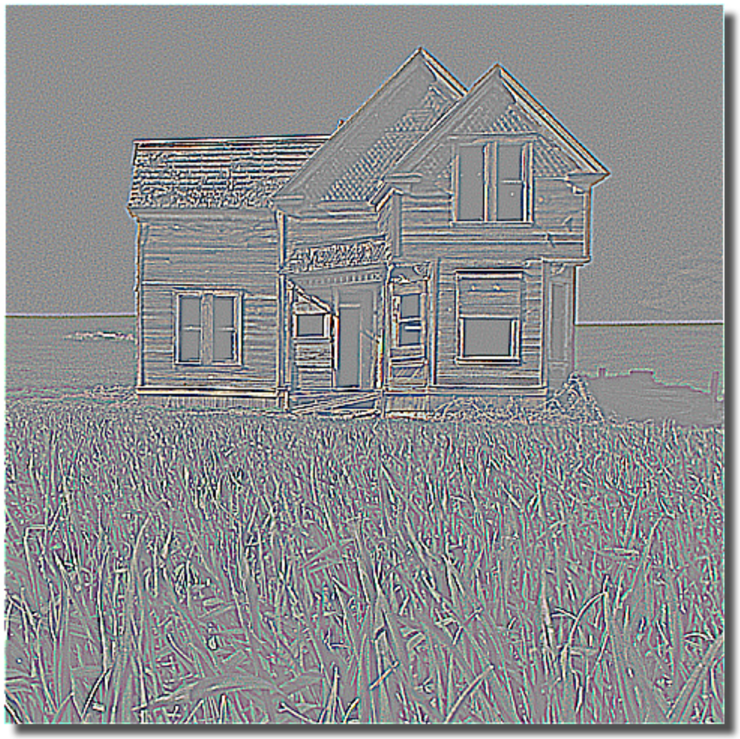
Convolution Examples: D + D Difference



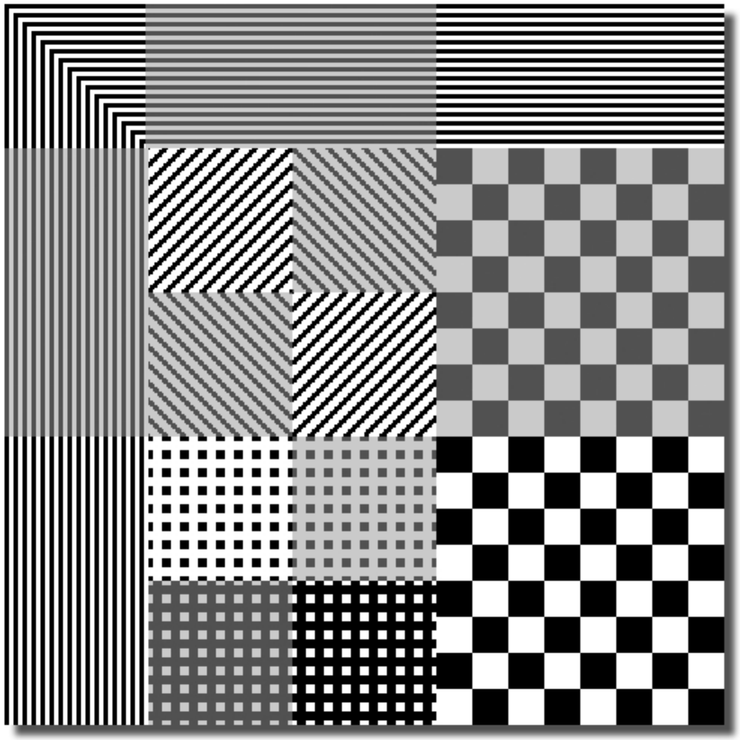


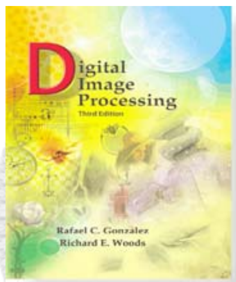
Convolution Examples: H + V + D Diff.





Convolution Examples: Original Images





Laplacian

$$\nabla^2 f = \frac{d^2 f}{dx^2} + \frac{d^2 f}{dy^2}$$

$$-\nabla^2 f = -\frac{d^2 f}{dx^2} - \frac{d^2 f}{dy^2}$$

H+V

0	1	0
1	−4	1
0	1	0

H+V+D

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.

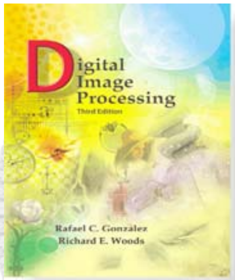
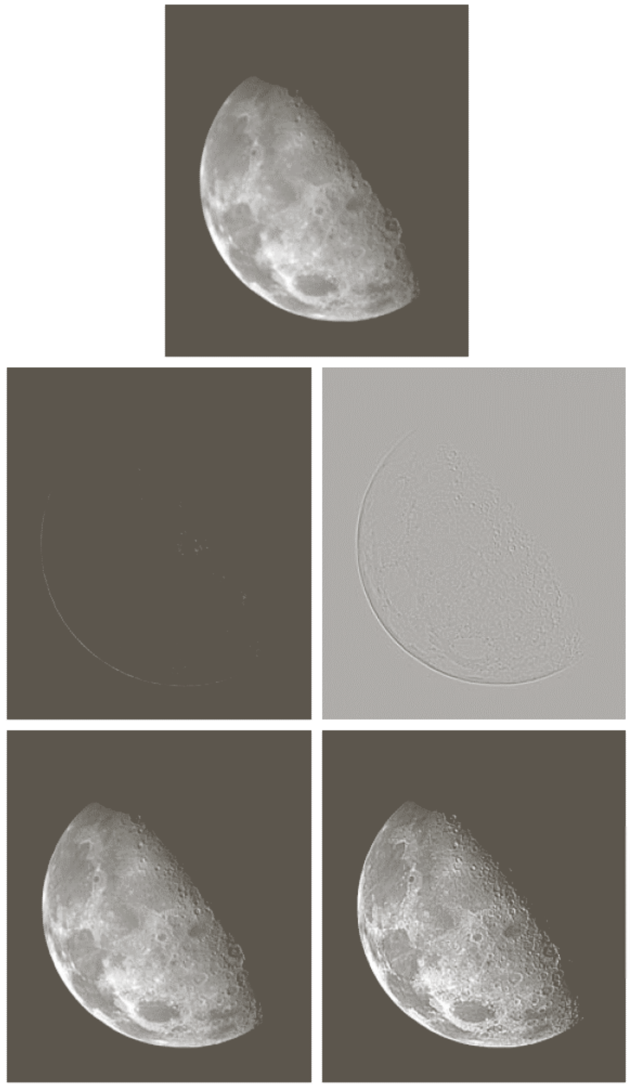


Image Sharpening



a
b c
d e

FIGURE 3.38 (a) Blurred image of the North Pole of the moon. (b) Laplacian without scaling. (c) Laplacian with scaling. (d) Image sharpened using the mask in Fig. 3.37(a). (e) Result of using the mask in Fig. 3.37(b). (Original image courtesy of NASA.)

$\nabla^2 f$

$\nabla^2 f + \min(\nabla^2 f)$

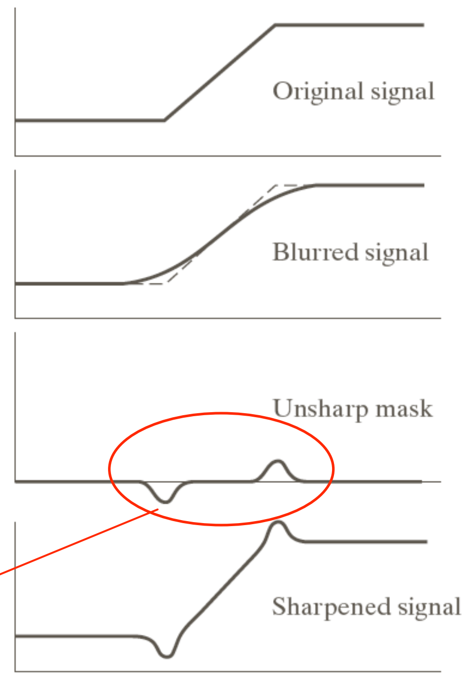
Sharpening using H+V Laplacian

Sharpening using H+V+D Laplacian

Image Sharpening

Sharpening
1. Blur original image
2. Subtract blurred image from original (this is called the mask)
3. Add mask to original image

The mask looks like a 2nd derivative



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).

Image Sharpening

$$f(x,y)$$

$$\bar{f}(x,y)$$

$$g_{mask}(x,y) = f(x,y) - \bar{f}(x,y)$$

$$g(x,y) = f(x,y) + k \times g_{mask}(x,y)$$

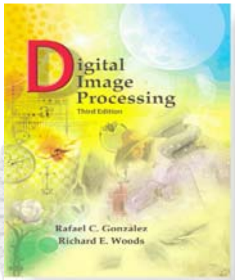
$$g(x,y) = f(x,y) + k \times g_{mask}(x,y)$$

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.

Unsharp masking,k=1

Highboost filtering,k=4.5



Named Derivative (Gradient) Filters

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

Roberts diagonal derivative operators

−1	0
0	1

0	−1
1	0

Sobel center-weighted derivative operators

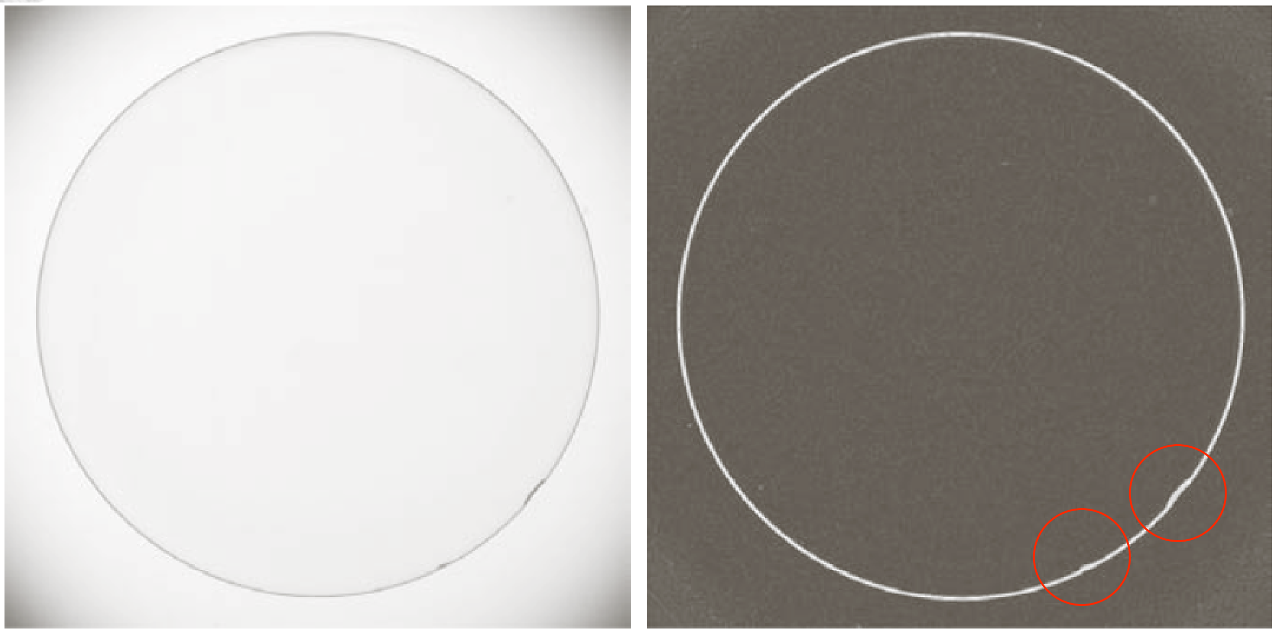
−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

a
b c
d e

FIGURE 3.41 A 3 × 3 region of an image (the zs are intensity values). (b)–(c) Roberts cross gradient operators. (d)–(e) Sobel operators. All the mask coefficients sum to zero, as expected of a derivative operator.

Using the Sobel Filter



a b

FIGURE 3.42 (a) Optical image of contact lens (note defects on the boundary at 4 and 5 o'clock). (b) Sobel gradient. (Original image courtesy of Pete Sites, Perceptics Corporation.)

Sobel S_x+S_y operators

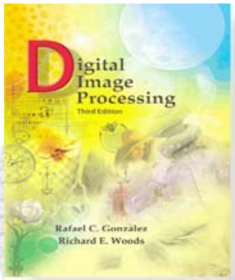
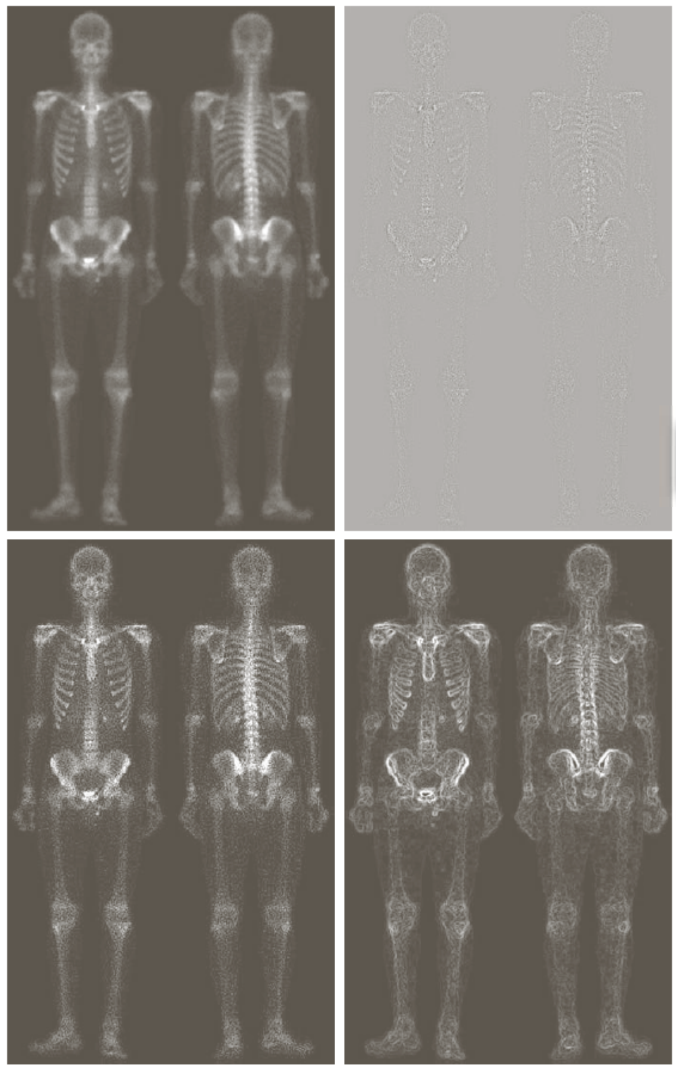


Image Processing Example

original image

Laplacian of image



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).

(c) Sharpened by adding Laplacian

Sobel gradient image

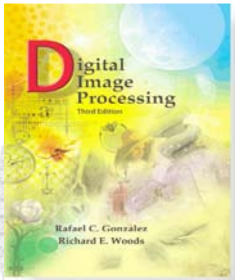
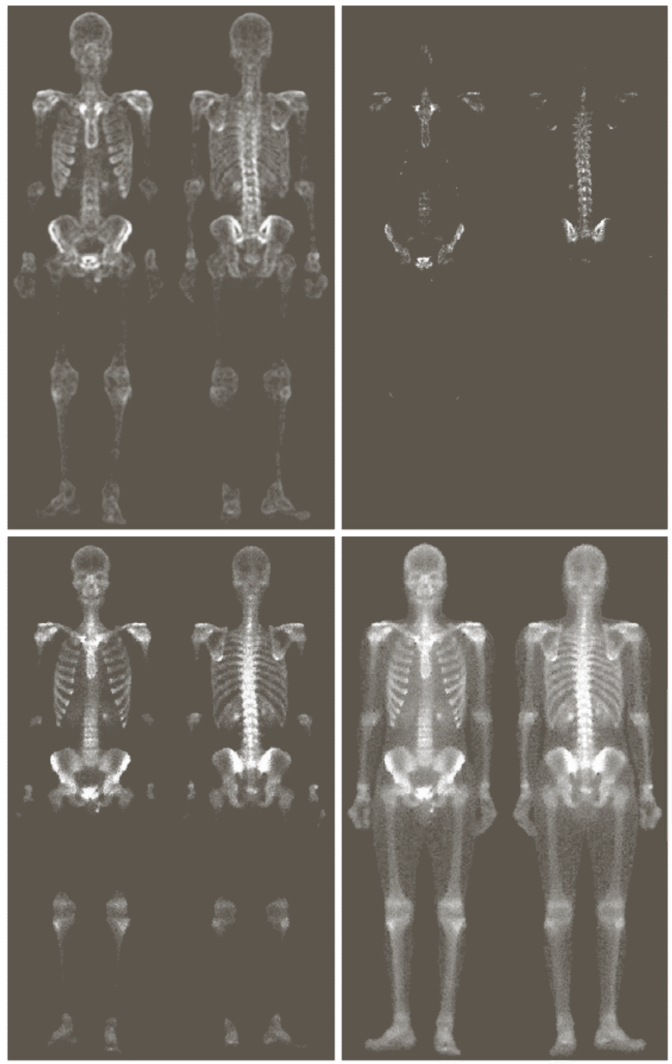


Image Processing Example

(e) Blurred Sobel gradient image

Add Mask to original



e f
g h

FIGURE 3.43 *(Continued)* (e) Sobel image smoothed with a 5 ... fi ... 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.)

Mask = (c) x (e)

Power Law Intensity Transform