Spatial Filtering

Dr. Navjot Singh

Image and Video Processing

Acknowledgements

- Gonzalez, Rafael C. Digital image processing. Pearson, 4th edition, 2018.
- Jain, Anil K. Fundamentals of digital image processing. Prentice-Hall, Inc., 1989.
- Digital Image Processing course by Brian Mac Namee, Dublin Institute of Technology
- Digital Image Processing course by Christophoros Nikou, University of Ioannina

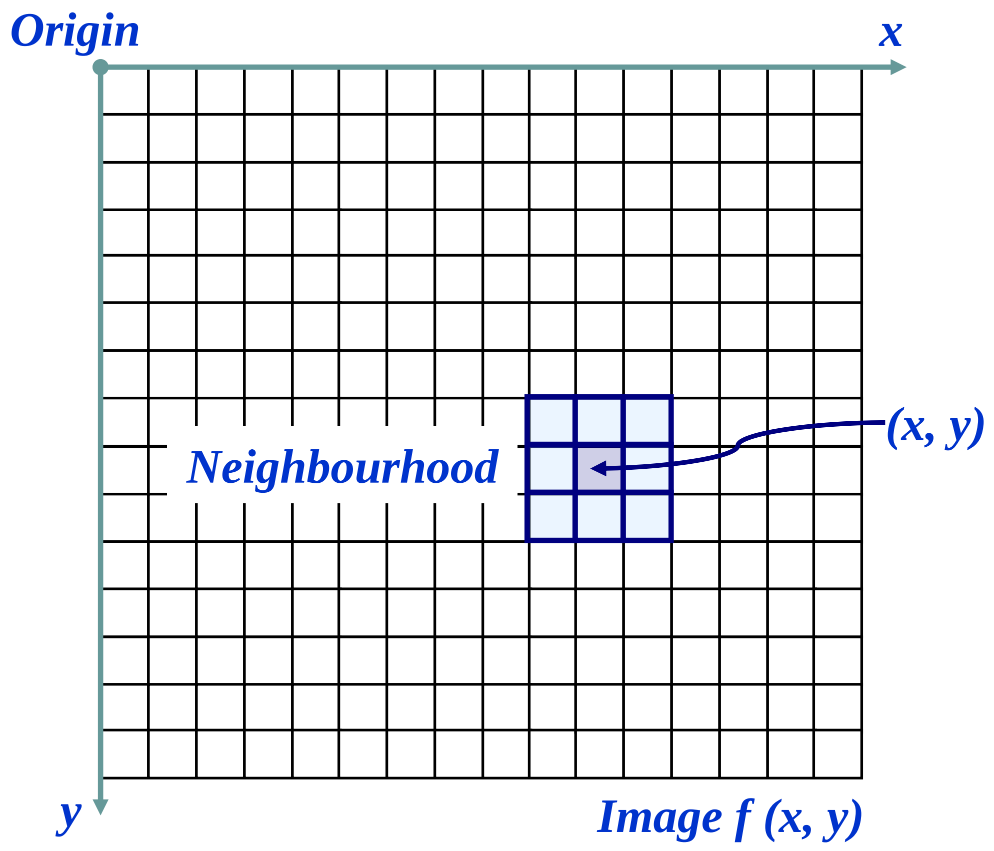
Contents

In this lecture we will look at spatial filtering techniques:

- Neighbourhood operations
- What is spatial filtering?
- Smoothing operations
- What happens at the edges?
- Correlation and convolution
- Sharpening filters
- Combining filtering techniques

Neighbourhood Operations

- Neighbourhood operations simply operate on a larger neighbourhood of pixels than point operations
- Neighbourhoods are mostly a rectangle around a central pixel
- Any size rectangle and any shape filter are possible

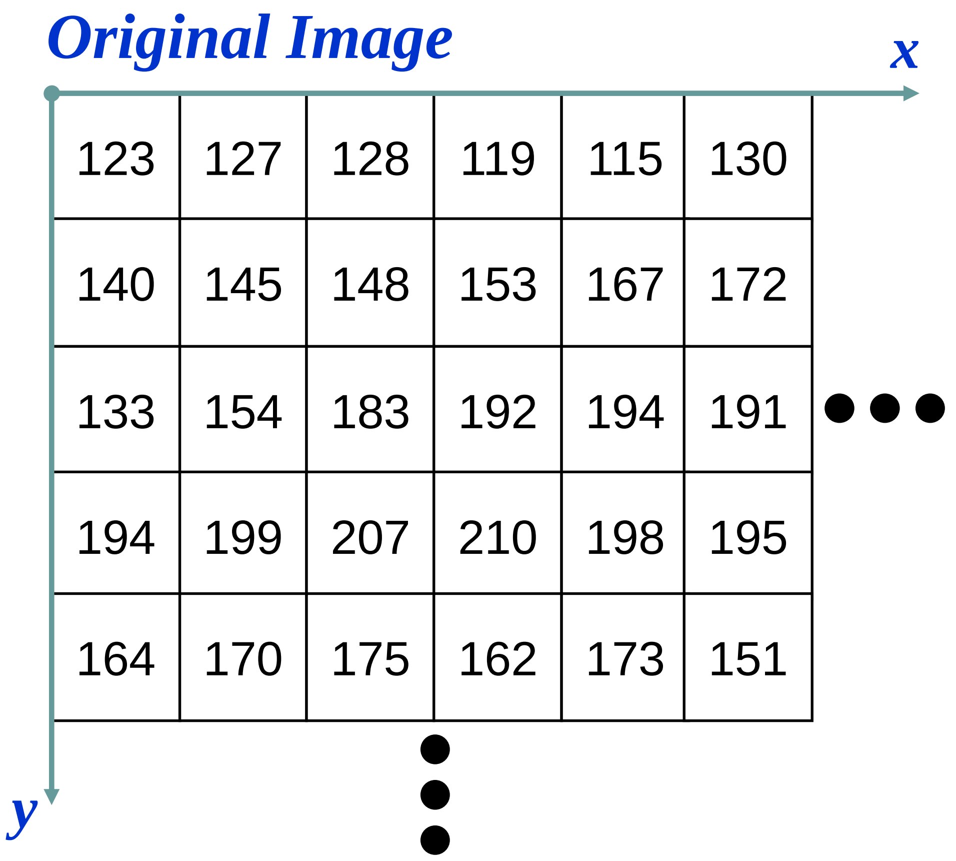


Simple Neighbourhood Operations

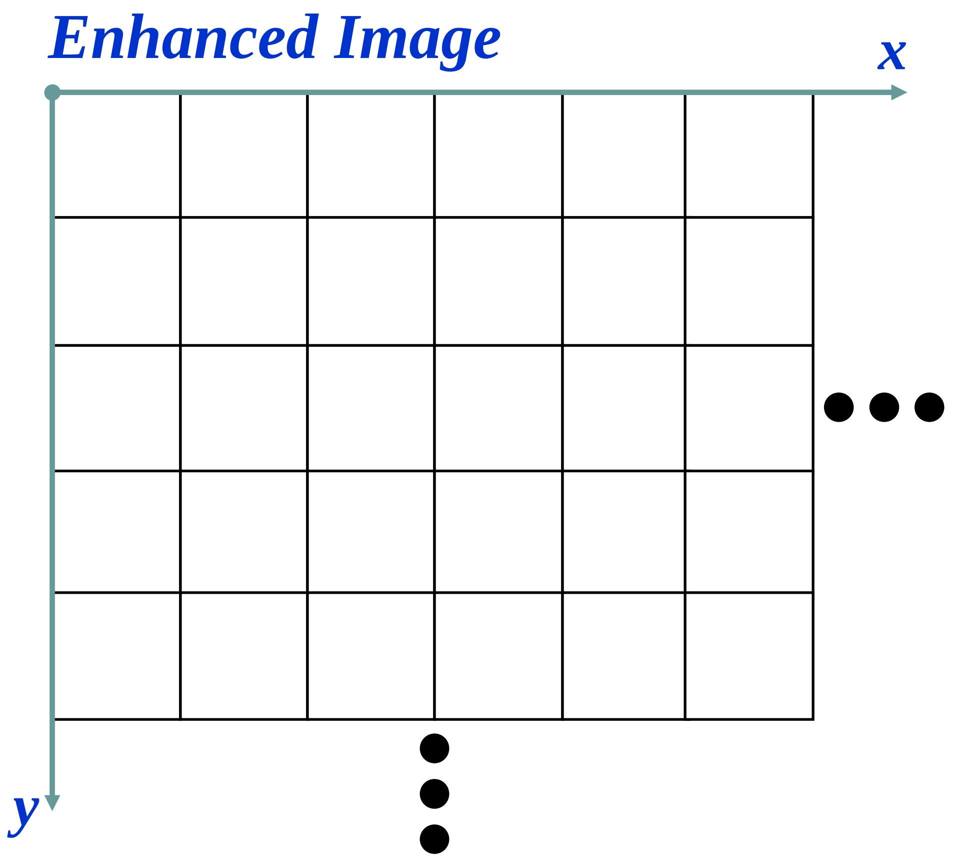
Some simple neighbourhood operations include:

- **Min:** Set the pixel value to the minimum in the neighbourhood
- **Max:** Set the pixel value to the maximum in the neighbourhood
- **Median:** The median value of a set of numbers is the midpoint value in that set (e.g. from the set [1, 7, 15, 18, 24] 15 is the median). Sometimes the median works better than the average

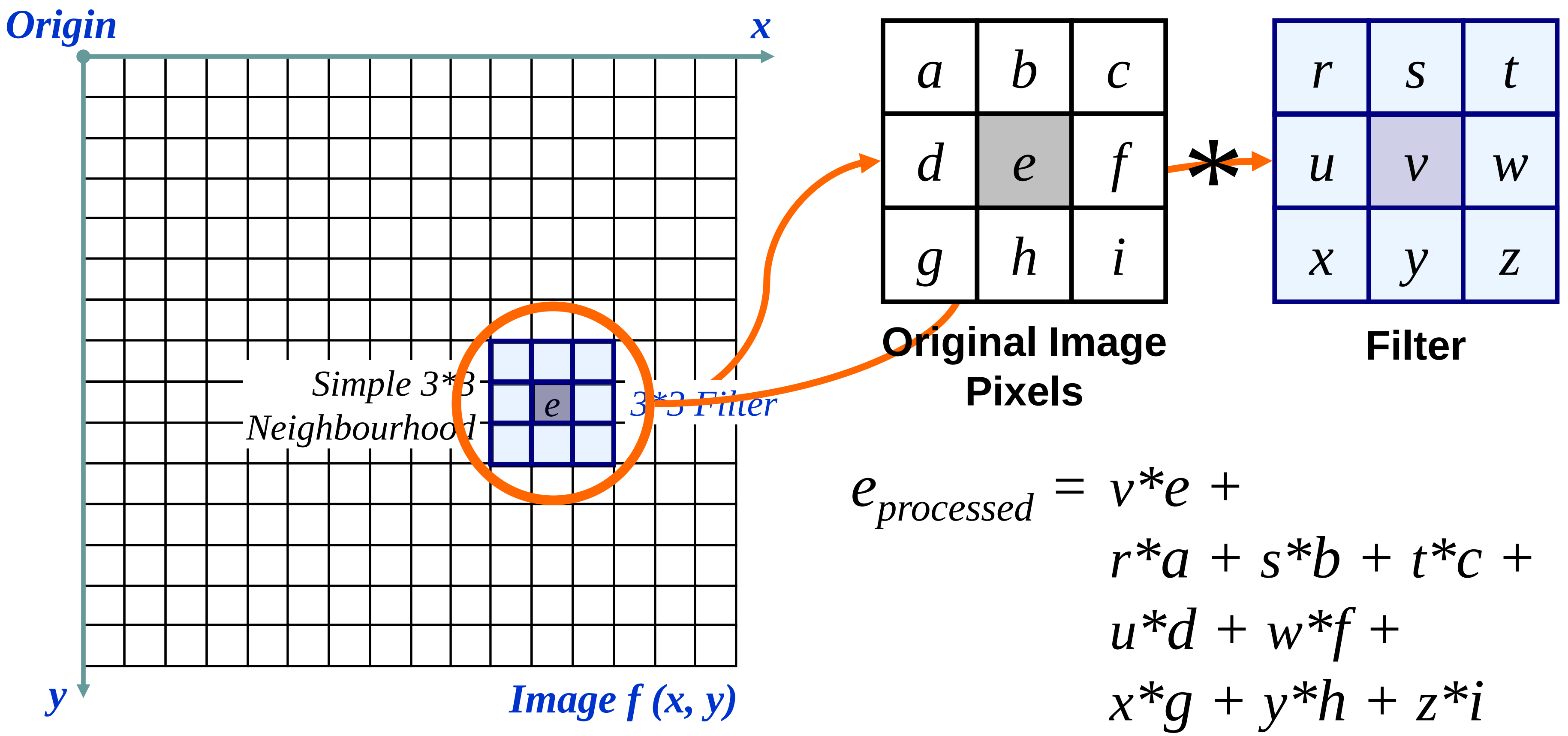
Simple Neighbourhood Operations Example


Original Image
x
y

123	127	128	119	115	130
140	145	148	153	167	172
133	154	183	192	194	191
194	199	207	210	198	195
164	170	175	162	173	151

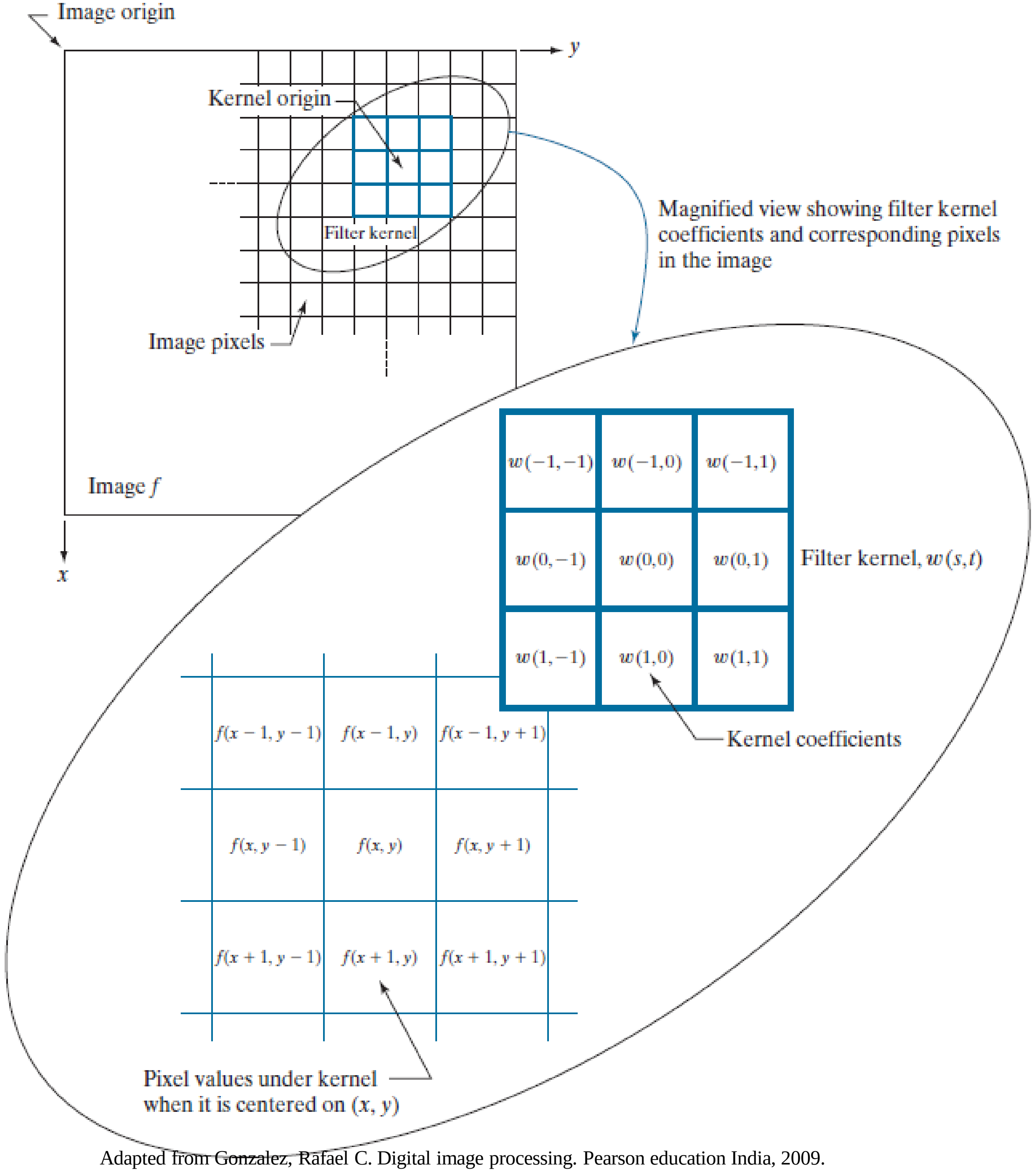


The Spatial Filtering Process



$$e_{processed} = v*e + r*a + s*b + t*c + u*d + w*f + x*g + y*h + z*i$$

The above is repeated for every pixel in the original image to generate the filtered image

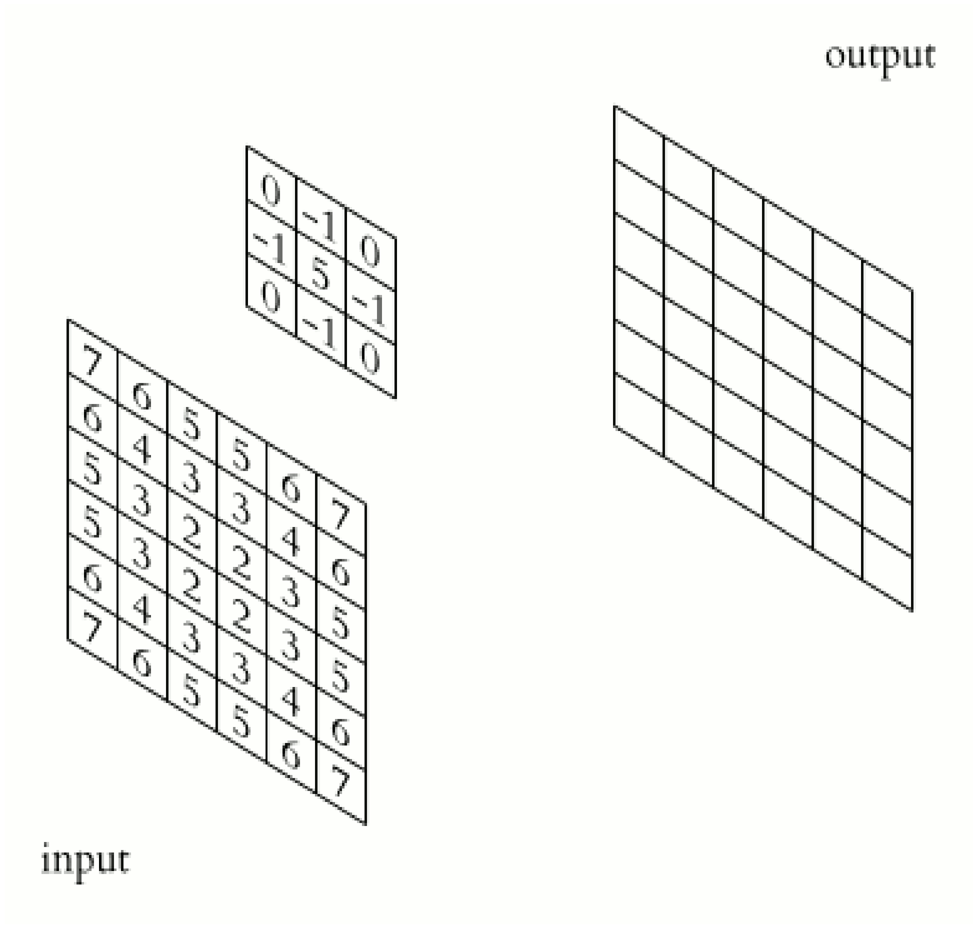


$$g(x,y)=\sum_{s=-a}^{a}\sum_{t=-b}^{b}w(s,t)\,f(x+s,y+t)$$

Filtering can be given in equation form as shown above

Notations are based on the image shown to the left

Filtering process



Source: Google Images

Spatial Correlation and Convolution

Spatial Correlation

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

Spatial Convolution

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

For symmetric filters it makes no difference.

Spatial Convolution and Correlation

- Padding

$$S_v \times S_h$$
$$S_v = m + M - 1$$
$$S_h = n + N - 1$$

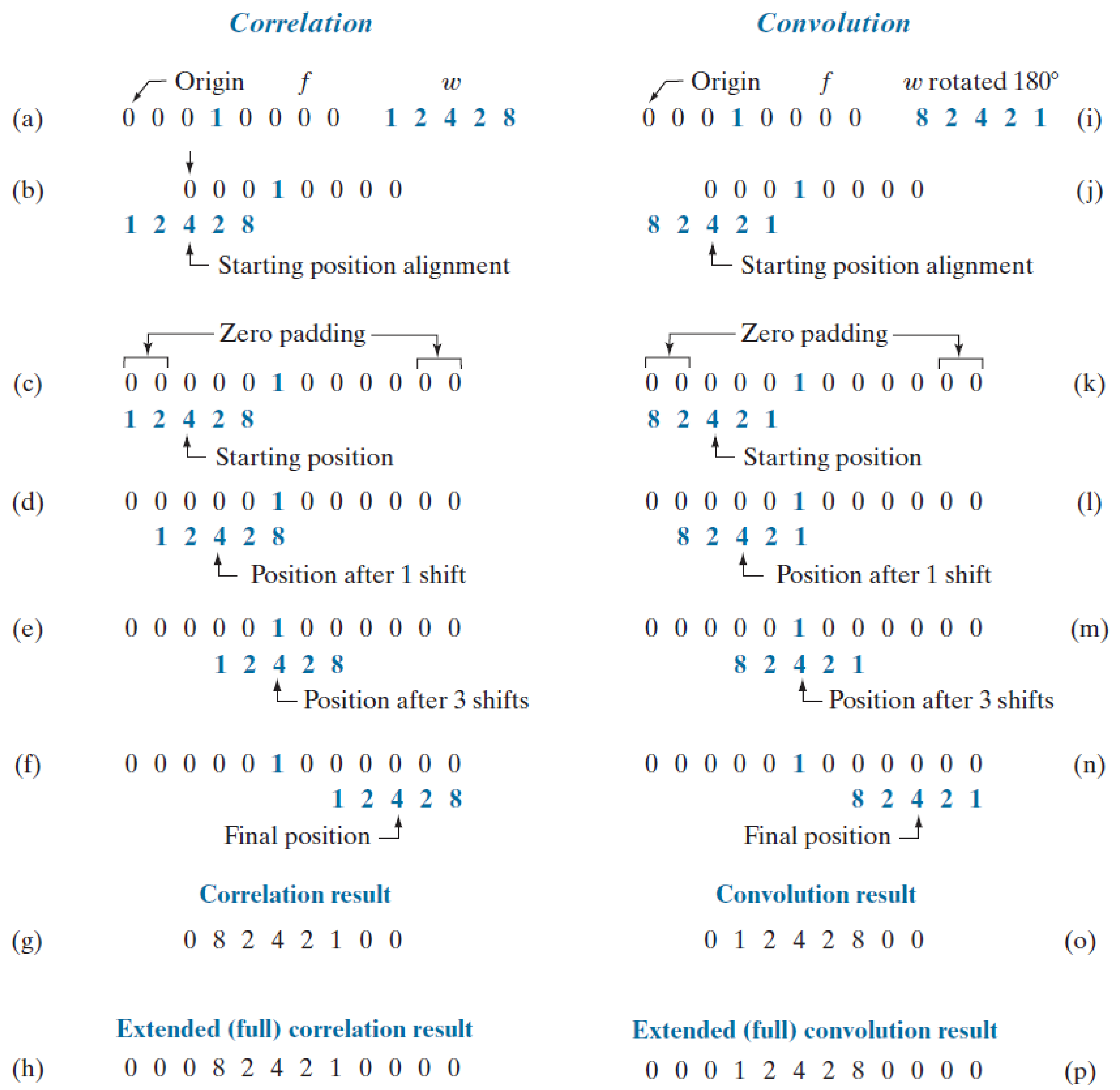
  - Zero
  - Replicate
  - Mirror

Spatial Convolution and Correlation

f								
	0	**0**	**0**	**1**	**0**	**0**	**0**	**0**

w					
	1	**2**	**4**	**2**	**8**

Find result of convolution and correlation.


Correlation
Convolution
Origin
f
w
w rotated 180°
(a) 0 0 0 1 0 0 0 0 1 2 4 2 8
(i) 0 0 0 1 0 0 0 0 8 2 4 2 1
(b) 0 0 0 1 0 0 0 0 / 1 2 4 2 8
(j) 0 0 0 1 0 0 0 0 / 8 2 4 2 1
Starting position alignment
Zero padding
(c) 0 0 0 0 0 1 0 0 0 0 0 0 / 1 2 4 2 8
(k) 0 0 0 0 0 1 0 0 0 0 0 0 / 8 2 4 2 1
Starting position
(d) 0 0 0 0 0 1 0 0 0 0 0 0 / 1 2 4 2 8
(l) 0 0 0 0 0 1 0 0 0 0 0 0 / 8 2 4 2 1
Position after 1 shift
(e) 0 0 0 0 0 1 0 0 0 0 0 0 / 1 2 4 2 8
(m) 0 0 0 0 0 1 0 0 0 0 0 0 / 8 2 4 2 1
Position after 3 shifts
(f) 0 0 0 0 0 1 0 0 0 0 0 0 / 1 2 4 2 8
(n) 0 0 0 0 0 1 0 0 0 0 0 0 / 8 2 4 2 1
Final position
Correlation result
(g) 0 8 2 4 2 1 0 0
Convolution result
(o) 0 1 2 4 2 8 0 0
Extended (full) correlation result
(h) 0 0 0 8 2 4 2 1 0 0 0 0
Extended (full) convolution result
(p) 0 0 0 1 2 4 2 8 0 0 0 0

Origin f

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

(a)

Padded f

$$\begin{matrix}0&0&0&0&0&0&0\\0&0&0&0&0&0&0\\0&0&0&0&0&0&0\\0&0&0&1&0&0&0\\0&0&0&0&0&0&0\\0&0&0&0&0&0&0\\0&0&0&0&0&0&0\end{matrix}$$

(b)

Initial position for w

$$\begin{array}{:ccc:cccc}\hdashline 1&2&3&0&0&0&0\\4&5&6&0&0&0&0\\7&8&9&0&0&0&0\\\hdashline 0&0&0&1&0&0&0\\0&0&0&0&0&0&0\\0&0&0&0&0&0&0\\0&0&0&0&0&0&0\end{array}$$

(c)

Correlation result

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

(d)

Full correlation result

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

(e)

Rotated w

$$\begin{array}{:ccc:cccc}\hdashline 9&8&7&0&0&0&0\\6&5&4&0&0&0&0\\3&2&1&0&0&0&0\\\hdashline 0&0&0&1&0&0&0\\0&0&0&0&0&0&0\\0&0&0&0&0&0&0\\0&0&0&0&0&0&0\end{array}$$

(f)

Convolution result

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

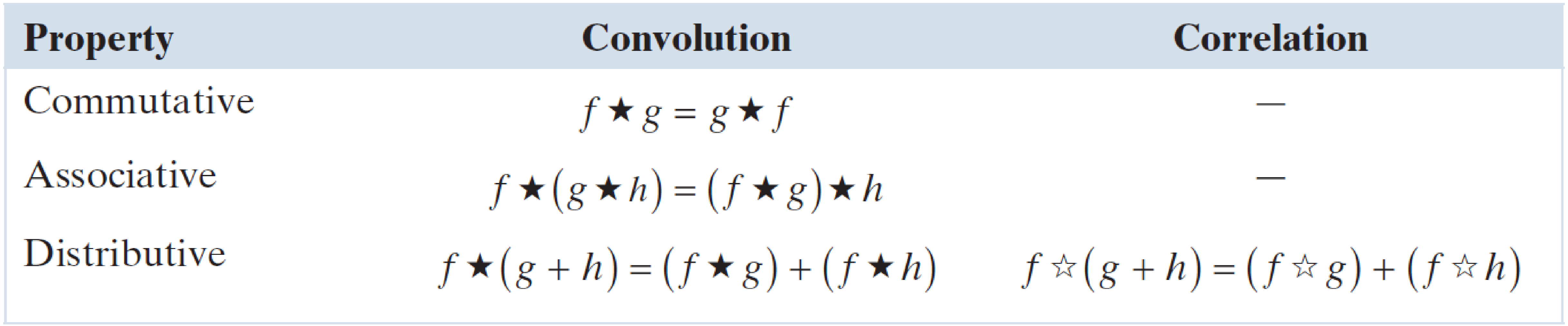
(g)

Full convolution result

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

(h)

Spatial Convolution and Correlation: Properties



Property	Convolution	Correlation
Commutative	$f \star g = g \star f$	—
Associative	$f \star (g \star h) = (f \star g) \star h$	—
Distributive	$f \star (g + h) = (f \star g) + (f \star h)$	$f \mathbin{\text{☆}} (g + h) = (f \mathbin{\text{☆}} g) + (f \mathbin{\text{☆}} h)$

Spatial filters

Smoothing (lowpass) Spatial Filters

Sharpening (high pass) Spatial Filters

Bandreject Filters

Bandpass Filters

Smoothing Spatial Filters

- Smoothing filters are used
  - Blurring
  - Noise reduction

- Blurring is used in removal of small details and bridging of small gaps in lines or curves

- Smoothing spatial filters include linear filters and nonlinear filters.

Smoothing Spatial Filters

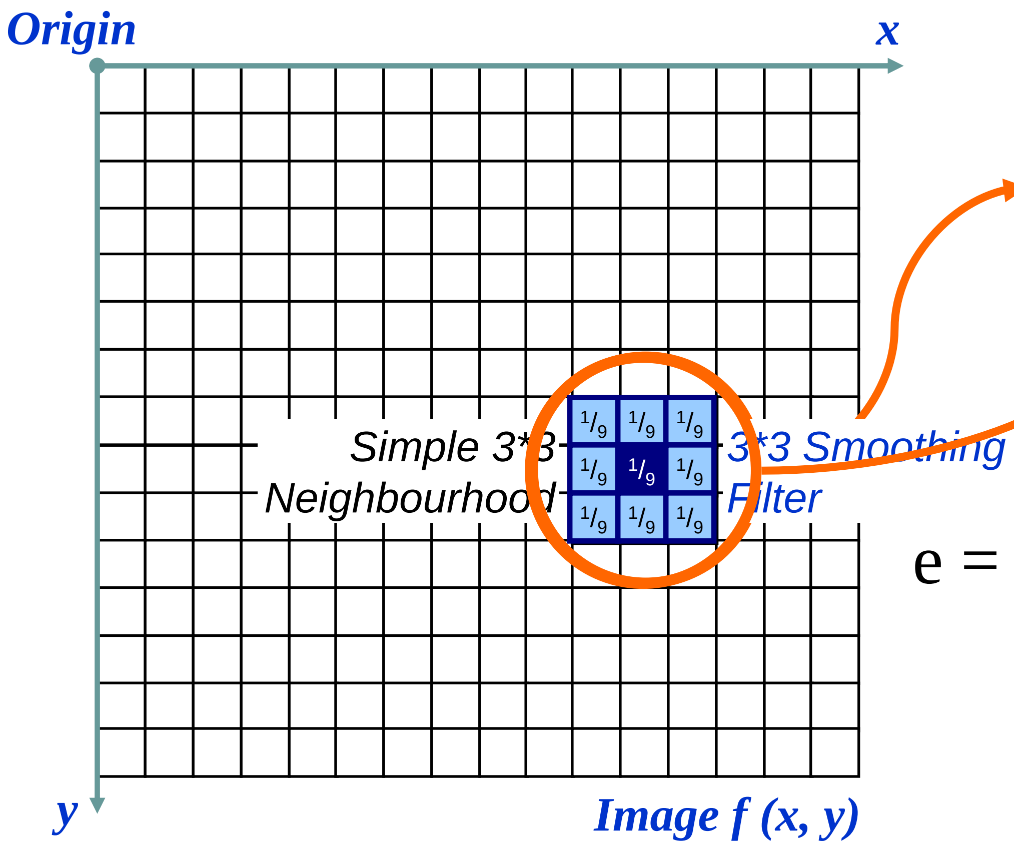
One of the simplest spatial filtering operations we can perform is a smoothing operation

- Simply average all of the pixels in a neighbourhood around a central value
- Especially useful in removing noise from images
- Also useful for highlighting gross detail

$^1/_9$	$^1/_9$	$^1/_9$
$^1/_9$	$^1/_9$	$^1/_9$
$^1/_9$	$^1/_9$	$^1/_9$

Simple Averaging Filter or Box Filter

Smoothing Spatial Filtering



104	*100*	*108*
99	*106*	*98*
95	*90*	*85*

Original Image Pixels

$*$

$^1/_9$	$^1/_9$	$^1/_9$
$^1/_9$	$^1/_9$	$^1/_9$
$^1/_9$	$^1/_9$	$^1/_9$

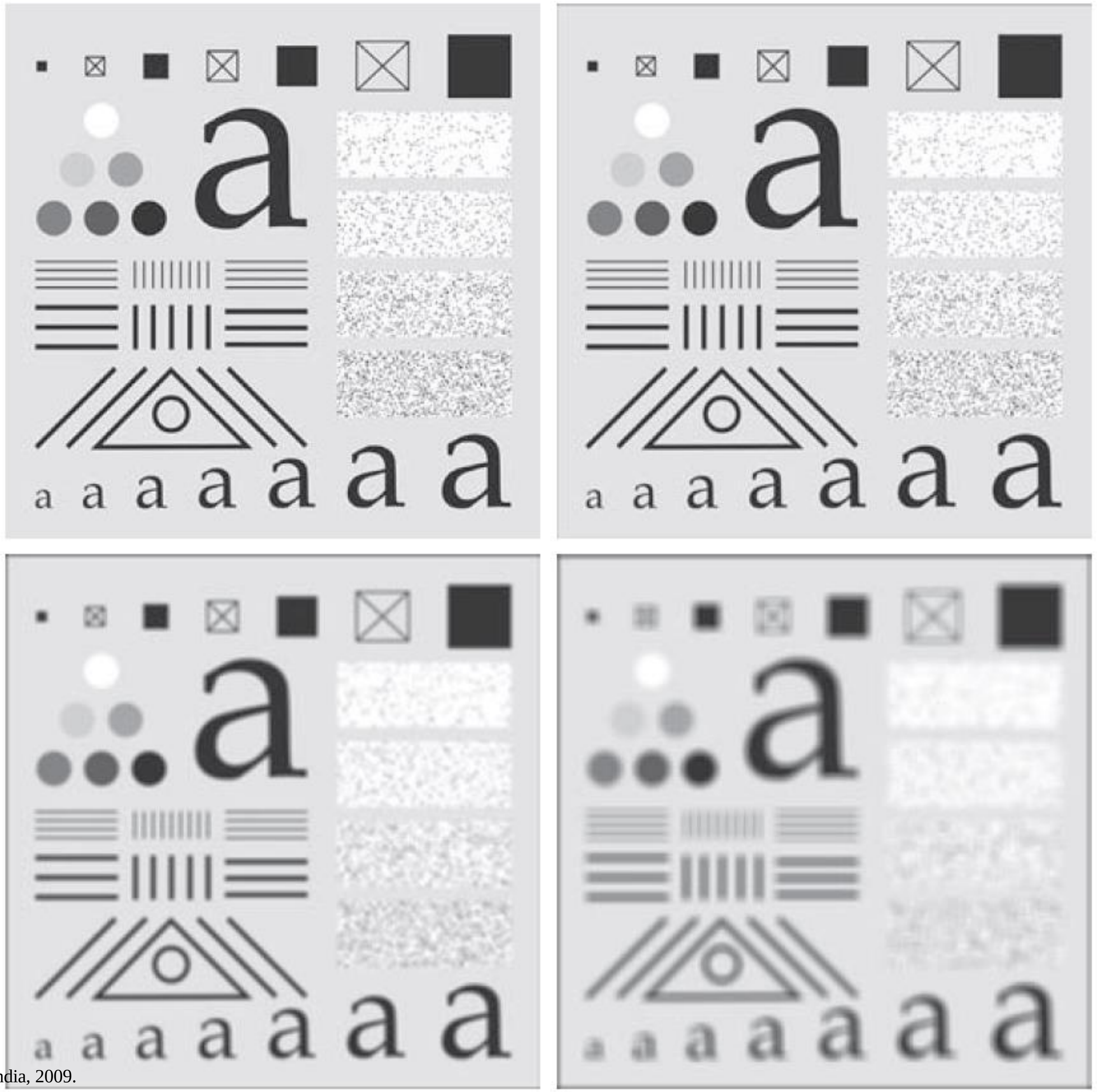
Filter

$$\begin{aligned} e = {} & {}^1/_9*106 + \\ & {}^1/_9*104 + {}^1/_9*100 + {}^1/_9*108 + \\ & {}^1/_9*99 + {}^1/_9*98 + \\ & {}^1/_9*95 + {}^1/_9*90 + {}^1/_9*85 \\ = {} & 98.3333 \end{aligned}$$

The above is repeated for every pixel in the original image to generate the smoothed image.

a	b
c	d

FIGURE 3.33
(a) Test pattern of size 1024×1024 pixels.
(b)-(d) Results of lowpass filtering with box kernels of sizes 3×3, 11×11, and 21×21, respectively.



Weighted Smoothing Filters

More effective smoothing filters can be generated by allowing different pixels in the neighbourhood different weights in the averaging function

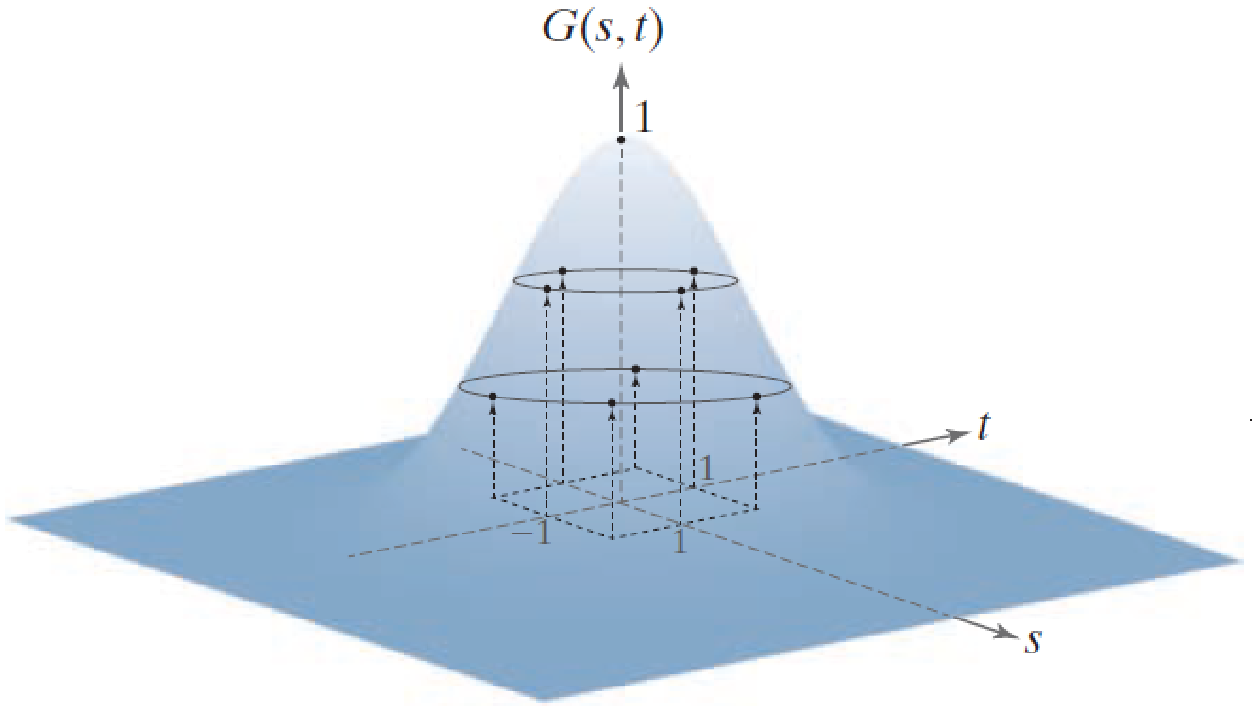
- Pixels closer to the central pixel are more important
- Often referred to as a *weighted averaging*

$^1/_{16}$	$^2/_{16}$	$^1/_{16}$
$^2/_{16}$	$^4/_{16}$	$^2/_{16}$
$^1/_{16}$	$^2/_{16}$	$^1/_{16}$

Weighted averaging filter

Low-pass Gaussian filter kernels

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



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

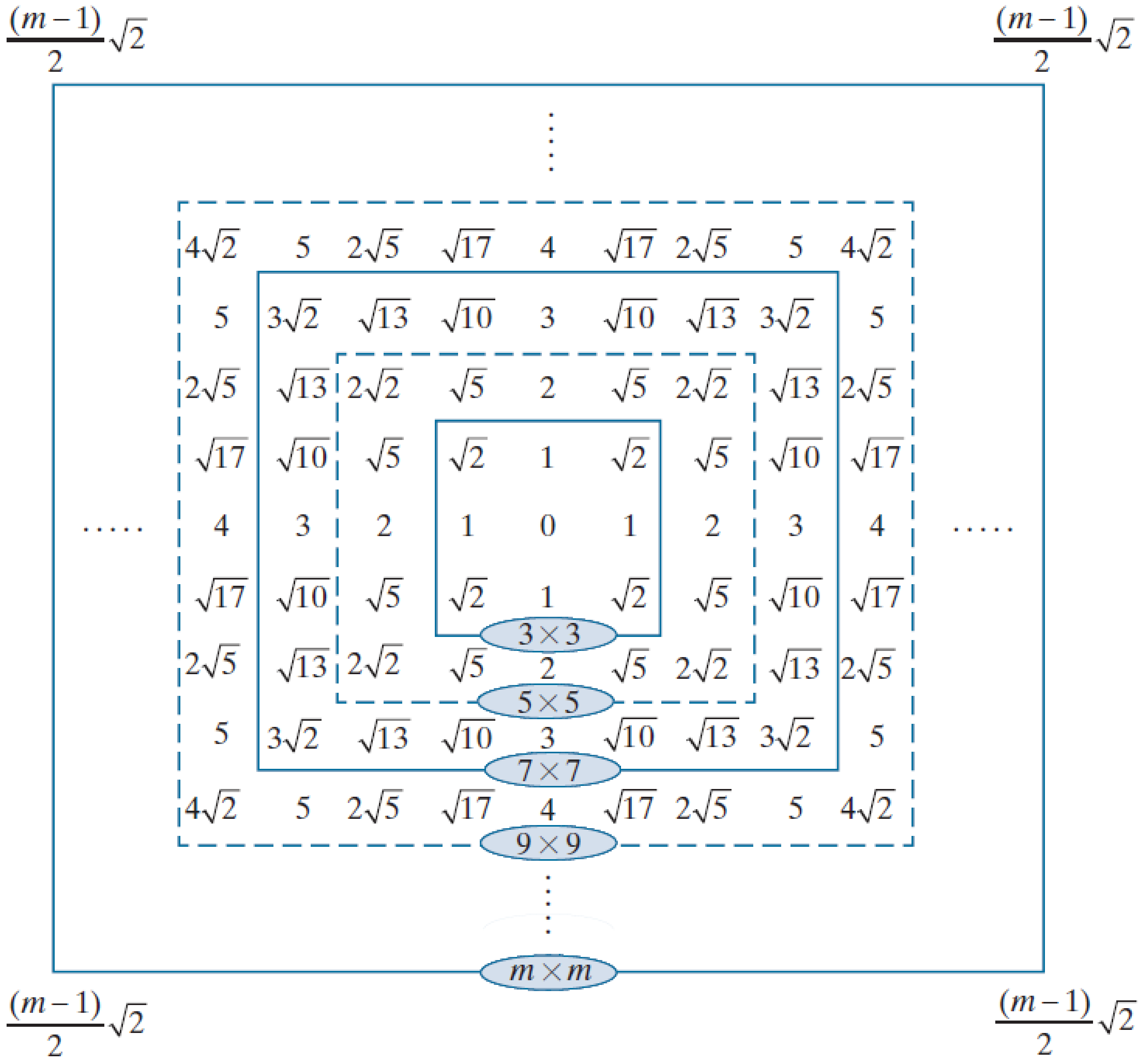
0.3679	0.6065	0.3679
0.6065	1.0000	0.6065
0.3679	0.6065	0.3679

a b

FIGURE 3.35 (a) Sampling a Gaussian function to obtain a discrete Gaussian kernel. The values shown are for $K = 1$ and $\sigma = 1$. (b) Resulting 3×3 kernel [this is the same as Fig. 3.31(b)].

Adapted from Gonzalez, Rafael C. Digital image processing. Pearson education India, 2009.

FIGURE 3.34
Distances from the center for various sizes of square kernels.



$\frac{(m-1)}{2}\sqrt{2}$
$\frac{(m-1)}{2}\sqrt{2}$
$4\sqrt{2}$ 5 $2\sqrt{5}$ $\sqrt{17}$ 4 $\sqrt{17}$ $2\sqrt{5}$ 5 $4\sqrt{2}$
5 $3\sqrt{2}$ $\sqrt{13}$ $\sqrt{10}$ 3 $\sqrt{10}$ $\sqrt{13}$ $3\sqrt{2}$ 5
$2\sqrt{5}$ $\sqrt{13}$ $2\sqrt{2}$ $\sqrt{5}$ 2 $\sqrt{5}$ $2\sqrt{2}$ $\sqrt{13}$ $2\sqrt{5}$
$\sqrt{17}$ $\sqrt{10}$ $\sqrt{5}$ $\sqrt{2}$ 1 $\sqrt{2}$ $\sqrt{5}$ $\sqrt{10}$ $\sqrt{17}$
4 3 2 1 0 1 2 3 4
$\sqrt{17}$ $\sqrt{10}$ $\sqrt{5}$ $\sqrt{2}$ 1 $\sqrt{2}$ $\sqrt{5}$ $\sqrt{10}$ $\sqrt{17}$
3×3
$2\sqrt{5}$ $\sqrt{13}$ $2\sqrt{2}$ $\sqrt{5}$ 2 $\sqrt{5}$ $2\sqrt{2}$ $\sqrt{13}$ $2\sqrt{5}$
5×5
5 $3\sqrt{2}$ $\sqrt{13}$ $\sqrt{10}$ 3 $\sqrt{10}$ $\sqrt{13}$ $3\sqrt{2}$ 5
7×7
$4\sqrt{2}$ 5 $2\sqrt{5}$ $\sqrt{17}$ 4 $\sqrt{17}$ $2\sqrt{5}$ 5 $4\sqrt{2}$
9×9
$m\times m$
$\frac{(m-1)}{2}\sqrt{2}$
$\frac{(m-1)}{2}\sqrt{2}$

Adapted from Gonzalez, Rafael C. Digital image processing. Pearson education India, 2009.

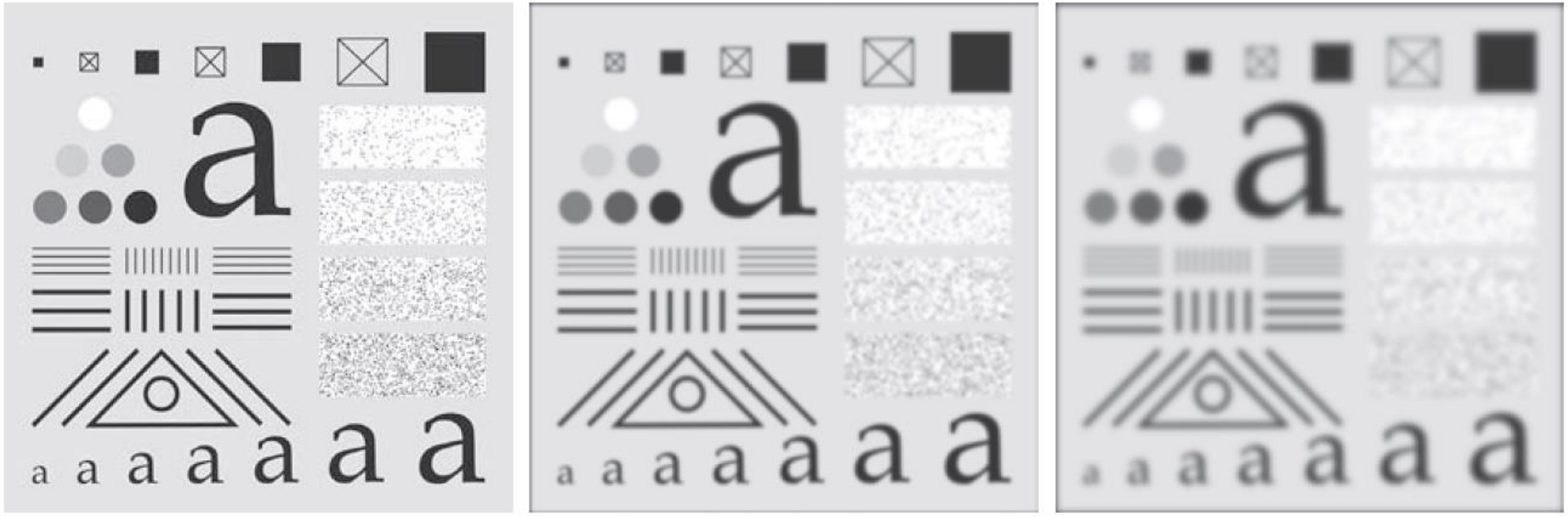
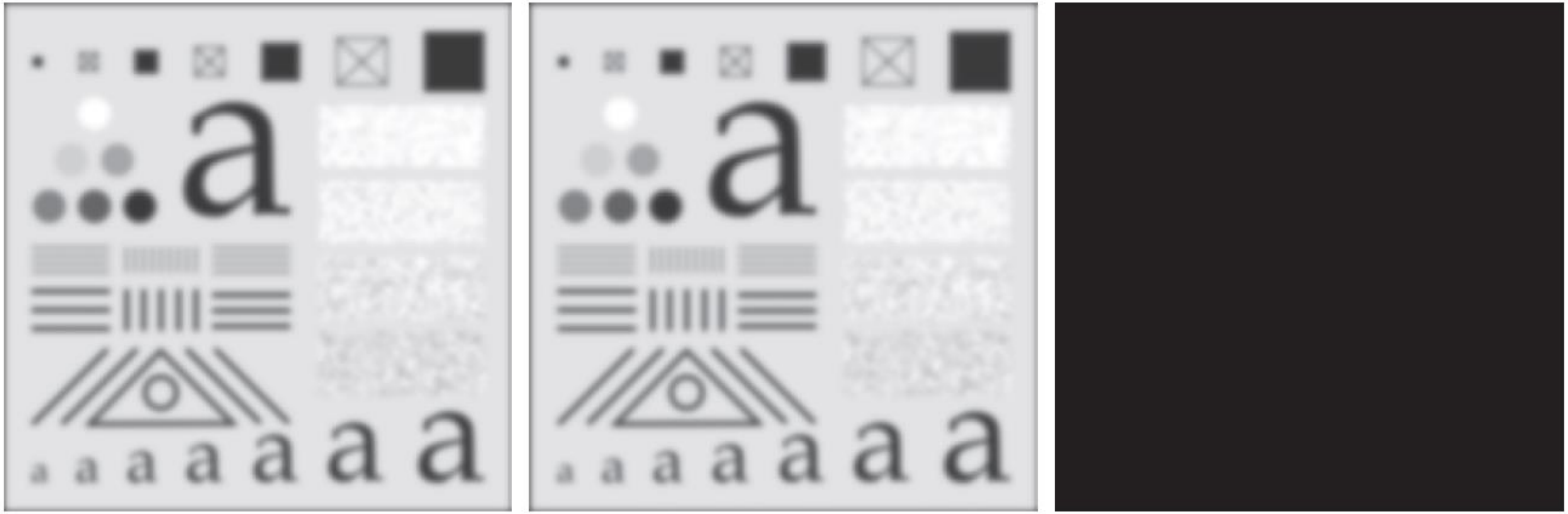
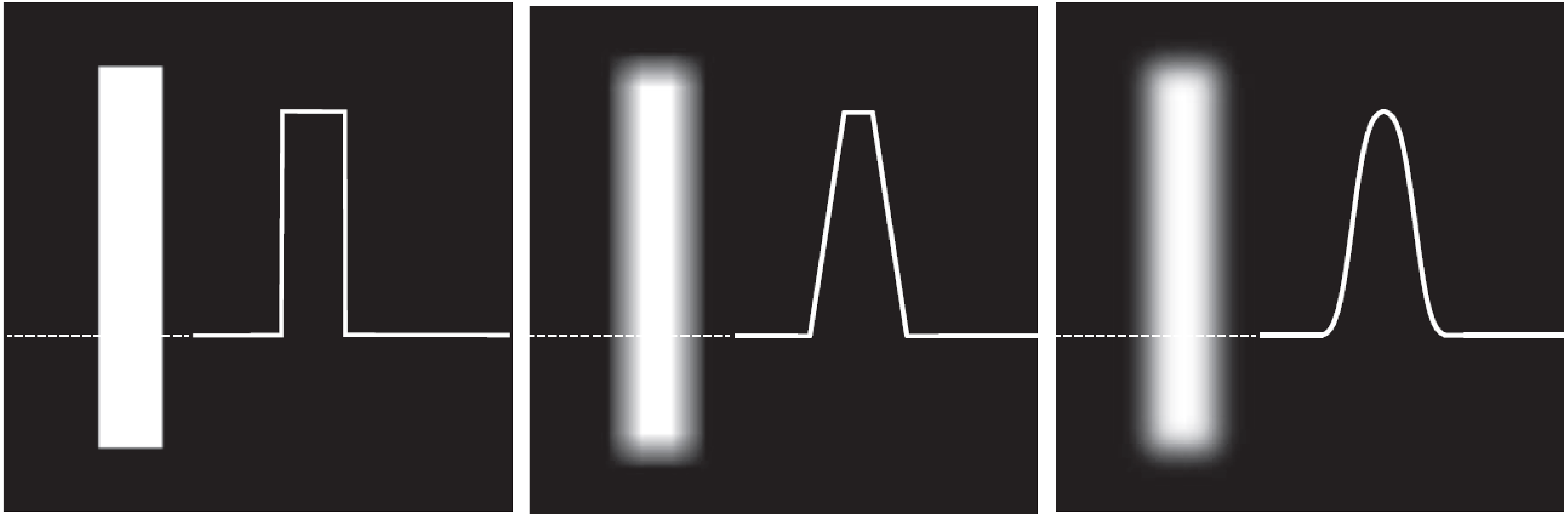

a b c

FIGURE 3.36 (a)A test pattern of size 1024×1024. (b) Result of lowpass filtering the pattern with a Gaussian kernel of size 21×21, with standard deviations $\sigma = 3.5$. (c) Result of using a kernel of size 43×43, with $\sigma = 7$. This result is comparable to Fig. 3.33(d). We used $K = 1$ in all cases.



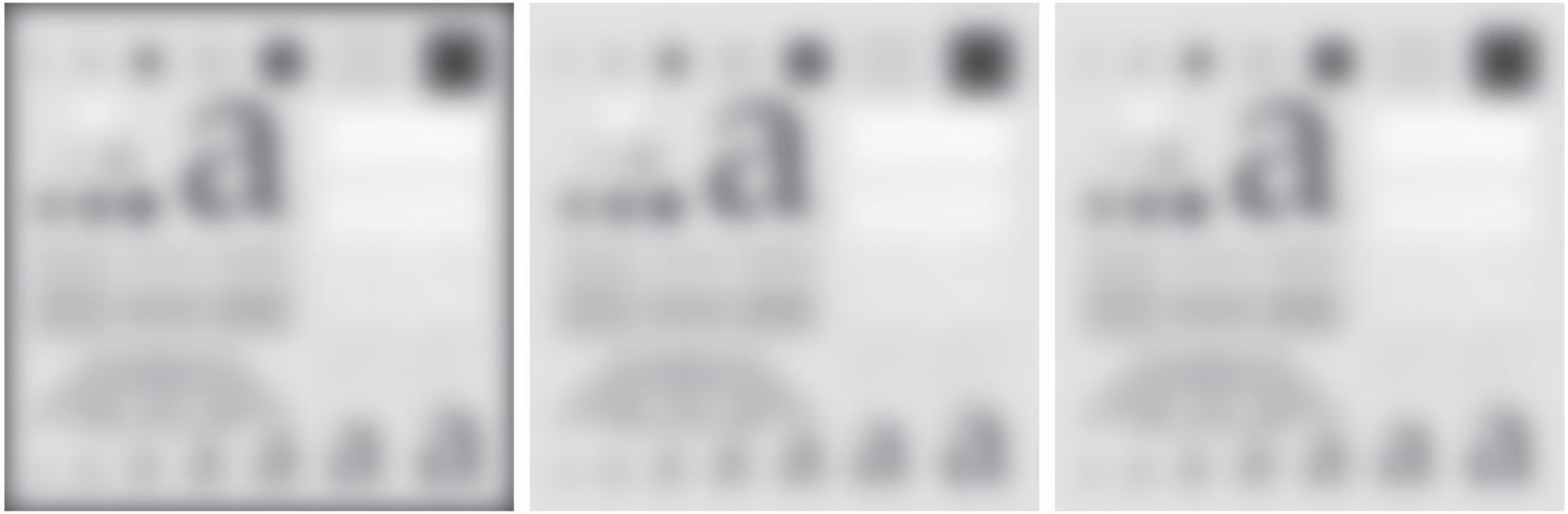
a b c

FIGURE 3.37 (a) Result of filtering Fig. 3.36(a) 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.



a b c

FIGURE 3.38 (a) Image of a white rectangle on a black background, and a horizontal intensity profile along the scan line shown dotted. (b) Result of smoothing this image with a box kernel of size 71×71, and corresponding intensity profile. (c) Result of smoothing the image using a Gaussian kernel of size 151×151, with $K = 1$ and $\sigma = 25$. Note the smoothness of the profile in (c) compared to (b). The image and rectangle are of sizes 1024×1024 and 768×128 pixels, respectively.



a b c

FIGURE 3.39 Result of filtering the test pattern in Fig. 3.36(a) using (a) zero padding, (b) mirror padding, and (c) replicate padding. A Gaussian kernel of size 187×187, with $K = 1$ and $\sigma = 31$ was used in all three cases.

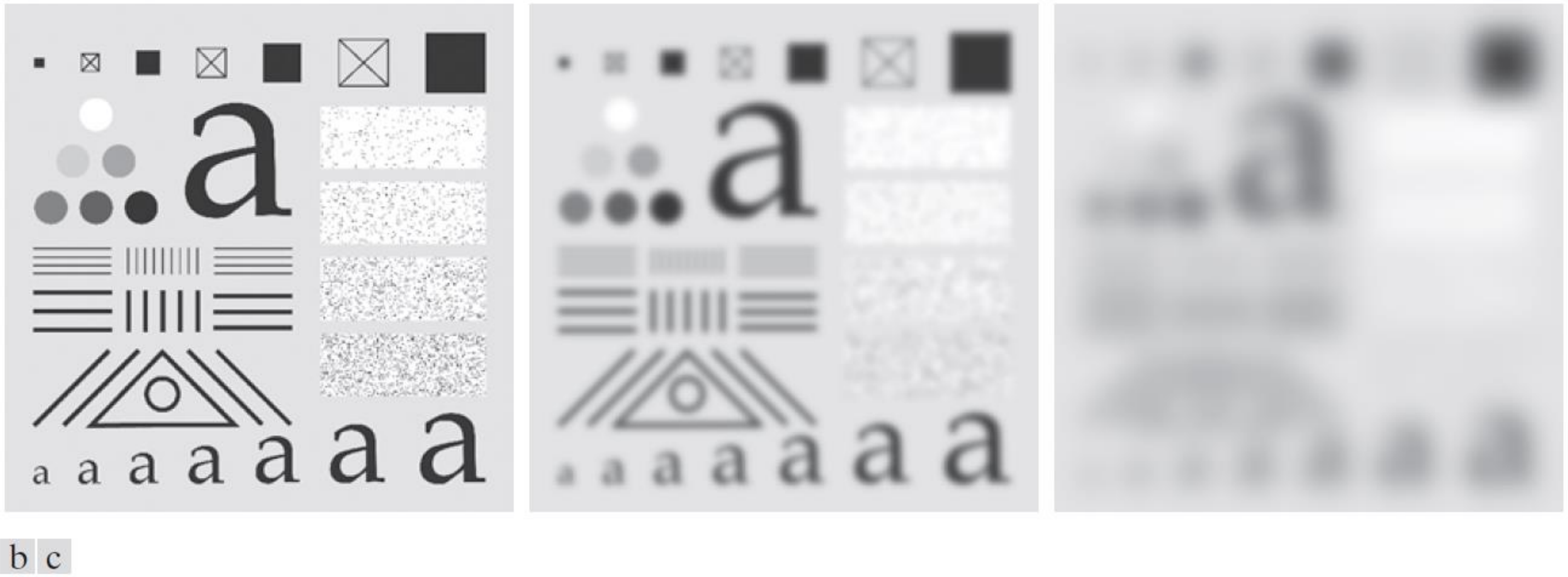


FIGURE 3.40 (a) Test pattern of size 4096 × 4096 pixels. (b) Result of filtering the test pattern with the same Gaussian kernel used in Fig. 3.39. (c) Result of filtering the pattern using a Gaussian kernel of size 745 × 745 elements, with $K = 1$ and $\sigma = 124$. Mirror padding was used throughout.

Importance of understanding relationship between kernel size and the size of objects

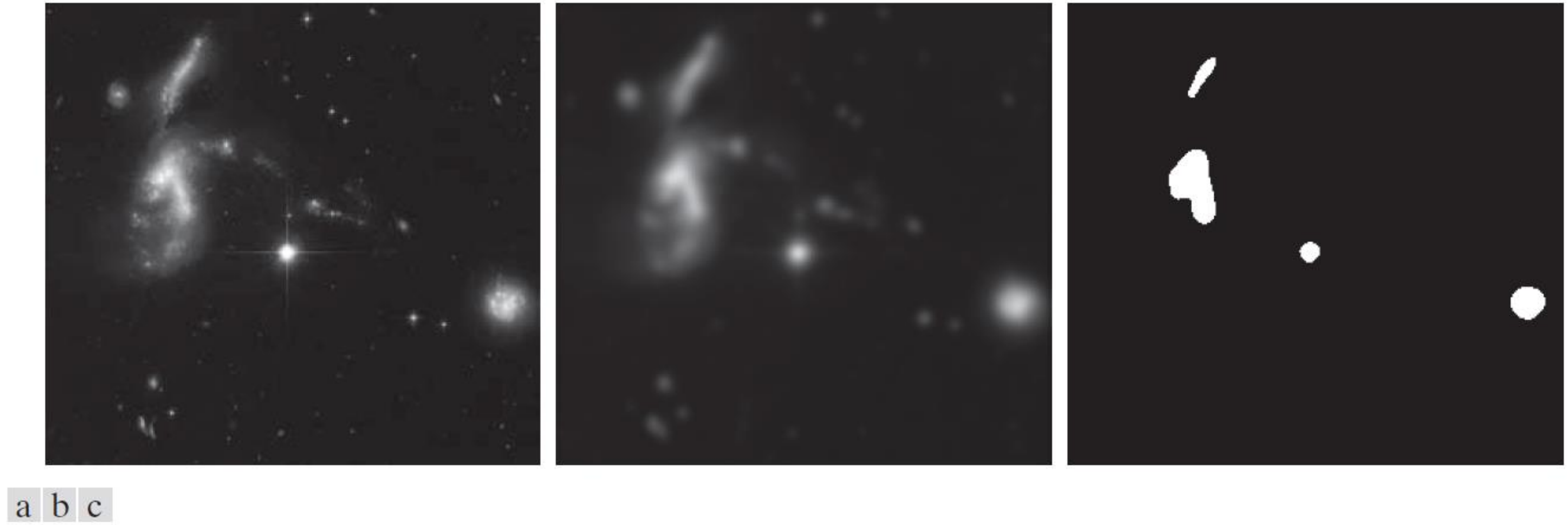


FIGURE 3.41 (a) A 2566 × 2758 Hubble Telescope image of the *Hickson Compact Group*. (b) Result of lowpass filtering with a Gaussian kernel. (c) Result of thresholding the filtered image (intensities were scaled to the range [0, 1]). The Hickson Compact Group contains dwarf galaxies that have come together, setting off thousands of new star clusters. (Original image courtesy of NASA.)

By smoothing the original image we get rid of lots of the finer detail which leaves only the gross features for thresholding

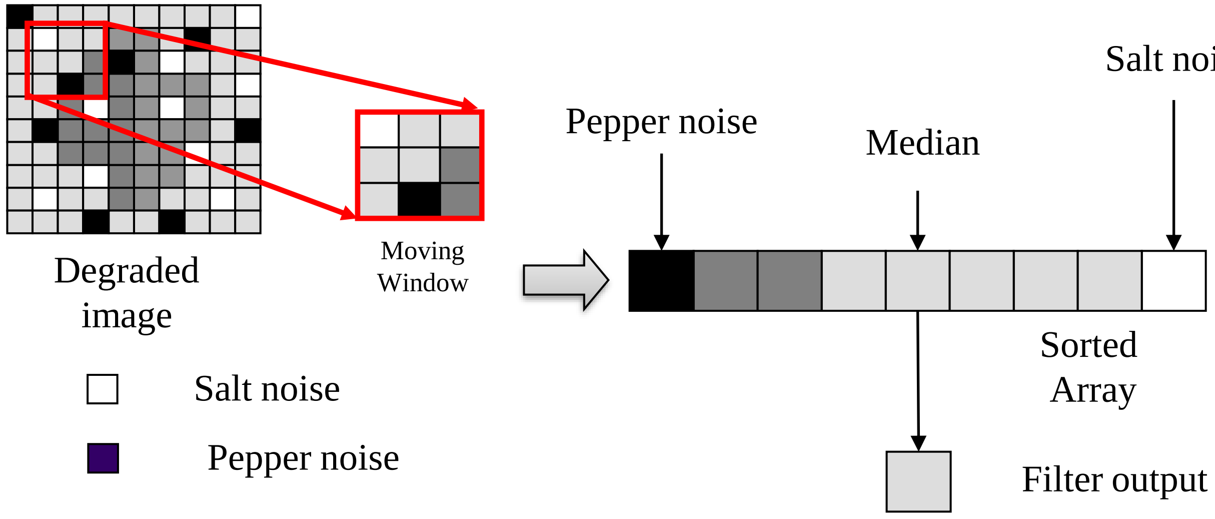
Order-statistic (Nonlinear) Filters

- 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 Filter: How it works

$$\hat{f}(x,y) = \underset{(s,t)\in S_{xy}}{median}\{g(s,t)\}$$

A median filter is good for removing impulse, isolated noise

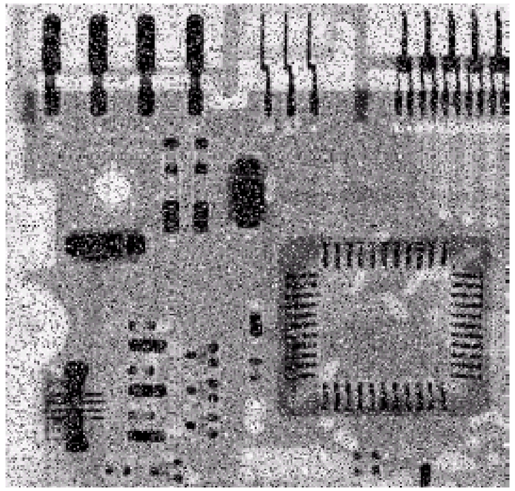


□ Salt noise

■ Pepper noise

Normally, impulse noise has high magnitude and is solated. When we sort pixels in the moving window, noise pixels are usually at the ends of the array.

Averaging Filter Vs. Median Filter Example


Original Image With Noise

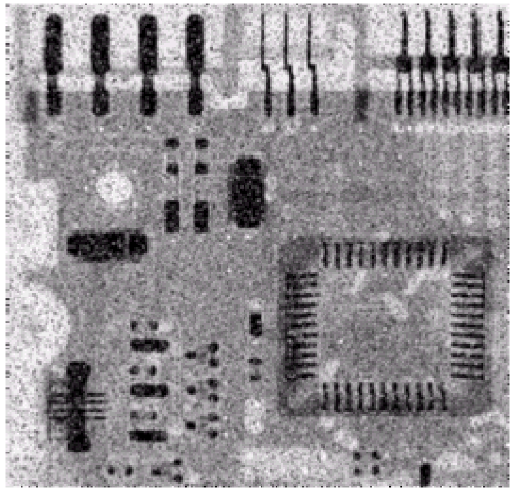


Image After Averaging Filter

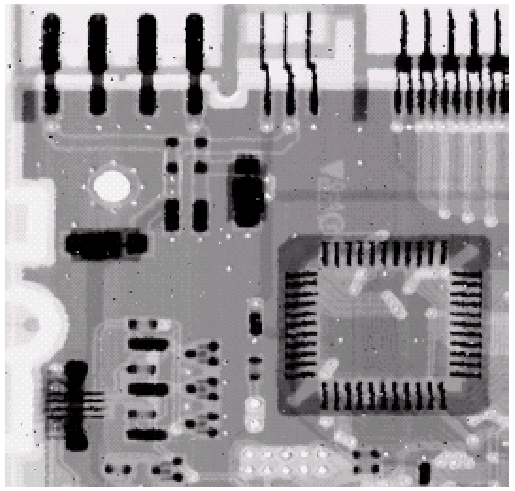


Image After Median Filter

Filtering is often used to remove noise from images

Sometimes a median filter works better than an averaging filter

Max and Min Filter

Max Filter:

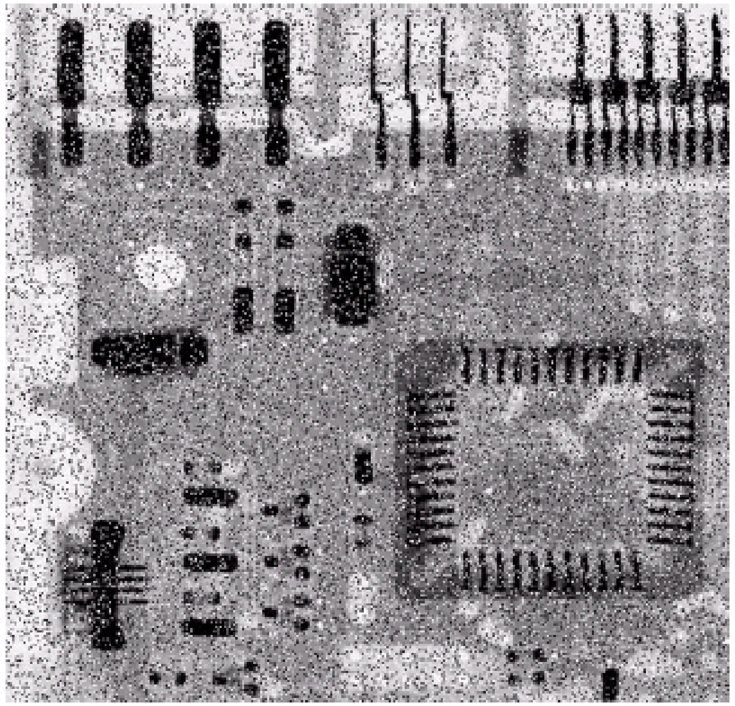
$$\hat{f}(x,y) = \max_{(s,t)\in S_{xy}} \{g(s,t)\}$$

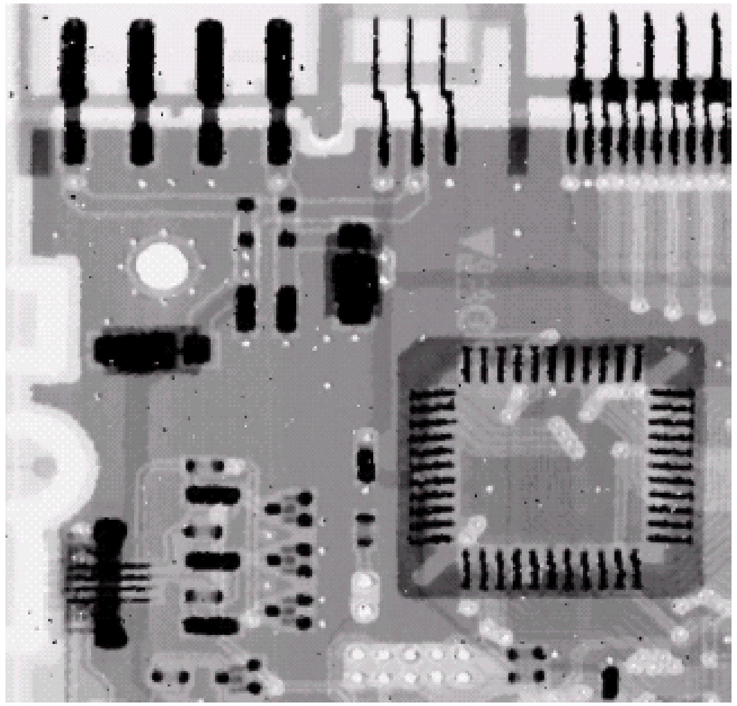
Min Filter:

$$\hat{f}(x,y) = \min_{(s,t)\in S_{xy}} \{g(s,t)\}$$

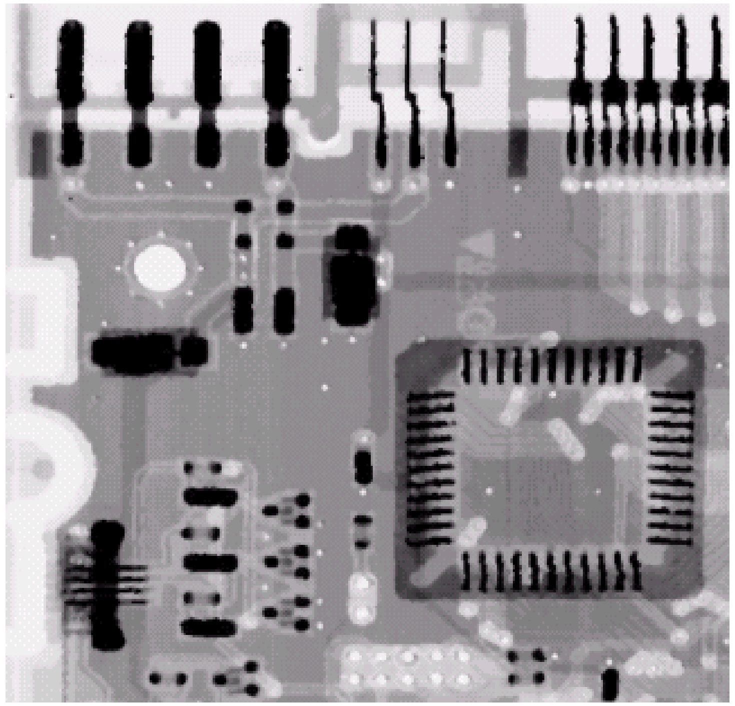
Max filter is good for pepper noise and Min filter is good for salt noise.

bipolar Noise
$P_a = 0.1$
$P_b = 0.1$

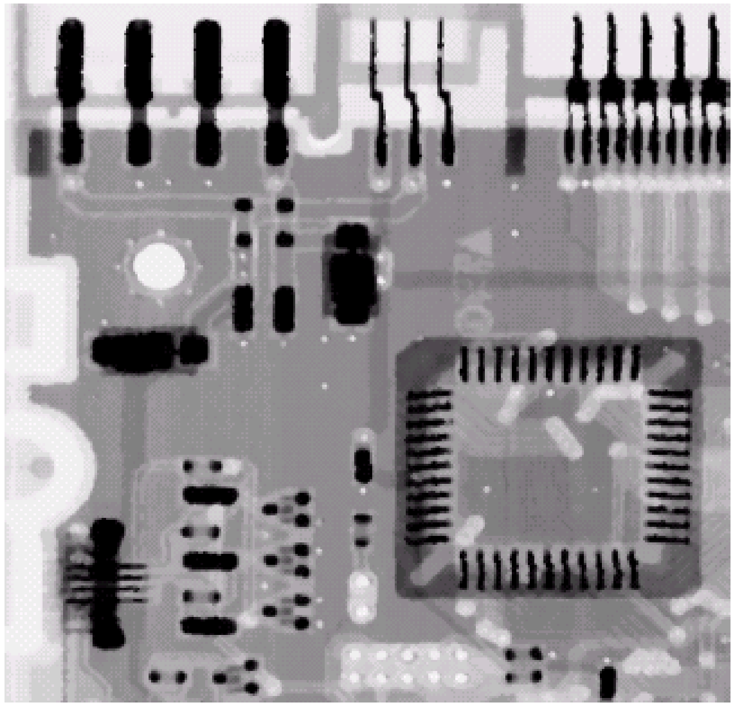




3x3 Median Filter Pass 1

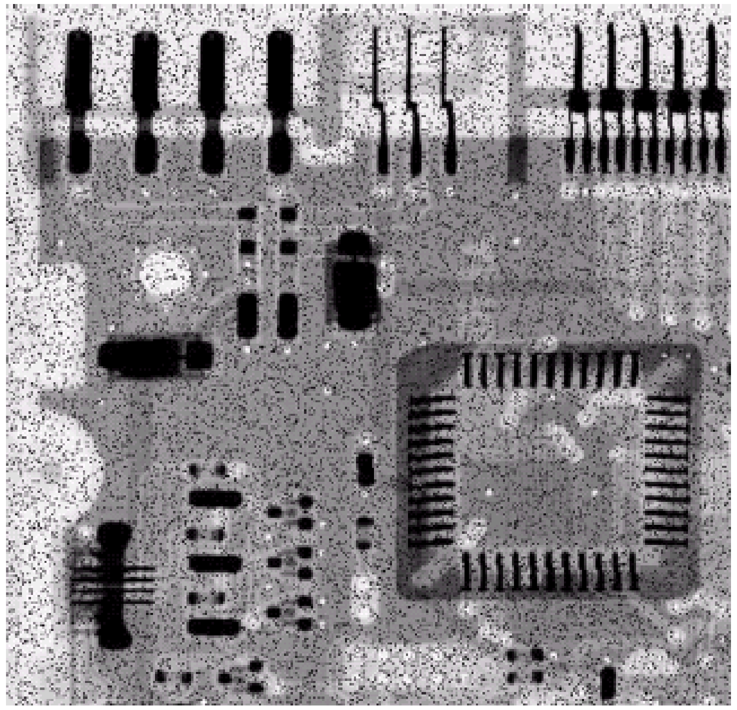


3x3 Median Filter Pass 2

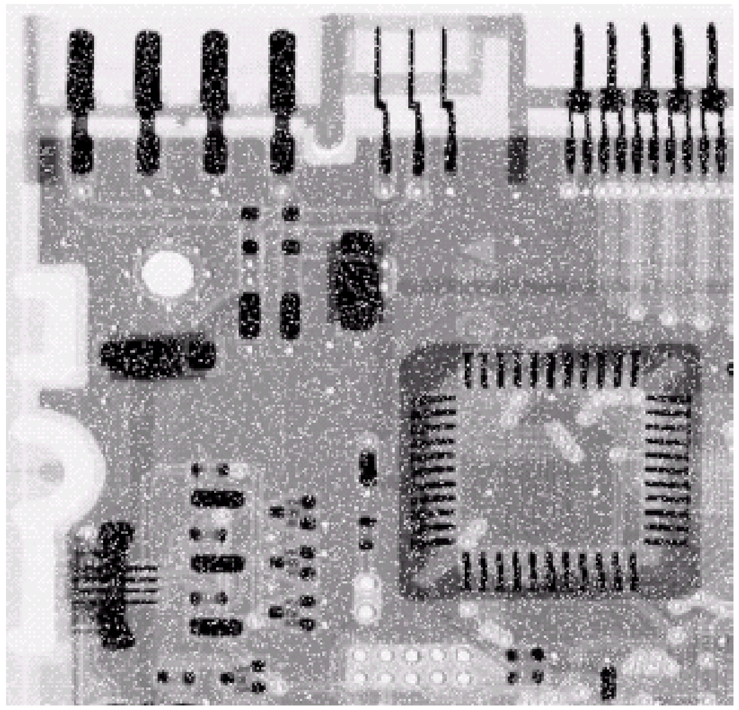


3x3 Median Filter Pass 3

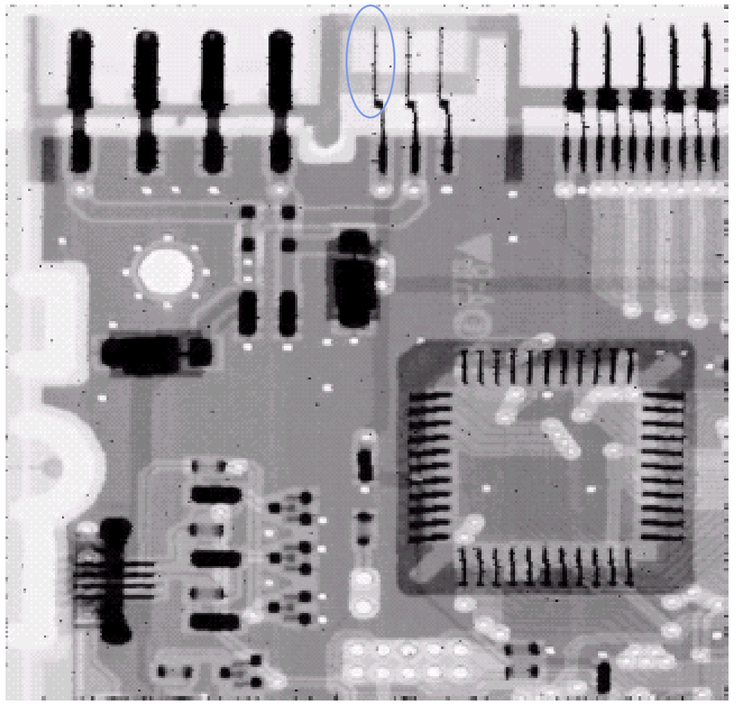
Pepper noise



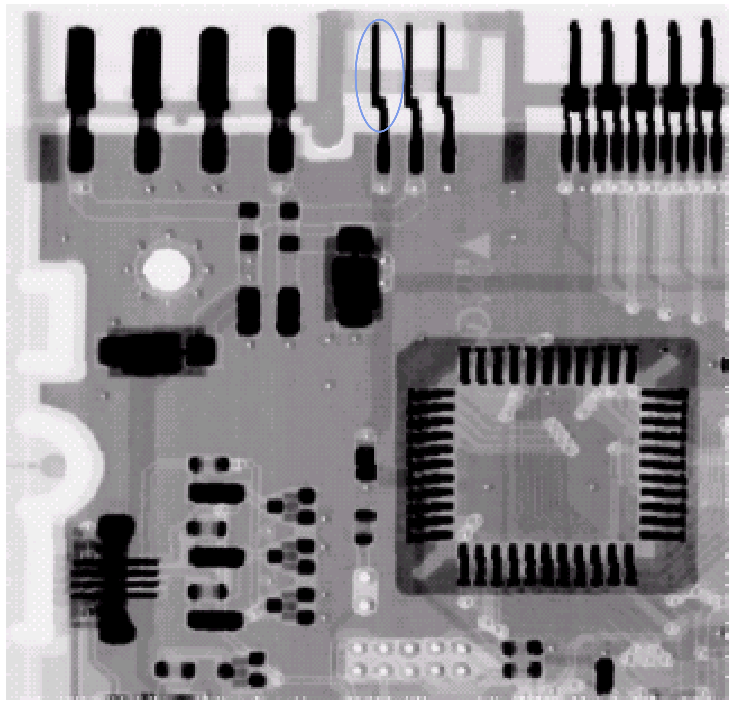
Salt noise



Max filter



Min filter



Spatial smoothing and image approximation

Spatial smoothing may be viewed as a process for estimating the value of a pixel from its neighbours.

What is the value that “best” approximates the intensity of a given pixel given the intensities of its neighbours?

We have to define “best” by establishing a criterion.

Spatial smoothing and image approximation (cont...)

A standard criterion is the the sum of squares differences.

$$E = \sum_{i=1}^{N} [x(i) - m]^2 \quad \Leftrightarrow m = \arg\min_{m} \left\{ \sum_{i=1}^{N} [x(i) - m]^2 \right\}$$

$$\frac{\partial E}{\partial m} = 0 \quad \Leftrightarrow -2 \sum_{i=1}^{N} \left(x(i) - m \right) = 0 \quad \Leftrightarrow \sum_{i=1}^{N} x(i) = \sum_{i=1}^{N} m$$

$$\Leftrightarrow \sum_{i=1}^{N} x(i) = Nm \quad \Leftrightarrow m = \frac{1}{N} \sum_{i=1}^{N} x(i)$$

The average value

Spatial smoothing and image approximation (cont...)

Another criterion is the the sum of absolute differences.

$$E = \sum_{i=1}^{N} \left| x(i) - m \right| \quad \Leftrightarrow m = \arg\min_{m} \left\{ \sum_{i=1}^{N} \left| x(i) - m \right| \right\}$$

$$\frac{\partial E}{\partial m} = 0 \Leftrightarrow -\sum_{i=1}^{N} sgn\big(x(i) - m\big) = 0, \quad sign(x) = \begin{cases} 1 & x > 0 \\ 0 & x = 0 \\ -1 & x < 0 \end{cases}$$

There must be equal in quantity positive and negative values.

$$m = \text{median}\{x(i)\}$$

Spatial smoothing and image approximation (cont...)

- The median filter is non linear:

$$\text{median}\{x + y\} \neq \text{median}\{x\} + \text{median}\{y\}$$

- It works well for impulse noise (e.g. salt and pepper).
- It requires sorting of the image values.
- It preserves the edges better than an average filter in the case of impulse noise.
- It is robust to impulse noise at 50%.

Spatial smoothing and image approximation (cont...)

Example

x[n]	**1**	**1**	**1**	**1**	**1**	**2**	**2**	**2**	**2**	**2**

edge

Impulse noise

x[n]	**1**	**3**	**1**	**1**	**1**	**2**	**3**	**2**	**2**	**3**

Median (N=3)

x[n]	**-**	**1**	**1**	**1**	**1**	**2**	**2**	**2**	**2**	**-**

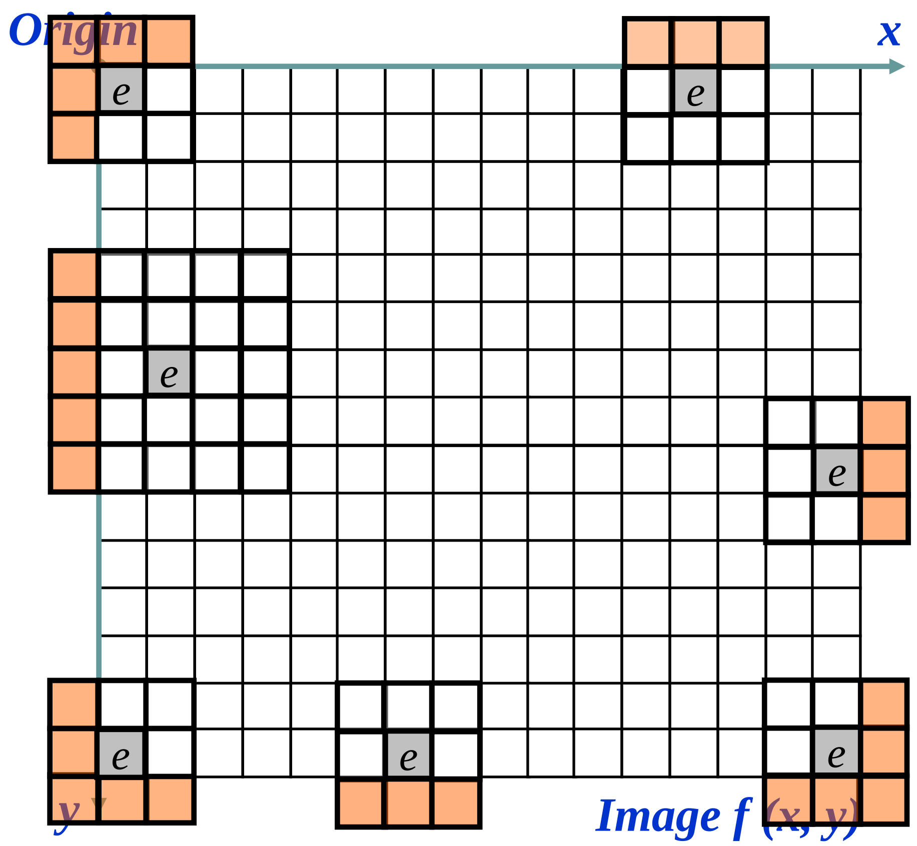
Average (N=3)

x[n]	**-**	**1.7**	**1.7**	**1**	**1.3**	**2**	**2.3**	**2.3**	**2.2**	**-**

The edge is smoothed

Strange Things Happen At The Edges!

At the edges of an image we are missing pixels to form a neighbourhood

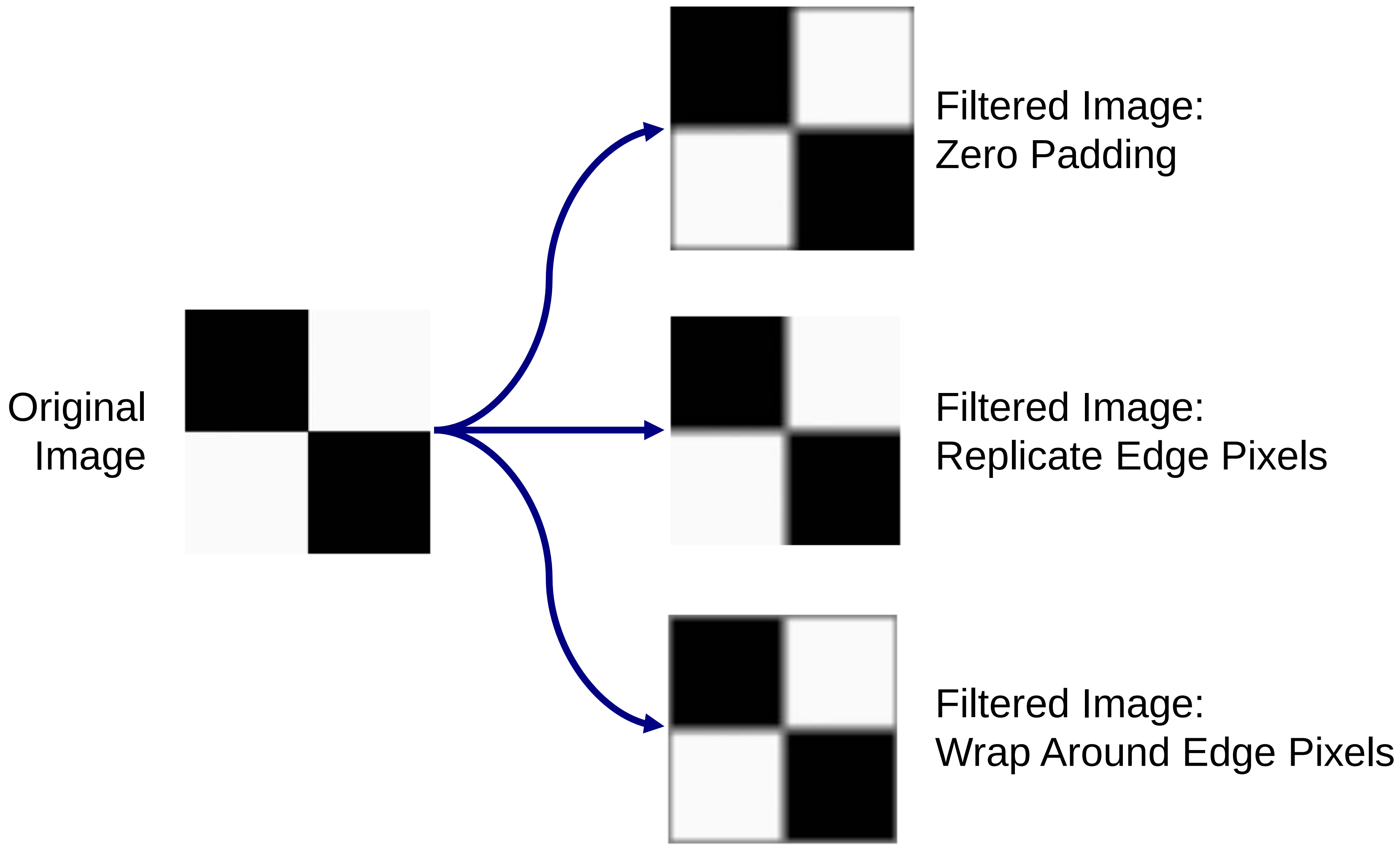


Strange Things Happen At The Edges! (cont…)

There are a few approaches to dealing with missing edge pixels:

- Omit missing pixels
  - Only works with some filters
  - Can add extra code and slow down processing
- Pad the image
  - Typically with either all white or all black pixels
- Replicate border pixels
- Truncate the image
- Allow pixels *wrap around* the image
  - Can cause some strange image artefacts

Strange Things Happen At The Edges! (cont…)


Original Image
Filtered Image: Zero Padding
Filtered Image: Replicate Edge Pixels
Filtered Image: Wrap Around Edge Pixels

Effect of Low Pass Filtering on White Noise

Let f be an observed instance of the image f_0 corrupted by noise w:

$$f = f_0 + w$$

with noise samples having mean value $E[w(n)]=0$ and being uncorrelated with respect to location:

$$E[w(m)w(n)] = \begin{cases} \sigma^2, & m = n \\ 0, & m \neq n \end{cases}$$

Effect of Low Pass Filtering on White Noise (cont...)

Applying a low pass filter h (e.g. an average filter) by convolution to the degraded image:

$$g = h * f = h * (f_0 + w) = h * f_0 + h * w$$

The expected value of the output is:

$$E[g] = E[h * f_0] + E[h * w] = h * f_0 + h * E[w]$$

$$= h * f_0 + h * 0 = h * f_0$$

The noise is removed in average.

Effect of Low Pass Filtering on White Noise (cont...)

What happens to the standard deviation of g?

Let $g = h * f_0 + h * w = \overline{f_0} + \overline{w}$

where the bar represents filtered versions of the signals, then

$$\sigma_g^2 = E[g^2] - \left(E[g]\right)^2 = \mathrm{E}[(\overline{f_0} + \overline{w})^2] - (\overline{f_0})^2$$

$$= \mathrm{E}[(\overline{f_0})^2 + (\overline{w})^2 + 2\overline{f_0}\,\overline{w}] - (\overline{f_0})^2$$

$$= E[(\overline{w})^2] + 2E[\overline{f_0}]E[\overline{w}] = E[(\overline{w})^2]$$

Effect of Low Pass Filtering on White Noise (cont...)

Considering that h is an average filter, we have at pixel n:

$$\overline{w}(n) = (h * w)(n) = \frac{1}{N} \sum_{k \in \Gamma(n)} w(k)$$

Therefore,

$$E[(\overline{w}(n))^2] = E\left[\left(\frac{1}{N} \sum_{k \in \Gamma(n)} w(k) \right)^2 \right]$$

Effect of Low Pass Filtering on White Noise (cont...)

$$E\left[\left(\frac{1}{N}\sum_{k\in\Gamma(n)} w(k)\right)^2\right]$$

$$=\frac{1}{N^2}\sum_{k\in\Gamma(n)} E\left[\{w(k)\}^2\right] \longrightarrow \text{Sum of squares}$$

$$+\frac{2}{N^2}\sum_{l\in\Gamma(n)}\sum_{\substack{m\in\Gamma(n)\\ m\neq l}} E\left[w(n-l)w(n-m)\right] \searrow$$

Cross products

Effect of Low Pass Filtering on White Noise (cont...)

Sum of squares

$$\frac{1}{N^2}\sum_{k\in\Gamma(n)} E\left[\left\{w(k)\right\}^2\right] = \frac{1}{N^2}\sum_{k\in\Gamma(n)} \sigma^2$$

Cross products (uncorrelated as $m \neq l$)

$$+\frac{2}{N^2}\sum_{l\in\Gamma(n)}\sum_{\substack{m\in\Gamma(n)\\ m\neq l}} E\left[w(n-l)w(n-m)\right] = 0$$

Effect of Low Pass Filtering on White Noise (cont...)

Finally, substituting the partial results:

$$\sigma_g^2 = E\left[\left(\frac{1}{N}\sum_{k\in\Gamma(n)} w(k)\right)^2\right] = \frac{1}{N^2}\sum_{k\in\Gamma(n)} \sigma^2$$

$$= \frac{1}{N^2} N\sigma^2 \quad = \frac{\sigma^2}{N}$$

The effect of the noise is reduced.
This processing is not optimal as it also smoothes image edges.

Sharpening Spatial Filters

Previously we have looked at smoothing filters which remove fine detail

Sharpening spatial filters seek to highlight fine detail

- Remove blurring from images
- Highlight edges

Sharpening filters are based on *spatial differentiation*

1st Derivative

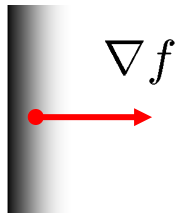
The formula for the 1st derivative of a function is as follows:

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

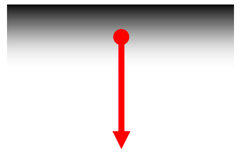
It's just the difference between subsequent values and measures the rate of change of the function

1st Derivative (cont.)

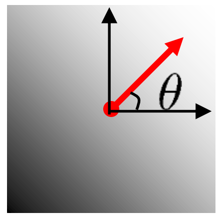
- The gradient of an image: $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$



$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$



$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$



$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$

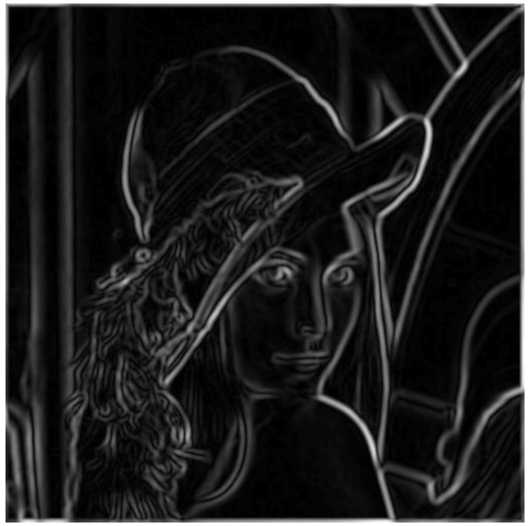
The gradient points in the direction of most rapid increase in intensity.

Gradient direction $\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$

The *edge strength* is given by the gradient magnitude

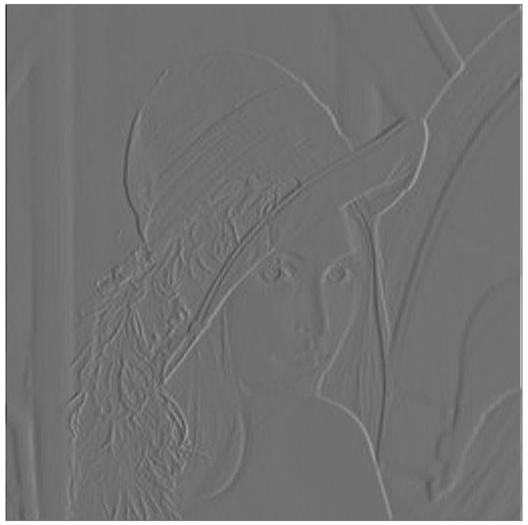
$$||\nabla f|| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

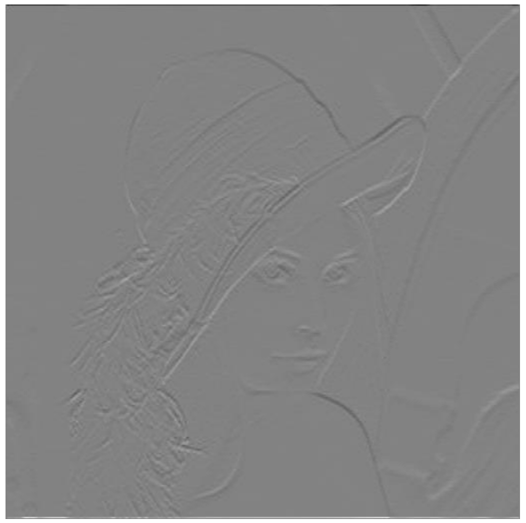
1st Derivative (cont.)



$$\|\nabla f\|$$

$$\frac{\partial f}{\partial x}$$





$$\frac{\partial f}{\partial y}$$

2nd Derivative

The formula for the 2nd derivative of a function is as follows:

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

Simply takes into account the values both before and after the current value

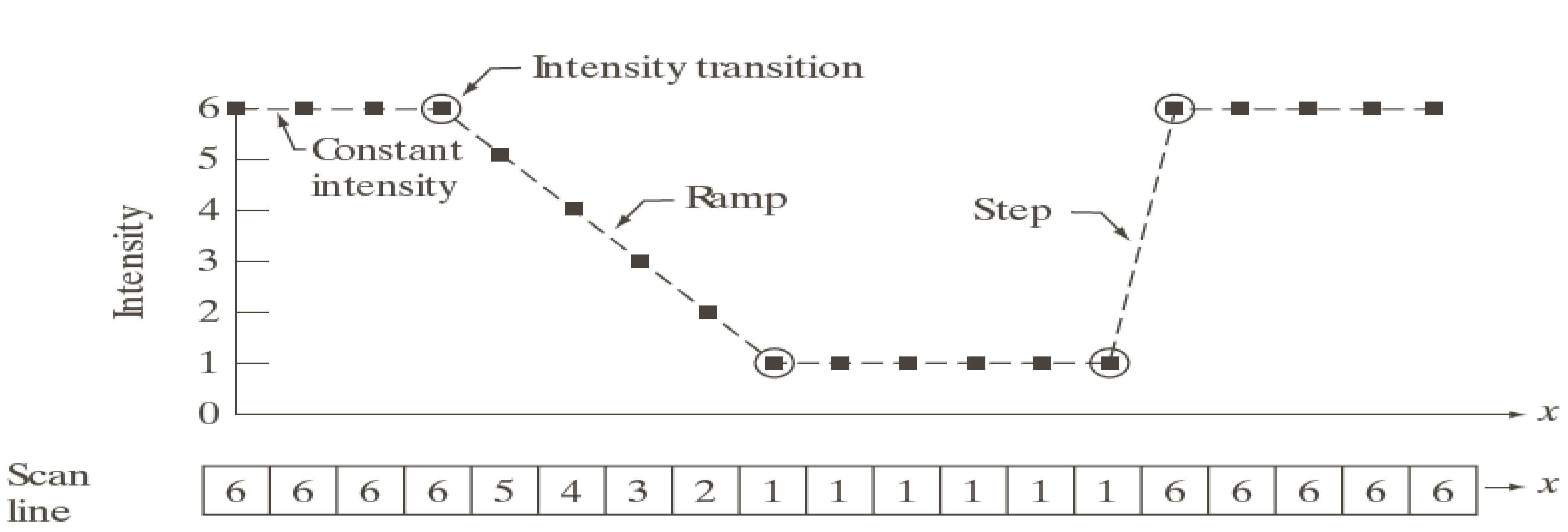
Step and Ramp

- Step edge
  - The image intensity abruptly changes from one value to one side of the discontinuity to a different value on the opposite side.
- Ramp edge
  - A step edge where the intensity change is not instantaneous but occur over a finite distance.

Sharpening Spatial Filters: Foundation

Definition for derivatives

	First Derivative	Second Derivative
Constant Intensity Areas	ZERO	ZERO
Onset of intensity step or ramp	Non-zero	Non-zero + at the end
Along intensity ramp	Non-zero	ZERO



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.

Adapted from Gonzalez, Rafael C. Digital image processing. Pearson education India, 2009.

Using Second Derivatives For Image Enhancement

Edges in images are often ramp-like transitions

- 1st derivative is constant and produces thick edges
- 2nd derivative zero crosses the edge (double response at the onset and end with opposite signs)

A common sharpening filter is the *Laplacian*

- Isotropic
- One of the simplest sharpening filters
- We will look at a digital implementation

The Laplacian

The Laplacian is defined as follows:

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

where the partial 1st order derivative in the x direction is defined as follows:

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

and in the y direction as follows:

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

The Laplacian (cont…)

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

0	1	0
1	-4	1
0	1	0

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

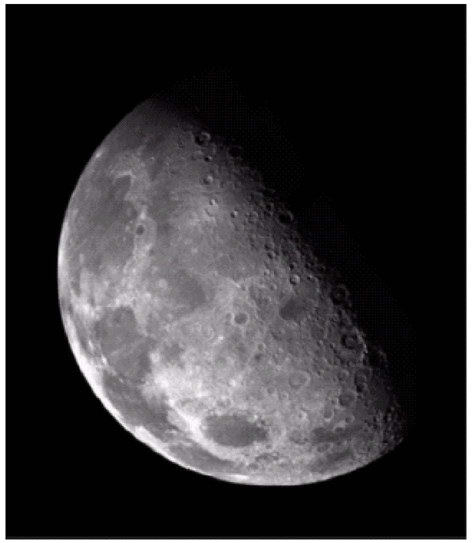
−1	−1	−1
−1	8	−1
−1	−1	−1

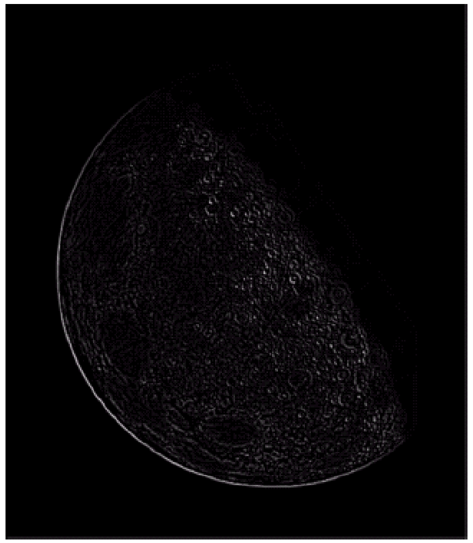
a b c d

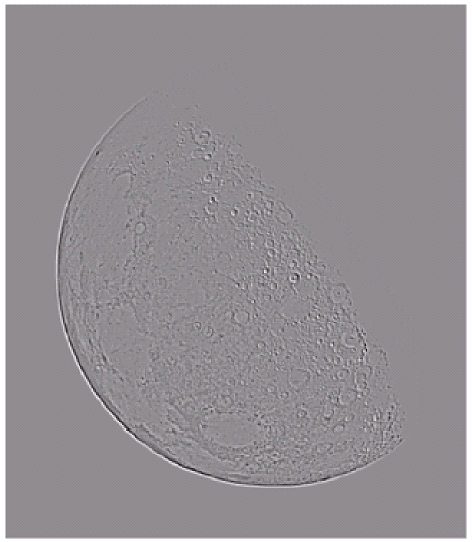
FIGURE 3.45 (a) Laplacian kernel used to implement Eq. (3-53). (b) Kernel used to implement an extension of this equation that includes the diagonal terms. (c) and (d) Two other Laplacian kernels.

The Laplacian (cont…)

Applying the Laplacian to an image we get a new image that highlights edges and other discontinuities


Original Image


Laplacian Filtered Image


Laplacian Filtered Image Scaled for Display

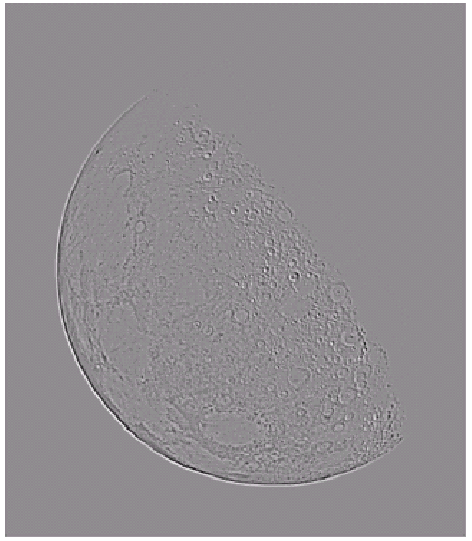
But That Is Not Very Enhanced!

The result of a Laplacian filtering is not an enhanced image

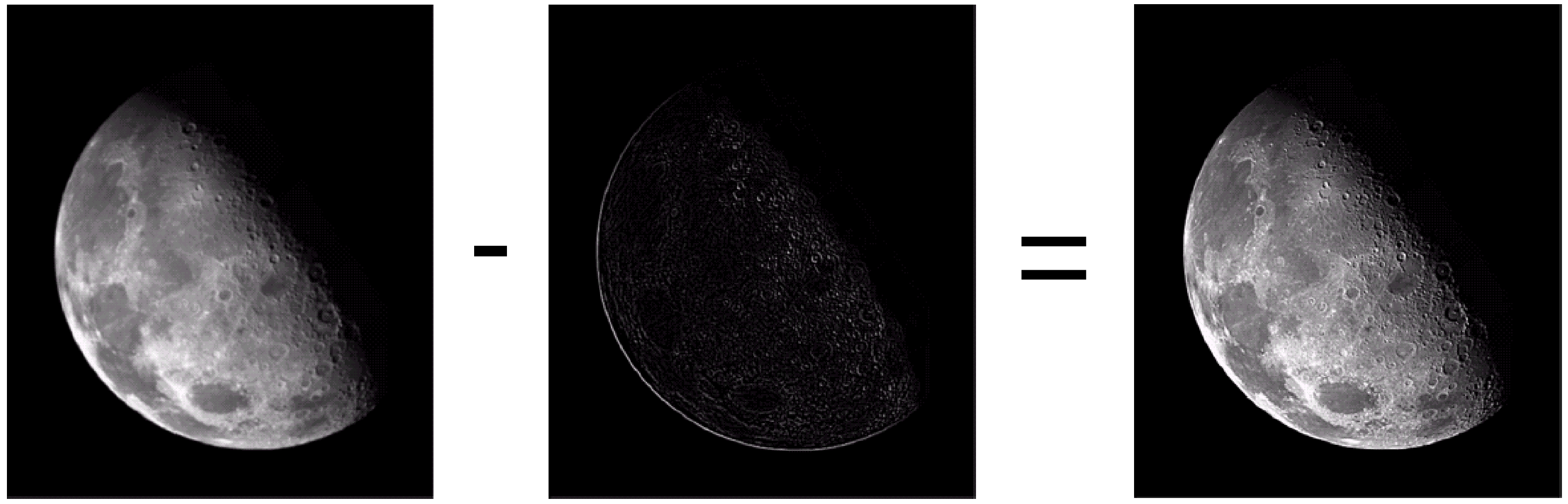
We have to do more work in order to get our final image

Subtract the Laplacian result from the original image to generate our final sharpened enhanced image

$$g(x,y) = f(x,y) - \nabla^2 f$$


Laplacian Filtered Image Scaled for Display

Laplacian Image Enhancement



Original Image

Laplacian Filtered Image

Sharpened Image

In the final sharpened image edges and fine detail are much more obvious

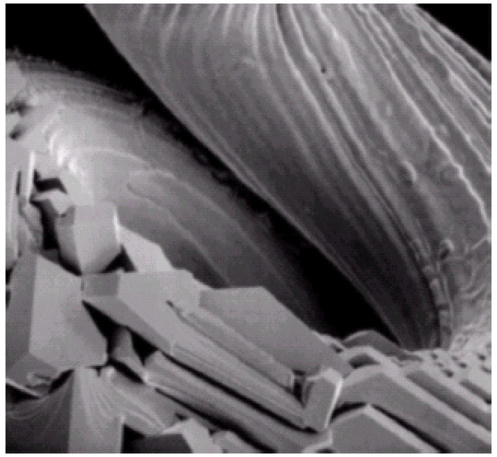
Simplified Image Enhancement

The entire enhancement can be combined into a single filtering operation

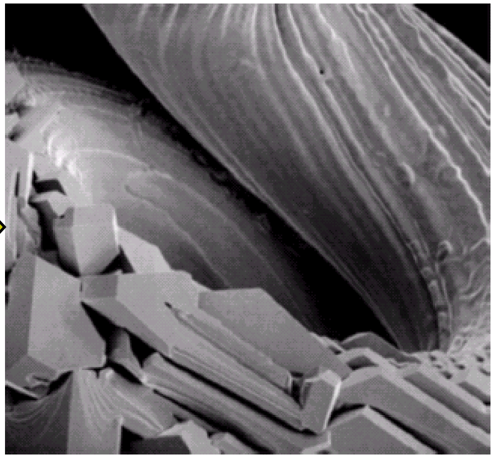
$$
\begin{aligned}
g(x,y) &= f(x,y) - \nabla^2 f \\
&= f(x,y) - [f(x+1,y) + f(x-1,y) \\
&\qquad + f(x,y+1) + f(x,y-1) \\
&\qquad - 4f(x,y)] \\
&= 5f(x,y) - f(x+1,y) - f(x-1,y) \\
&\qquad - f(x,y+1) - f(x,y-1)
\end{aligned}
$$

Simplified Image Enhancement (cont…)

This gives us a new filter which does the whole job for us in one step



0	-1	0
-1	5	-1
0	-1	0



Variants On The Simple Laplacian

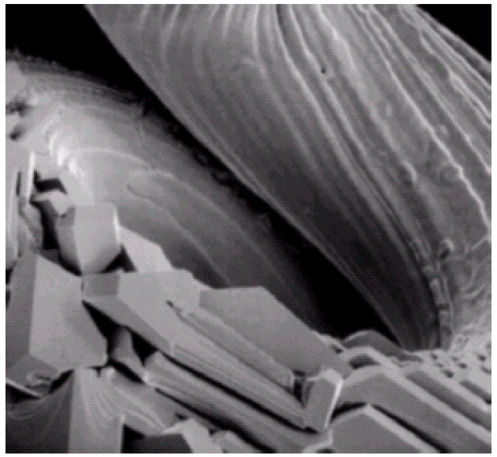
There are lots of slightly different versions of the Laplacian that can be used:

0	1	0
1	-4	1
0	1	0

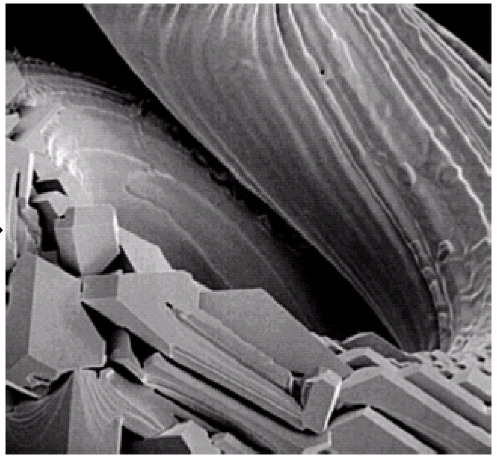
Simple Laplacian

1	1	1
1	-8	1
1	1	1

Variant of Laplacian



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



Unsharp masking and Highboost Filtering

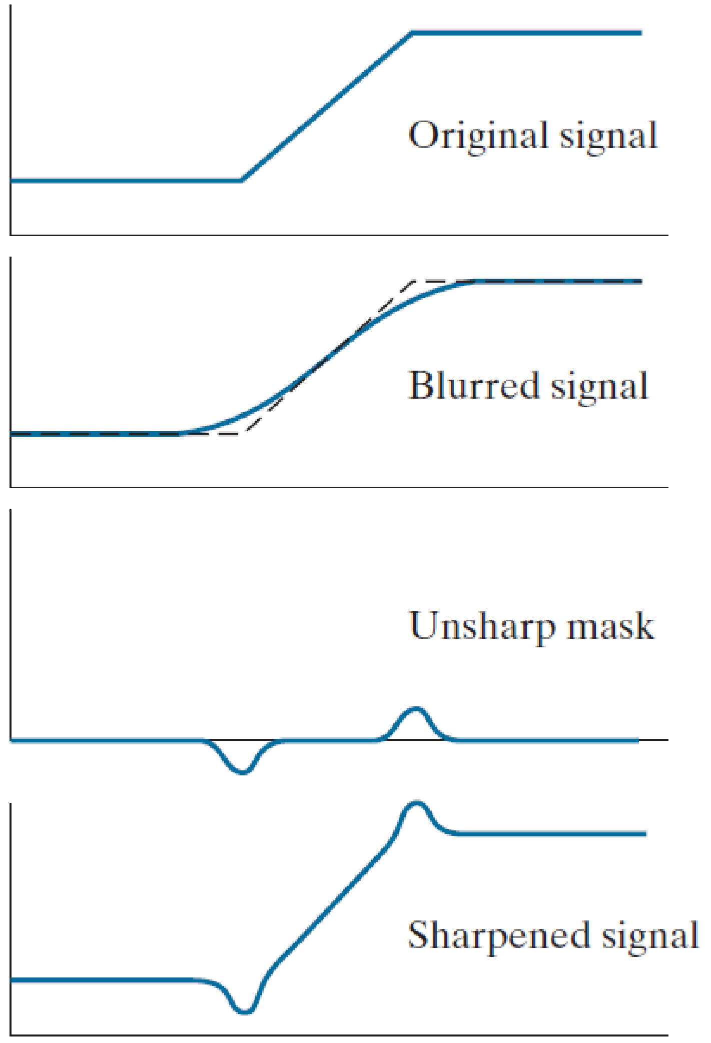
Used by the printing industry

Subtracts an unsharped (smooth) image from the original image $f(x,y)$.

- Blur the image $b(x,y)=Blur\{f(x,y)\}$
- Subtract the blurred image from the original (the result is called the *mask*) $g_{mask}(x,y)=f(x,y)-b(x,y)$
- Add the mask to the original with k non negative masks

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

Unsharp masking and Highboost Filtering (cont...)



a
b
c
d

FIGURE 3.48
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).

Unsharp masking and Highboost Filtering (cont…)

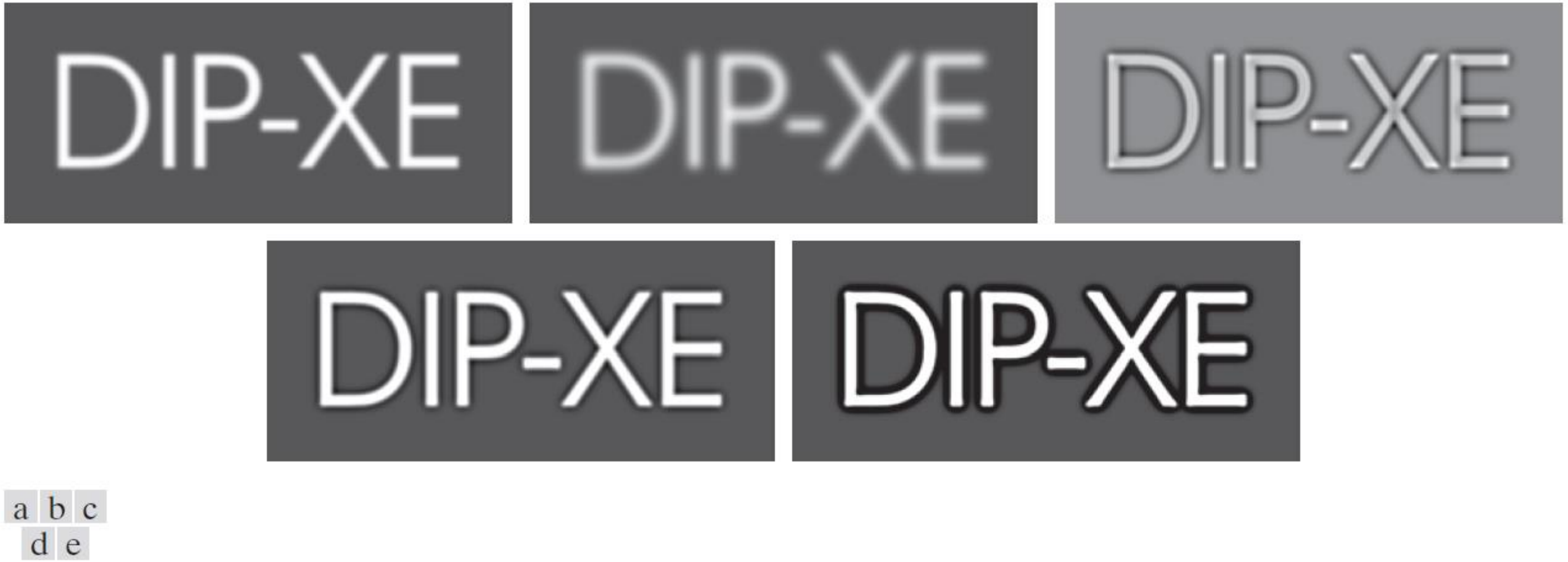


FIGURE 3.49 (a) Original image of size 600 × 259 pixels. (b) Image blurred using a 31 × 31 Gaussian lowpass filter with $\sigma = 5$. (c) Mask. (d) Result of unsharp masking using Eq. (3-56) with $k = 1$. (e) Result of highboost filtering with $k = 4.5$.

1st Derivative Filtering

Implementing 1st derivative filters is difficult in practice

For a function *f(x, y)* the gradient of *f* at coordinates *(x, y)* is given as the column vector:

$$\nabla \mathrm{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f}{\partial x} \\ \dfrac{\partial f}{\partial y} \end{bmatrix}$$

1st Derivative Filtering (cont…)

The magnitude of this vector is given by:

$$\begin{aligned}\nabla f &= mag(\nabla \mathrm{f}) \\ &= \left[G_x^2 + G_y^2\right]^{1/2} \\ &= \left[\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2\right]^{1/2}\end{aligned}$$

For practical reasons this can be simplified as:

$$\nabla f \approx \left|G_x\right| + \left|G_y\right|$$

1st Derivative Filtering (cont…)

Roberts Cross−gradient Operators

$M(x,y) \approx |z_9 - z_5| + |z_8 - z_6|$

Sobel Operators

$$M(x,y) \approx |(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)| + |(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)|$$

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

a
b c
d e

FIGURE 3.50
(a) A 3 × 3 region of an image, where the zs are intensity values.
(b)–(c) Roberts cross-gradient operators.
(d)–(e) Sobel operators. All the kernel coefficients sum to zero, as expected of a derivative operator.

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	0
0	1

0	−1
1	0

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

Sobel Example

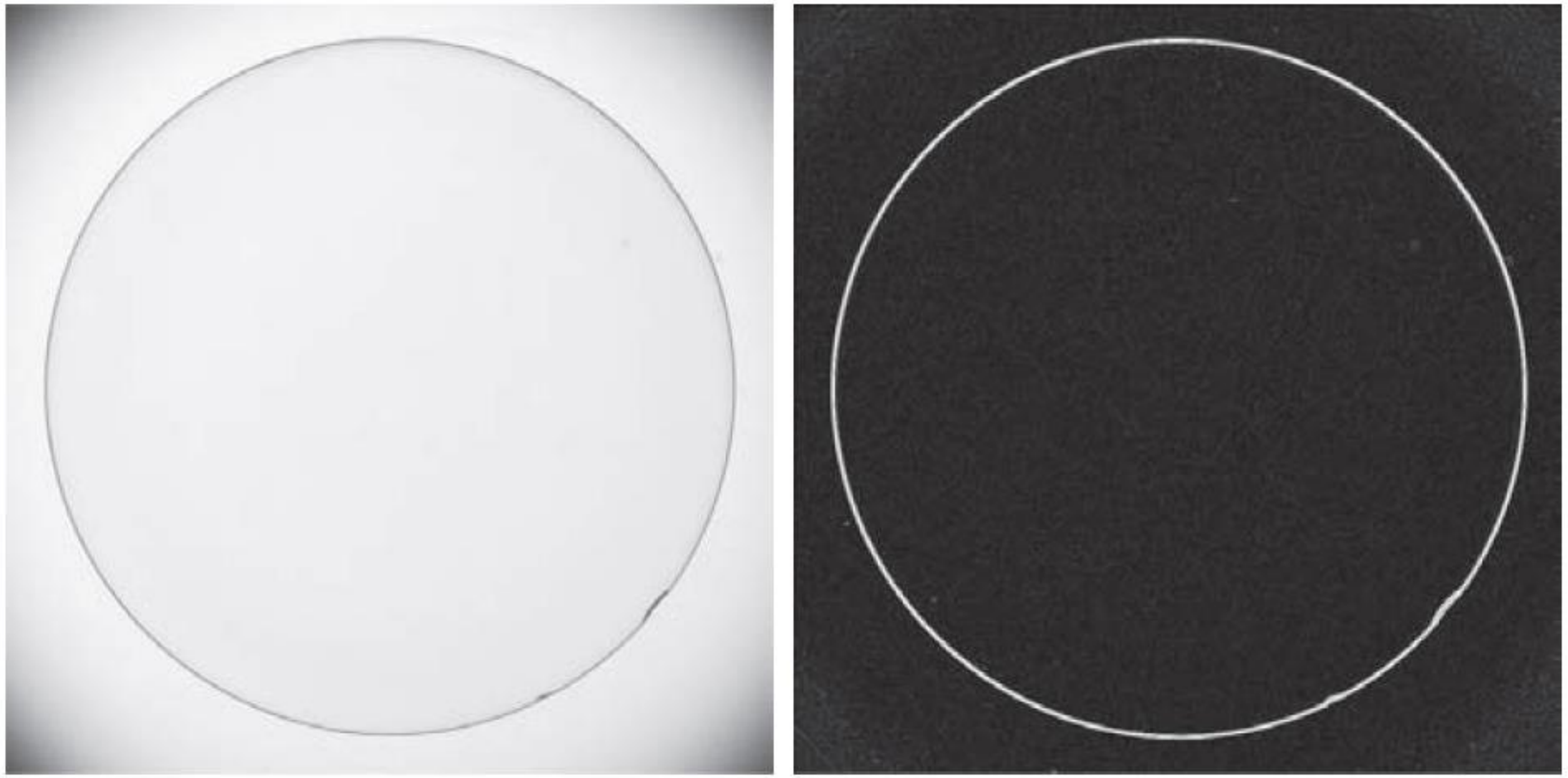


FIGURE 3.51
(a) Image of a contact lens (note defects on the boundary at 4 and 5 o'clock).
(b) Sobel gradient.
(Original image courtesy of Perceptics Corporation.)

1st & 2nd Derivatives

Comparing the 1st and 2nd derivatives we can conclude the following:

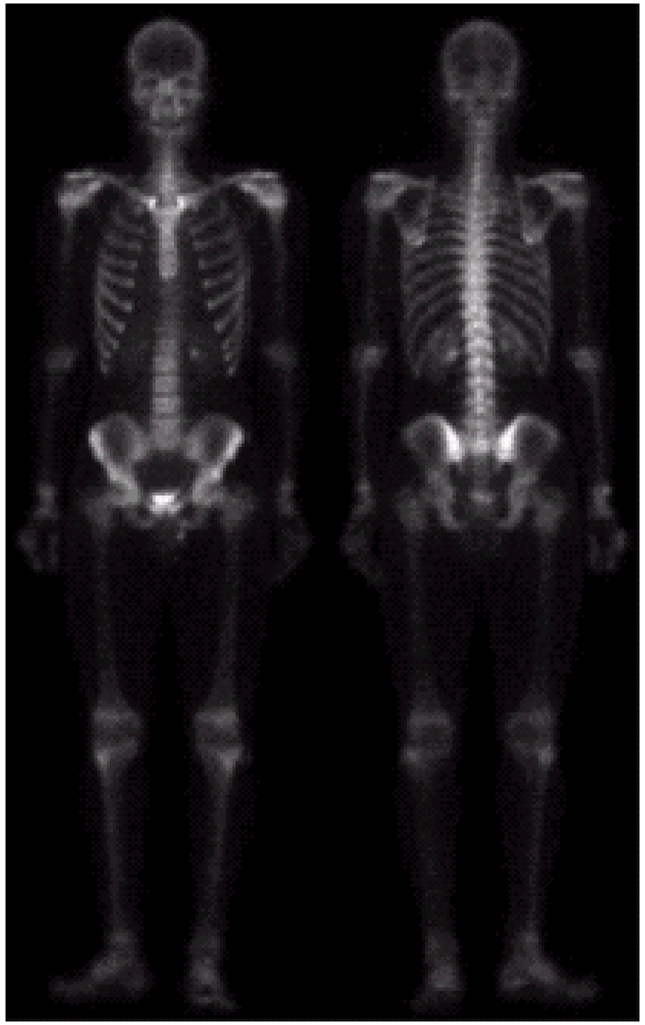
- 1st order derivatives generally produce thicker edges (if thresholded at ramp edges)
- 2nd order derivatives have a stronger response to fine detail e.g. thin lines
- 1st order derivatives have stronger response to grey level step
- 2nd order derivatives produce a double response at step changes in grey level (which helps in detecting zero crossings)

Combining Spatial Enhancement Methods

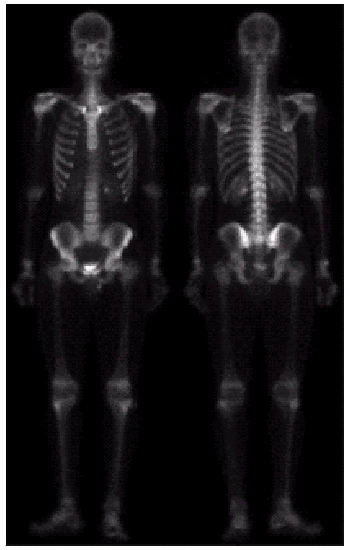
Successful image enhancement is typically not achieved using a single operation

Rather we combine a range of techniques in order to achieve a final result

This example will focus on enhancing the bone scan to the right

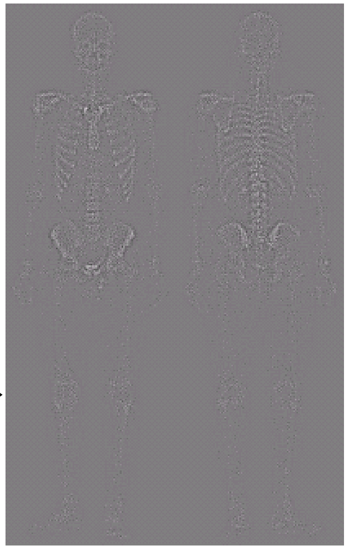


Combining Spatial Enhancement Methods (cont…)



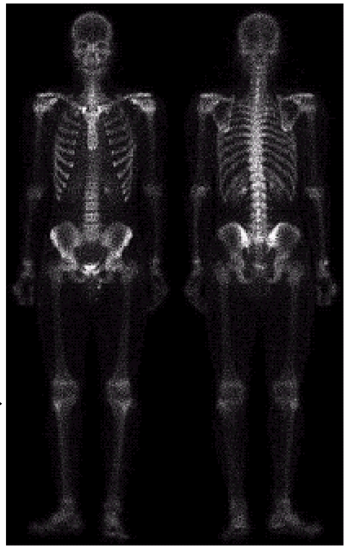
(a)

Laplacian filter of bone scan (a)



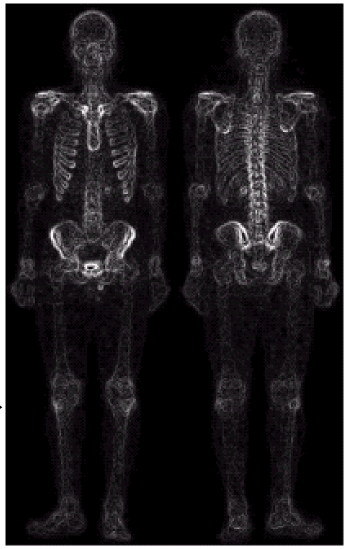
(b)

Sharpened version of bone scan achieved by subtracting (a) and (b)



(c)

Sobel filter of bone scan (a)

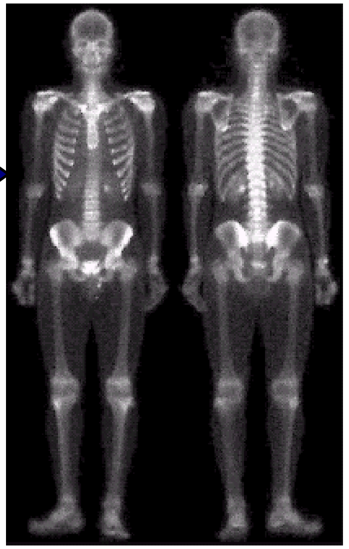


(d)

Combining Spatial Enhancement Methods (cont…)

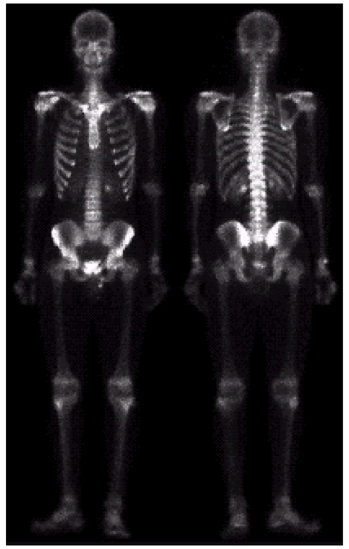
Result of applying a power-law trans. to (g)

(h)



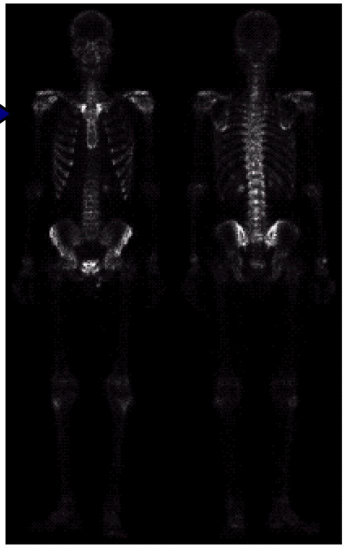
Sharpened image which is sum of (a) and (f)

(g)



The product of (c) and (e) which will be used as a mask

(f)



(e)

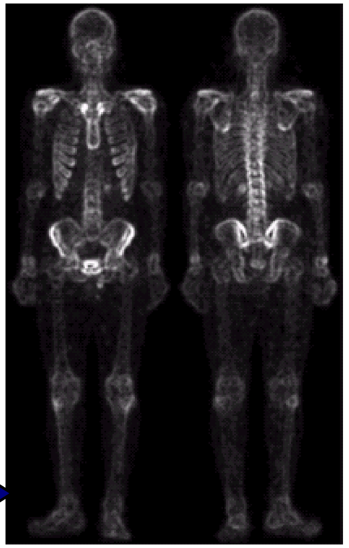
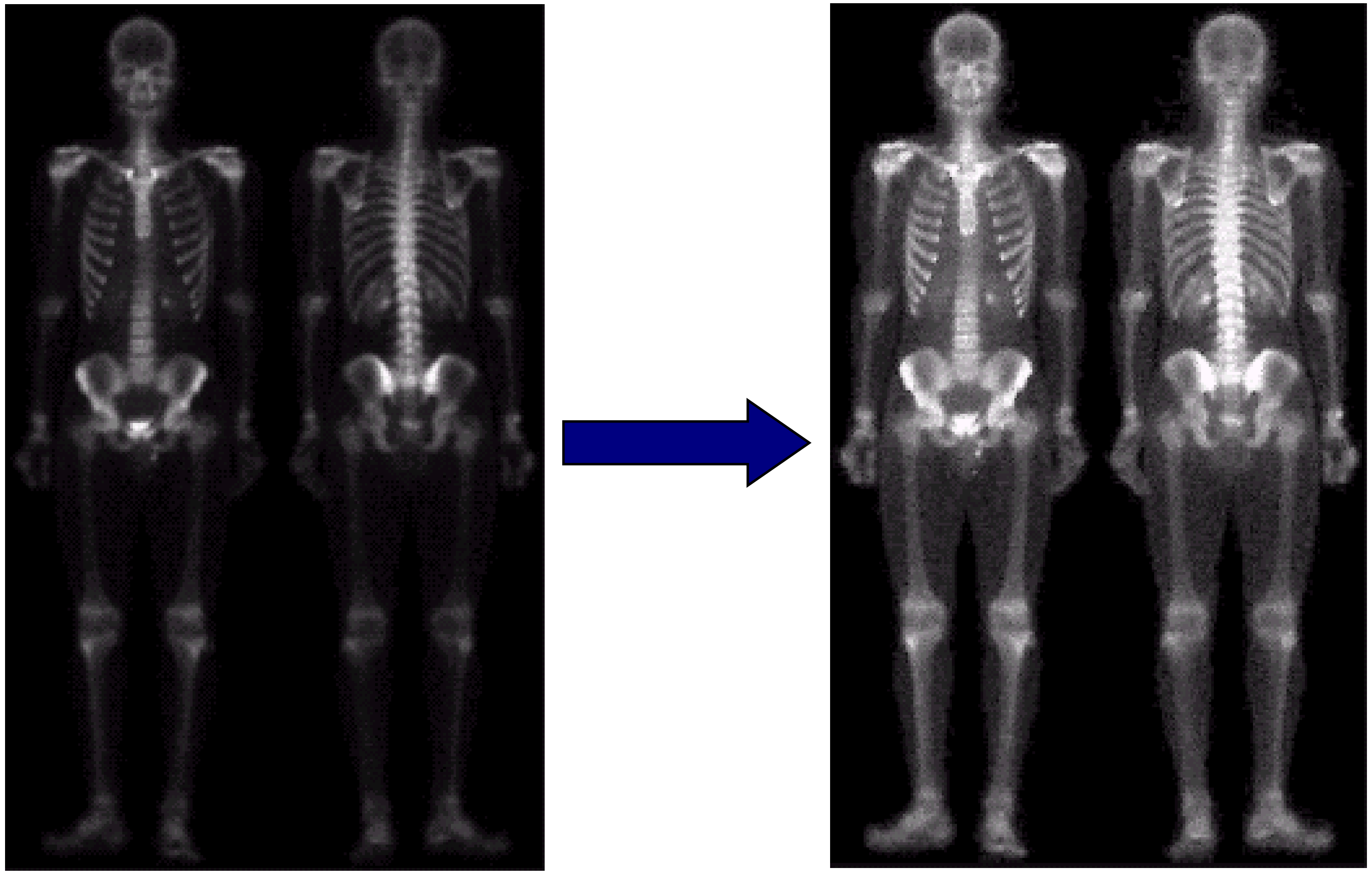


Image (d) smoothed with a 5*5 averaging filter

Combining Spatial Enhancement Methods (cont…)

Compare the original and final images



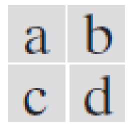
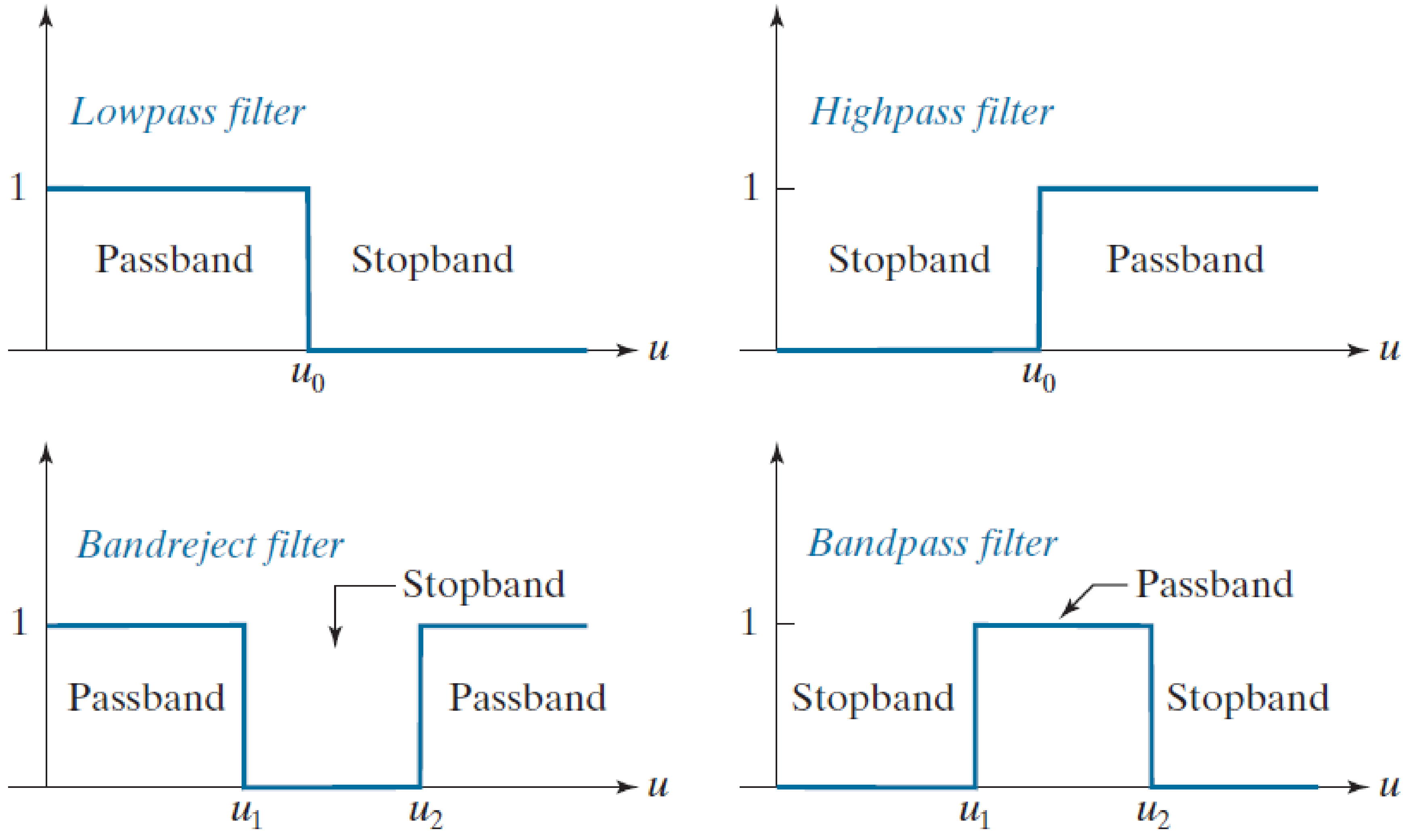
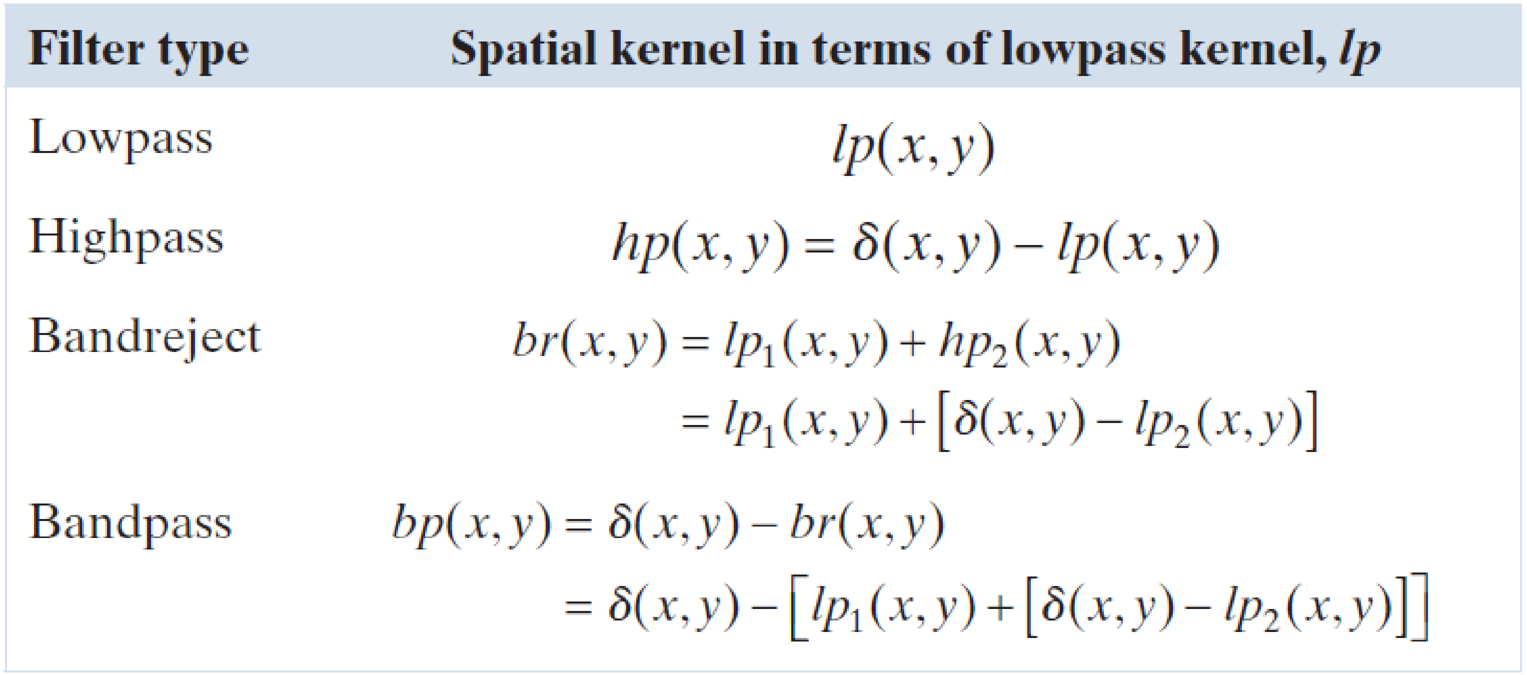


FIGURE 3.52 Transfer functions of ideal 1-D filters in the frequency domain (*u* denotes frequency). (a) Lowpass filter. (b) Highpass filter. (c) Bandreject filter. (d) Bandpass filter. (As before, we show only positive frequencies for simplicity.)



Summary of Spatial Filter Types



Filter type	**Spatial kernel in terms of lowpass kernel, *lp***
Lowpass	$lp(x,y)$
Highpass	$hp(x,y) = \delta(x,y) - lp(x,y)$
Bandreject	$br(x,y) = lp_1(x,y) + hp_2(x,y)$ $= lp_1(x,y) + [\delta(x,y) - lp_2(x,y)]$
Bandpass	$bp(x,y) = \delta(x,y) - br(x,y)$ $= \delta(x,y) - [lp_1(x,y) + [\delta(x,y) - lp_2(x,y)]]$

Summary

In this lecture we have looked at the idea of spatial filtering and in particular:

- Neighbourhood operations
- The filtering process
- Smoothing filters
- Dealing with problems at image edges when using filtering
- Correlation and convolution
- Sharpening filters
- Combining filtering techniques