

Ψηφιακή Επεξεργασία Εικόνας (ΨΕΕ) – ΜΥΕ037

Εαρινό εξάμηνο 2023-2024

Spatial Filtering

Άγγελος Γιώτης

a.giotis@uoi.gr

In this lecture we will look at spatial filtering techniques:

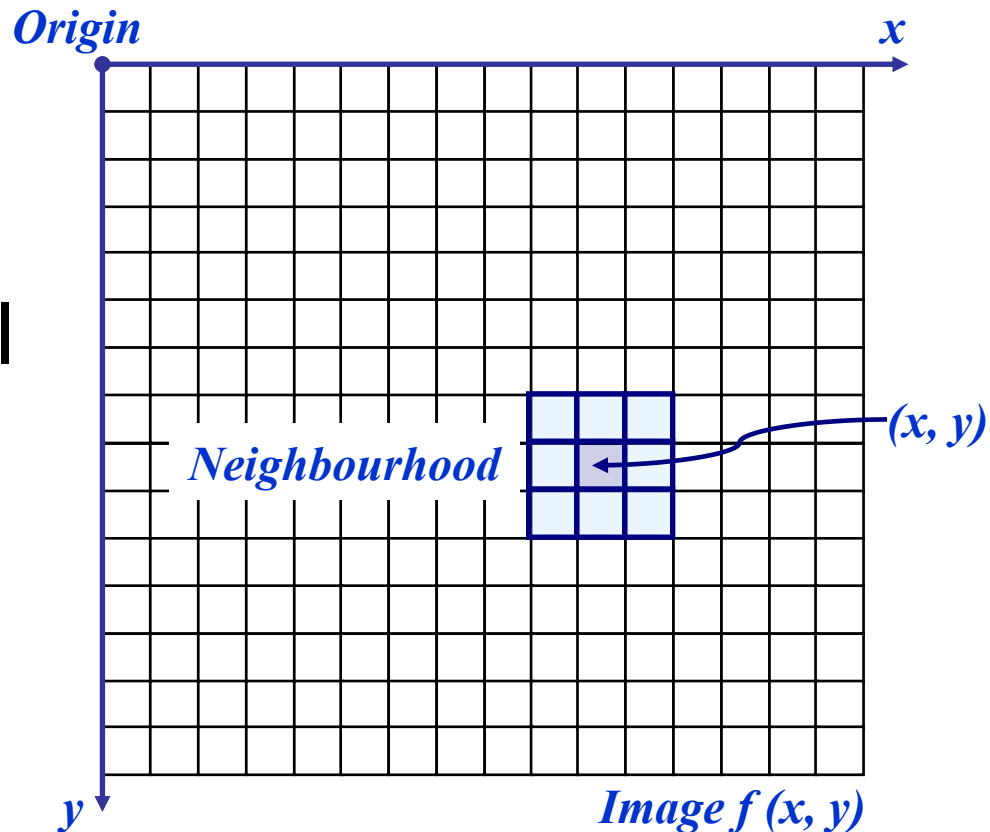
- Neighbourhood operations
- What is spatial filtering?
- Smoothing operations
- What happens at the edges?
- Correlation and convolution
- Sharpening filters
- Combining filtering techniques

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] the median is 15).

Simple Neighbourhood Operations

Some simple neighbourhood operations include:

- **Average/Mean:** Set the pixel value to the mean value over all the pixels in the neighbourhood
- Sometimes the median works better than the average

Simple Neighbourhood Operations - Examples

$\text{Min}(1, 7, 15, 18, 24) = 1$

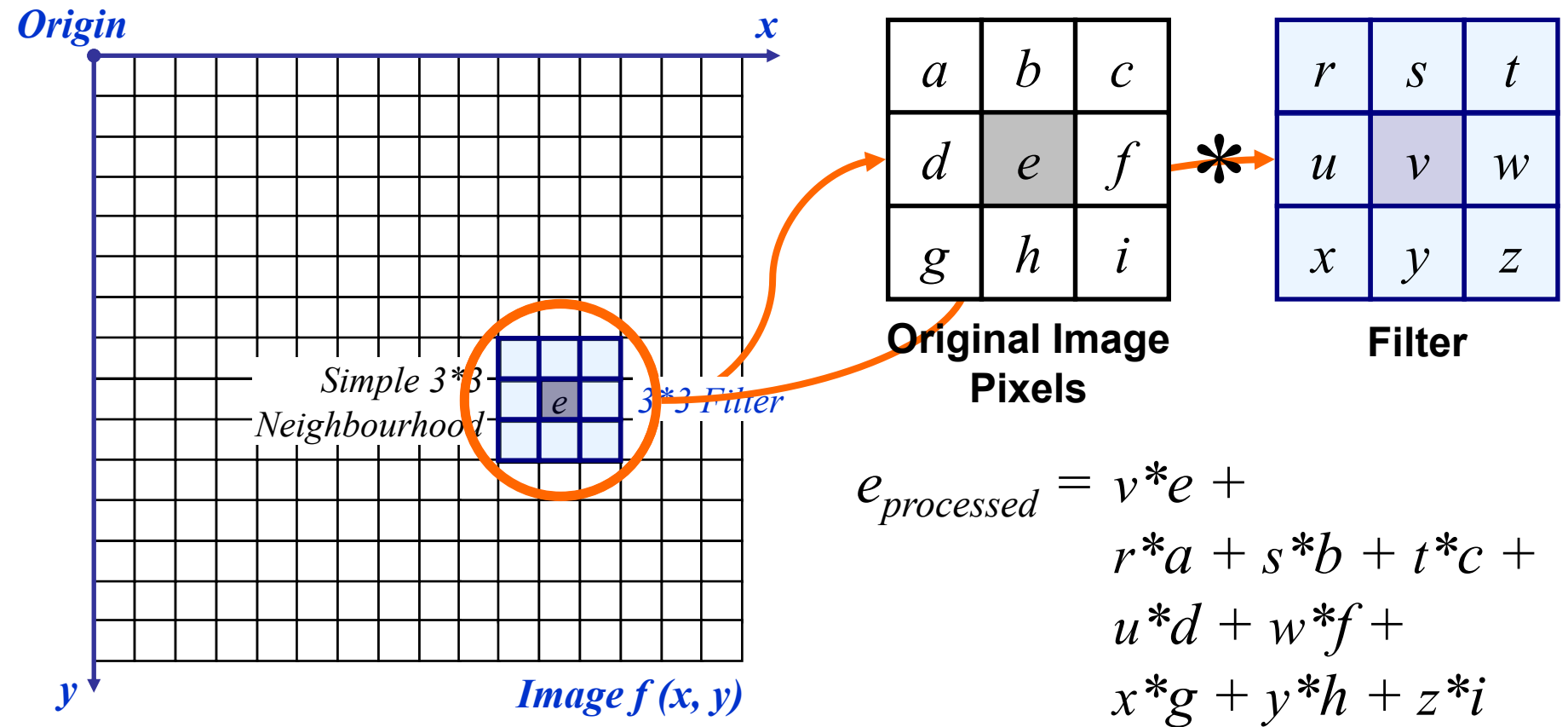
$\text{Max}(1, 7, 15, 18, 24) = 24$

$\text{Mean}(1, 7, 15, 18, 24) = 13$

$\text{Median}(1, 7, 15, 17, 18, 24) = 16$

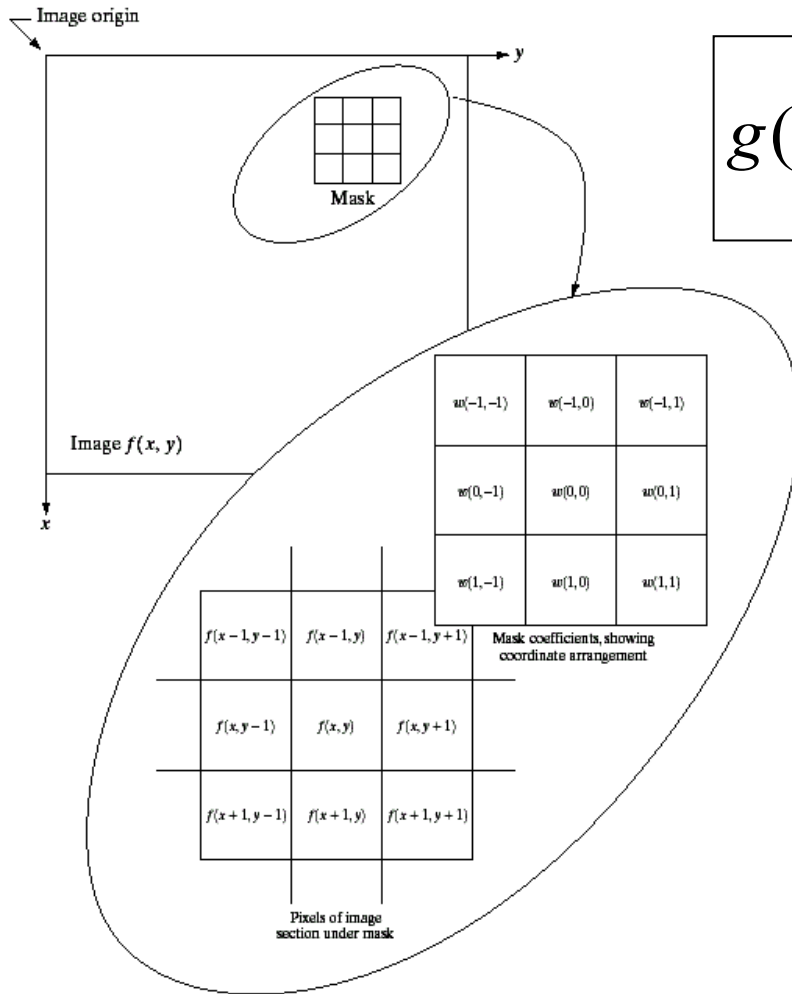
(even case: median = average of the two median values 15 and 17)

The Spatial Filtering Process



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

Spatial Filtering: Equation Form

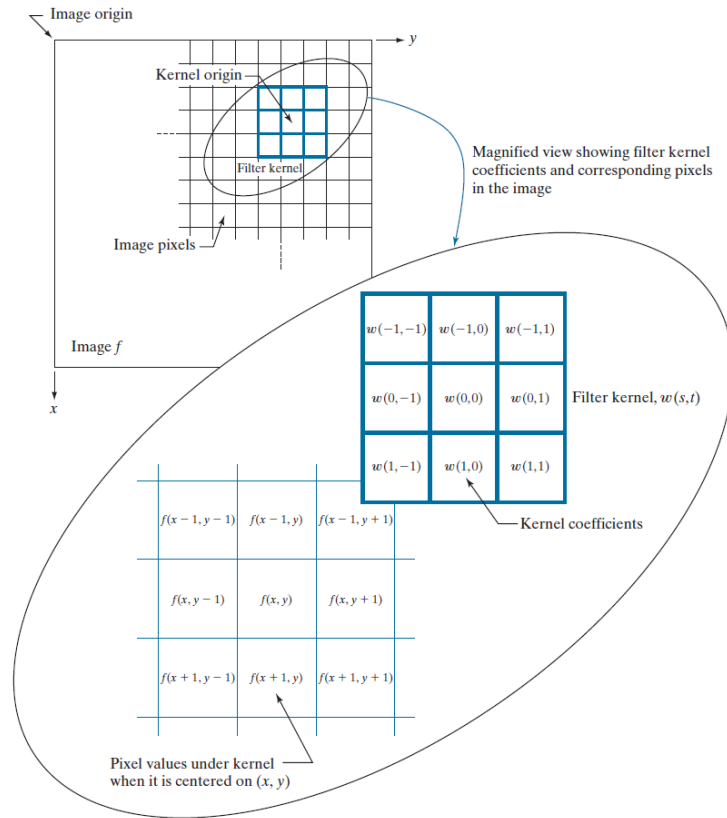


$$g(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x + s, y + t)$$

Filtering can be given in equation form as shown above

Notations are based on the image shown to the left

Spatial Filtering: Equation Form



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

Filtering can be given in equation form as shown above

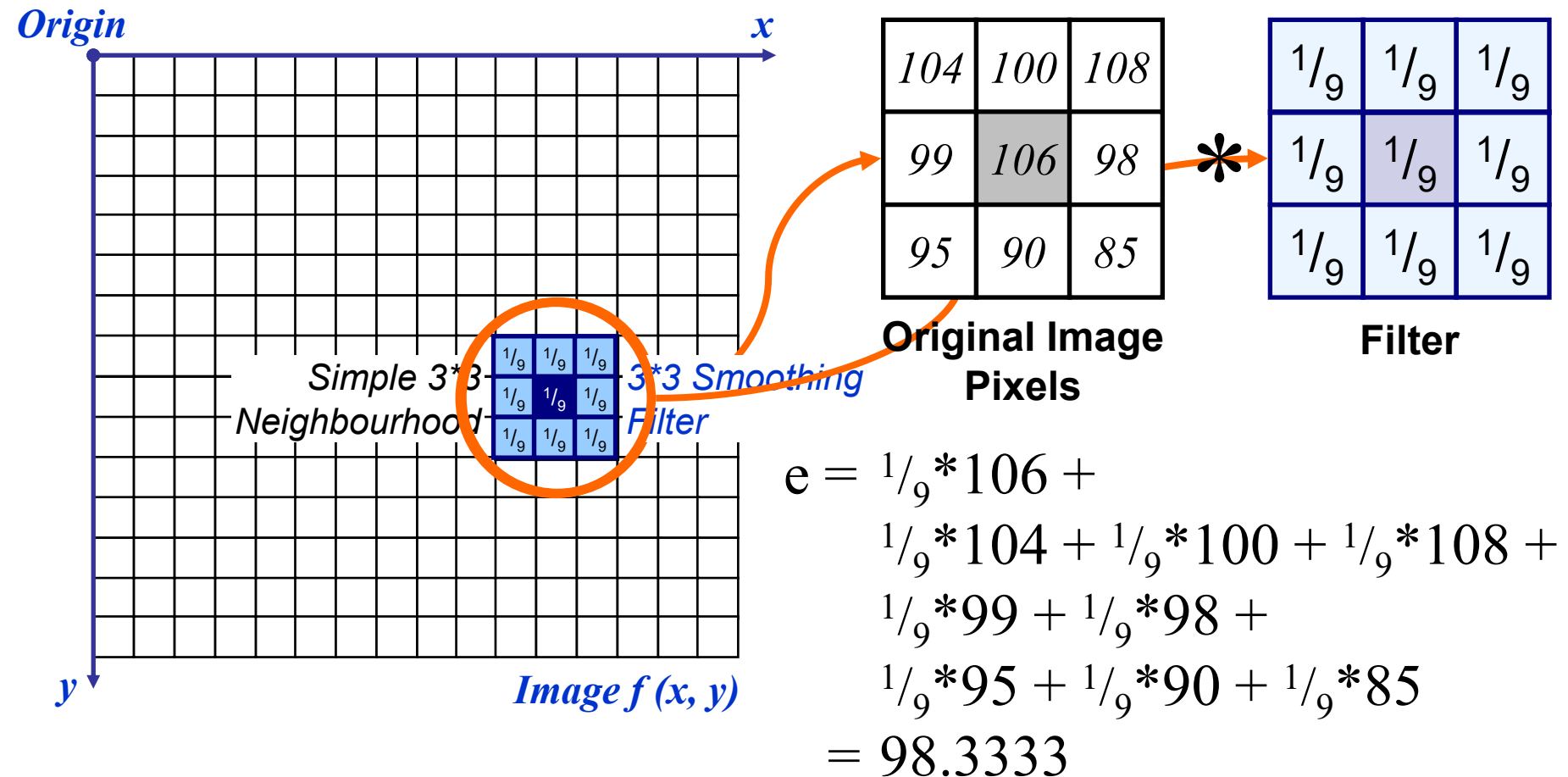
Notations are based on the image shown to the left

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

$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$

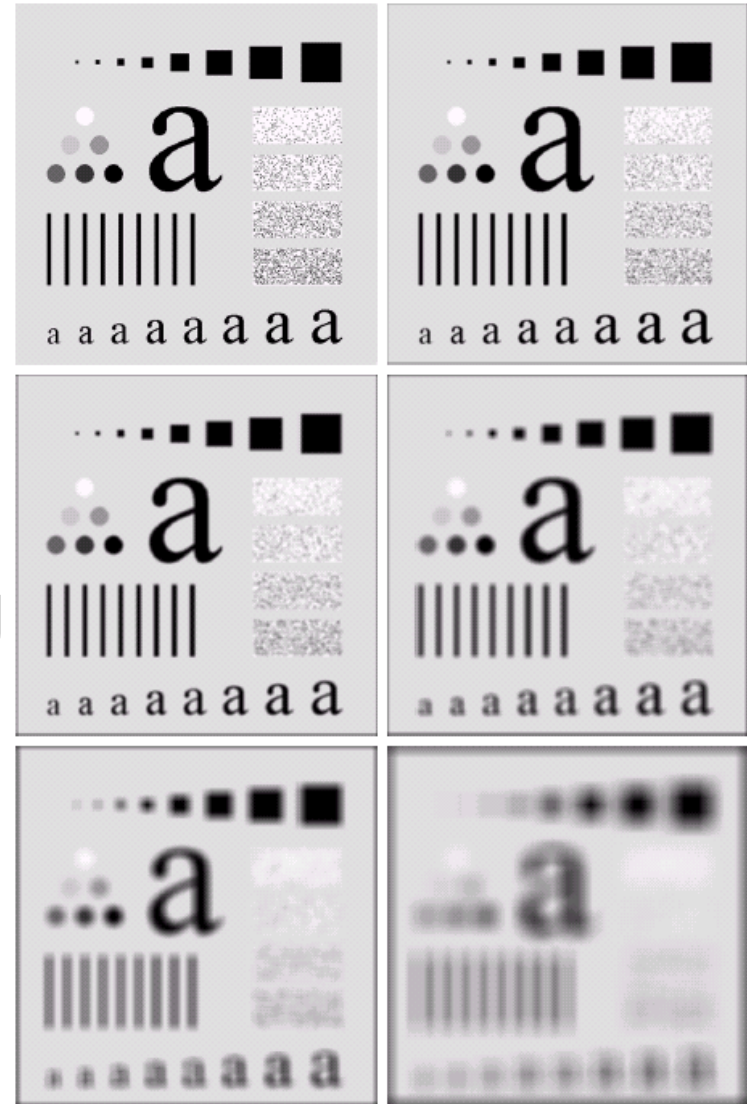
Smoothing Spatial Filtering



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

Image Smoothing Example

- The image at the top left is an original image of size 500*500 pixels
- The subsequent images show the image after filtering with an averaging filter of increasing sizes
 - 3, 5, 9, 15 and 35
- Notice how detail begins to disappear



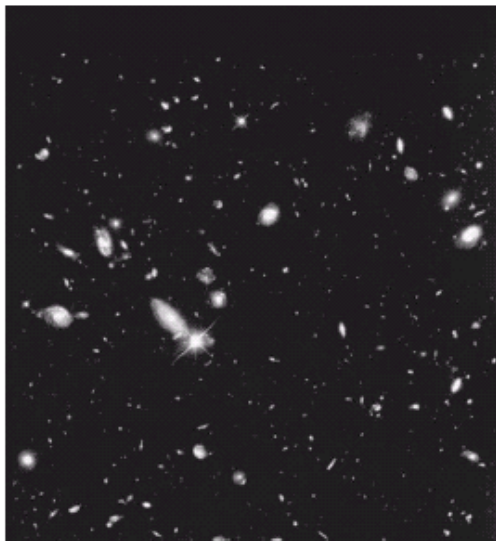
Weighted Smoothing Filters

- 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*

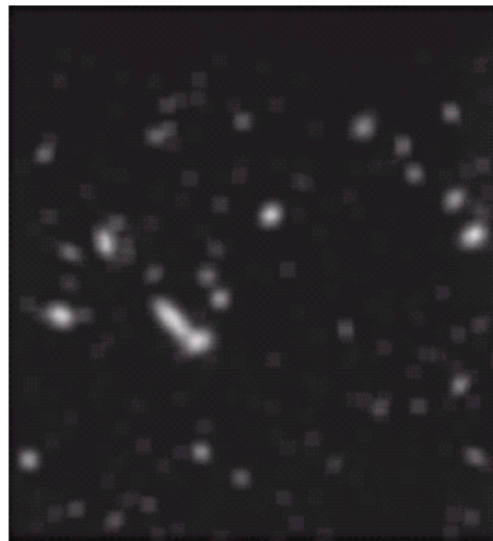
$1/16$	$2/16$	$1/16$
$2/16$	$4/16$	$2/16$
$1/16$	$2/16$	$1/16$

Another Smoothing Example

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



Original Image

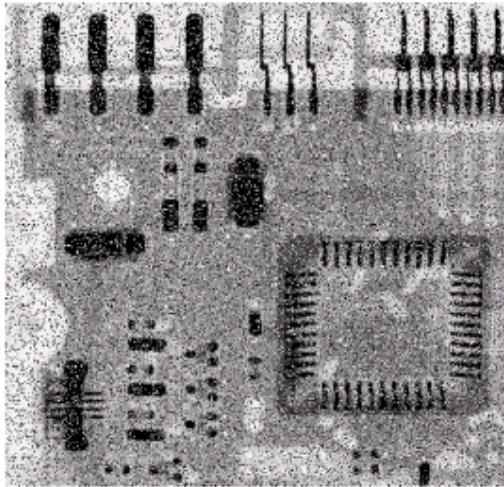


Smoothed Image

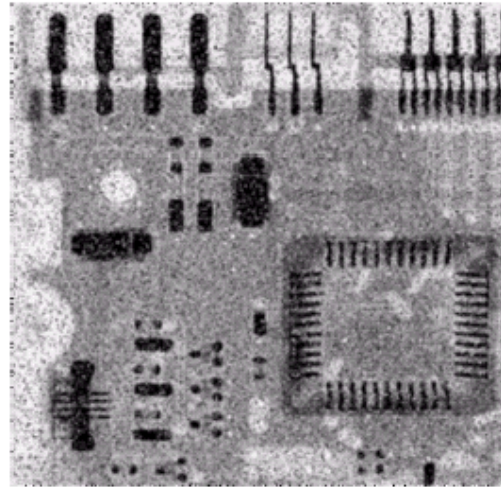


Thresholded Image

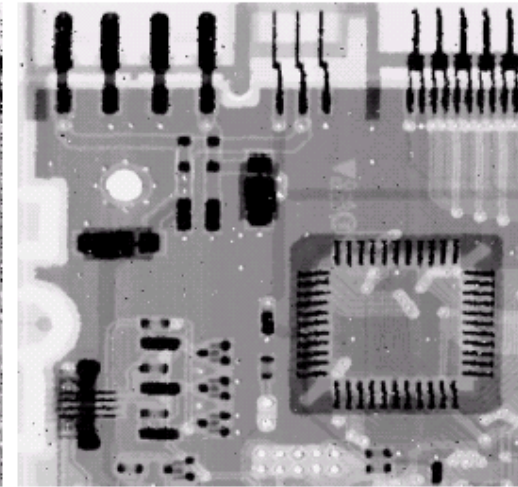
Averaging Filter vs. Median Filter Example



**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

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.

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

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

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

Another criterion is the the sum of absolute differences.

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

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

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)

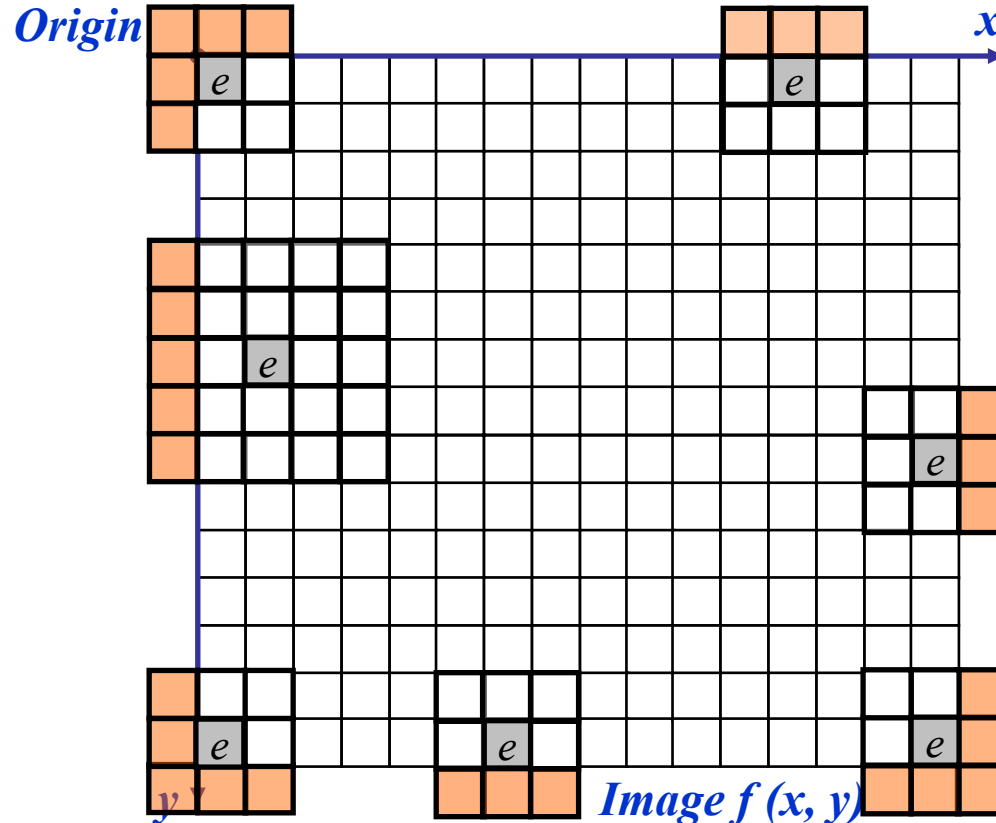
x[n]	-	1.7	1.7	1	1.3	2	2.3	2.3	2.2	-
------	---	-----	-----	---	-----	---	-----	-----	-----	---

↓ ↓ ↓

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:

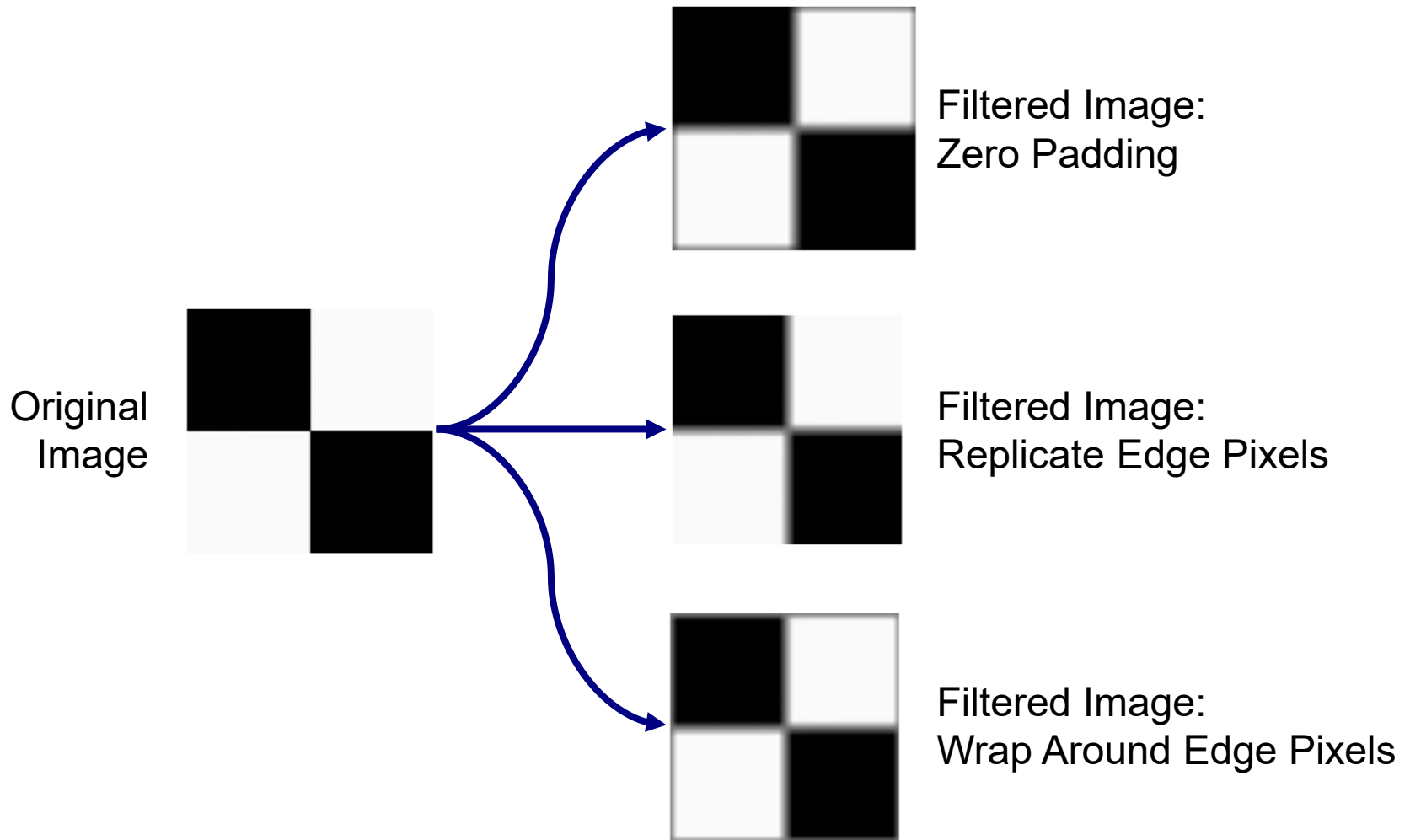
- Omit missing pixels
 - Only works with some filters
 - Can add extra code and slow down processing

Strange Things Happen At The Edges! (cont...)

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

- 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...)



Correlation & Convolution

- The filtering we have been talking about so far is referred to as *correlation* with the filter itself referred to as the *correlation kernel*
- Convolution* is a similar operation, with just one subtle difference

a	b	c
d	e	e
f	g	h

Original Image
Pixels



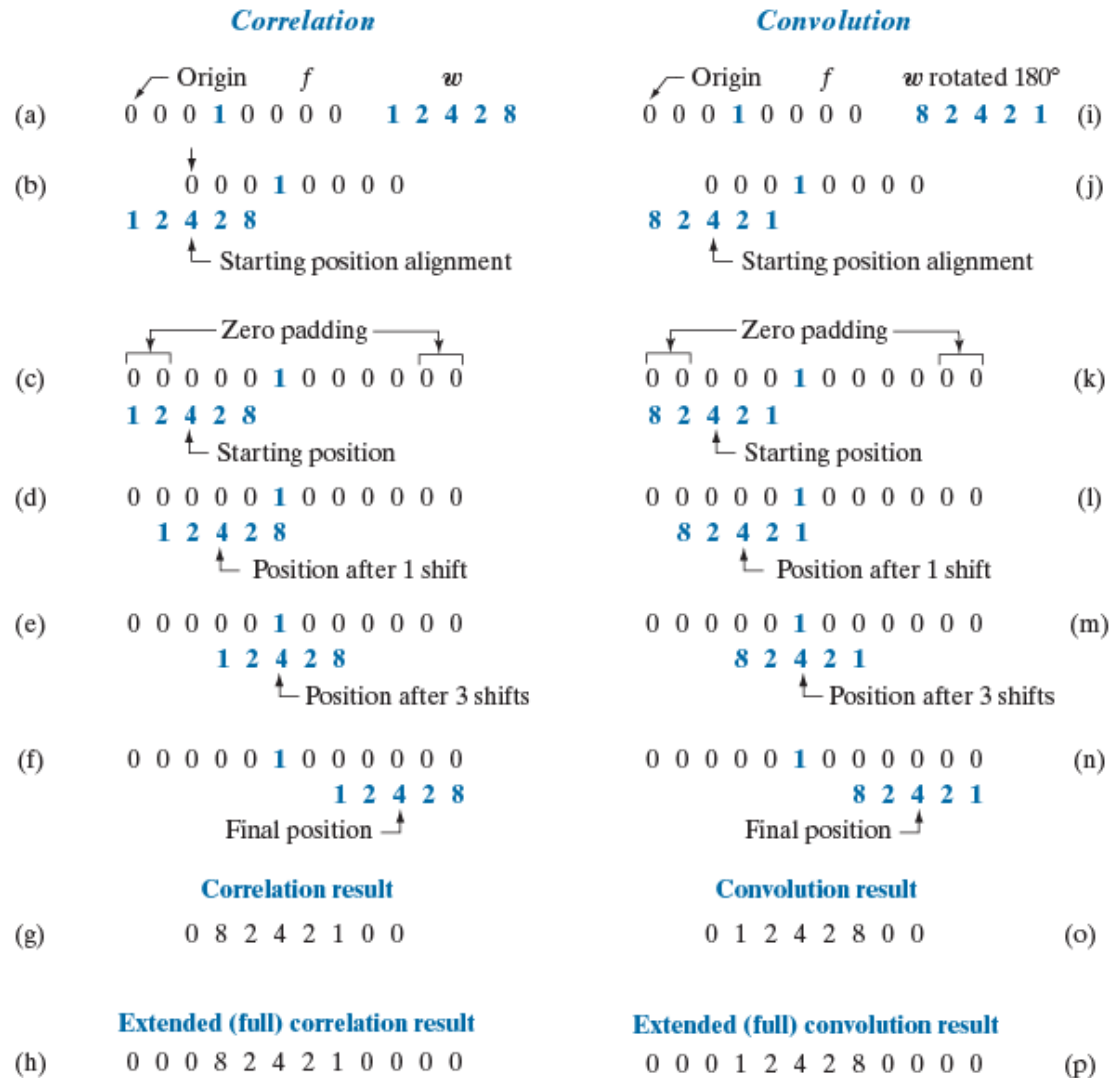
r	s	t
u	v	w
x	y	z

Filter

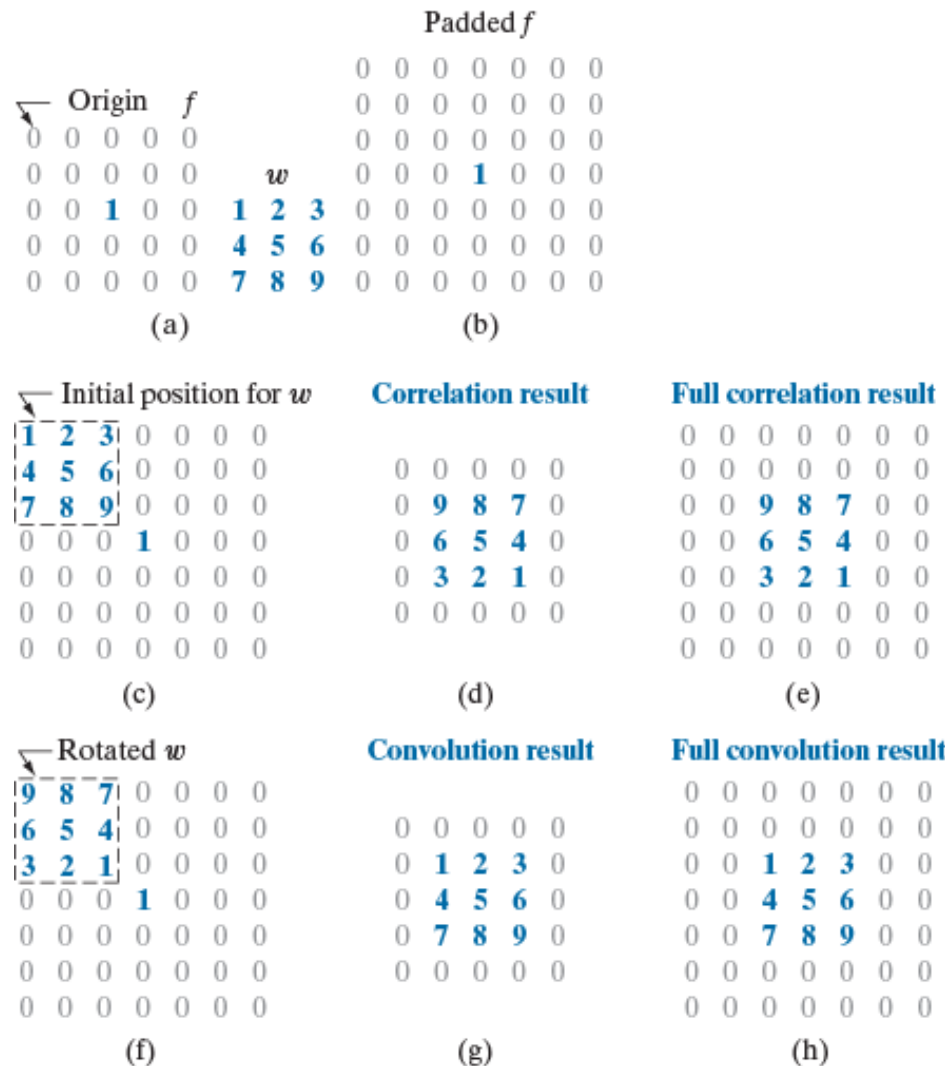
$$e_{processed} = v * e + z * a + y * b + x * c + w * d + u * e + t * f + s * g + r * h$$

- For symmetric filters it makes no difference.

Correlation & Convolution (cont.)



Correlation & Convolution (cont.)



Fundamental properties of convolution and correlation

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

- A 2-D function $G(x, y)$ is considered separable if it can be expressed as the product of two 1-D functions, $G_1(x)$ and $G_2(y)$, such that: $G(x, y) = G_1(x)G_2(y)$.
- For example, the $2 * 3$ kernel

$$w = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

is separable because it can be expressed as the outer product of the vectors

$$\mathbf{c} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \text{and} \quad \mathbf{r} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$\mathbf{c} \mathbf{r}^T = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = w$$

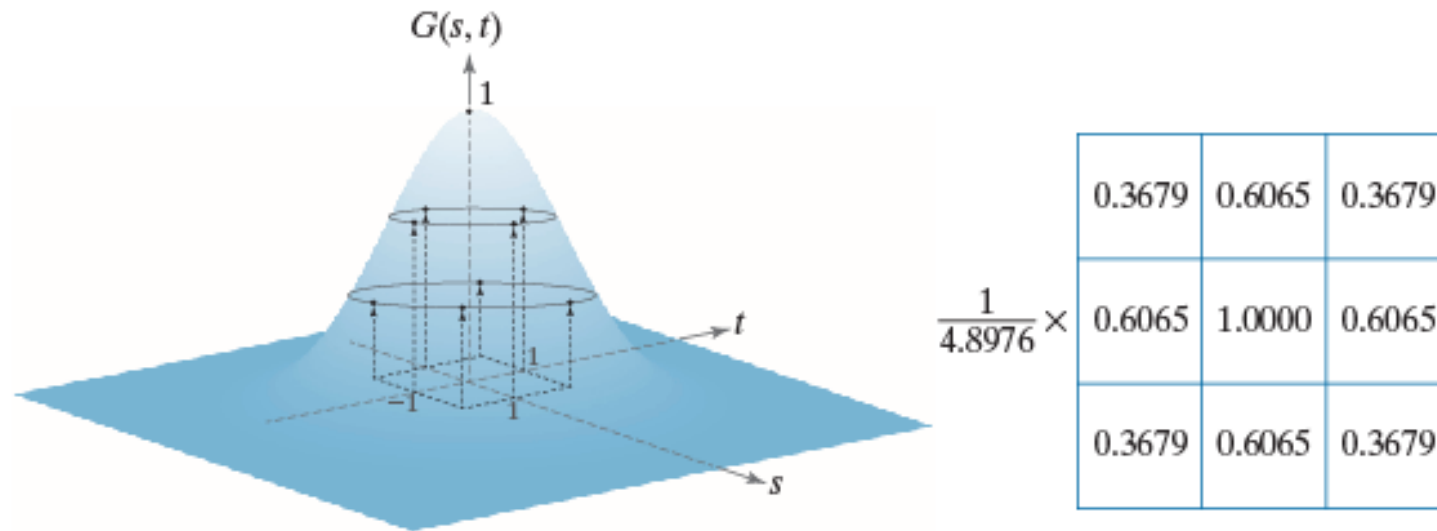
- A separable kernel of size $m \times n$ can be expressed as the outer product of two vectors, \mathbf{v} and \mathbf{w} of size $m \times 1$ and $n \times 1$, respectively: $w = \mathbf{v} \mathbf{w}^T$
- The product of a column vector and a row vector is the same as the 2-D convolution of the vectors.
- Convoluting a separable kernel: $w = w_1 \star w_2$,

with an image is the same as convolving w_1 with f first, and then convolving the result with w_2

- Separable
 - Fast computation
- Isotropic
 - Independent of orientation
- The product of two Gaussians is also a Gaussian
- The convolution of two Gaussians is also a Gaussian

$$w(s, t) = K e^{-\left(\frac{s^2 + t^2}{2\sigma^2}\right)}$$

Gaussian filter (cont.)



A sampling size of $6\sigma \times 6\sigma$ is sufficient

Gaussian filter (cont.)



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.

Gaussian filter (cont.)

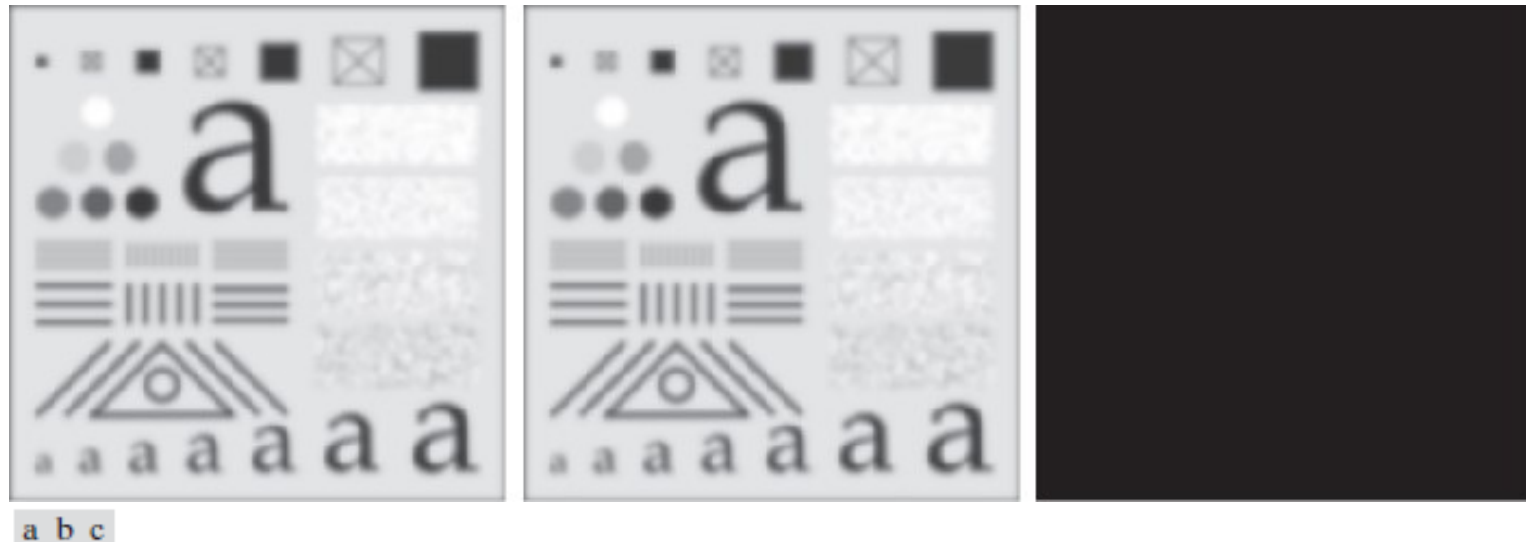


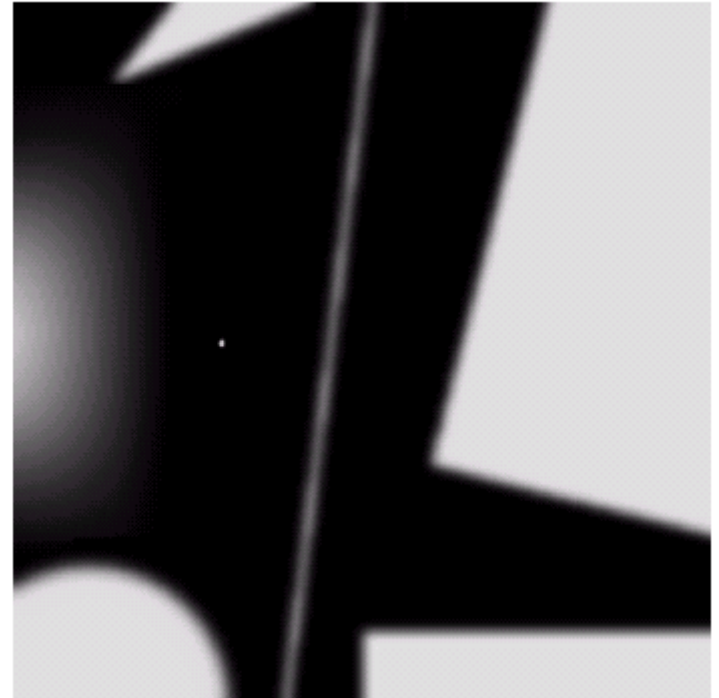
FIGURE 3.37 (a) Result of filtering Fig. 3.36(a) using a Gaussian kernels of size 43×43 , with $\sigma = 7$. (b) Result of using a kernel of 85×85 , with the same value of σ . (c) Difference image.

Sharpening Spatial Filters

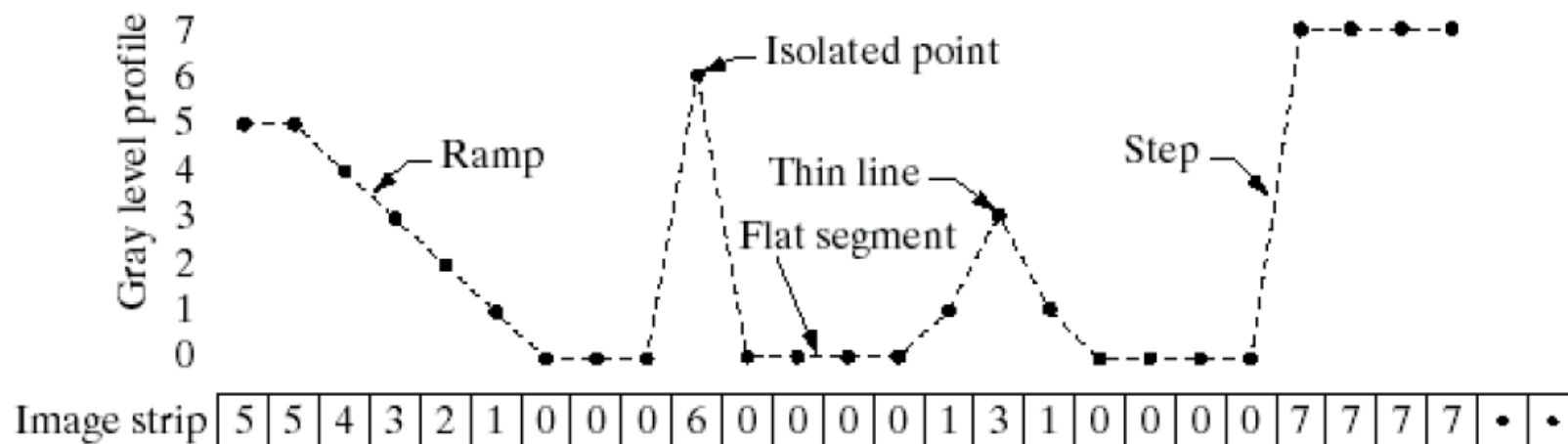
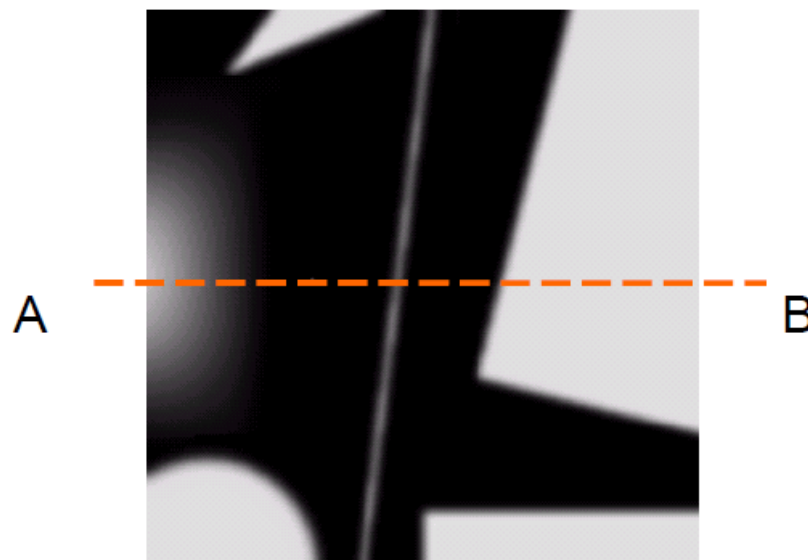
- Previously we have looked at smoothing filters which remove fine detail
- *Sharpening spatial filters* seek to highlight fine detail
 - Remove blurring from images
 - Highlight edges
- Sharpening filters are based on spatial differentiation

Sharpening Spatial Filters

- We want to measure the rate of change.
- We consider an example in 1D.



Sharpening Spatial Filters



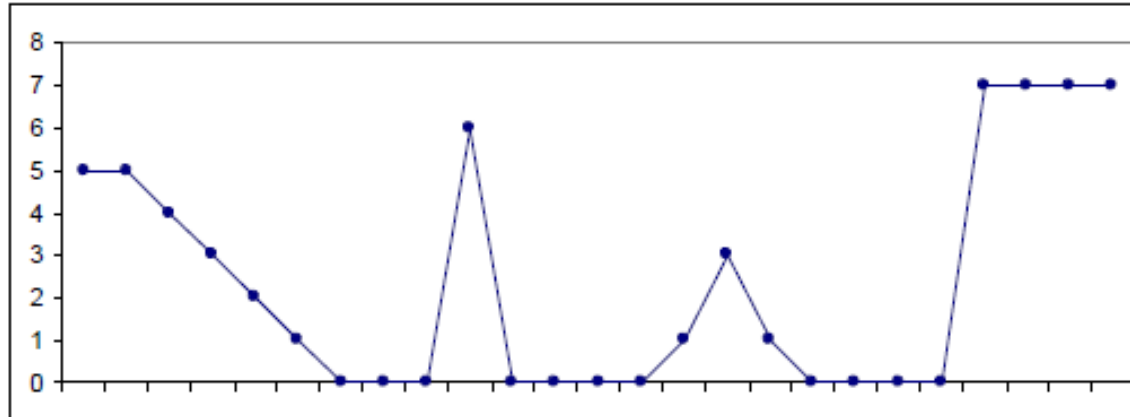
Derivative Filters Requirements

- First derivative filter output
 - Zero at constant intensities
 - Non zero at the onset of a step or ramp
 - Non zero along ramps
- Second derivative filter output
 - Zero at constant intensities
 - Non zero at the onset and end of a step or ramp
 - Zero along ramps of constant slope

- Discrete approximation of the 1st derivative

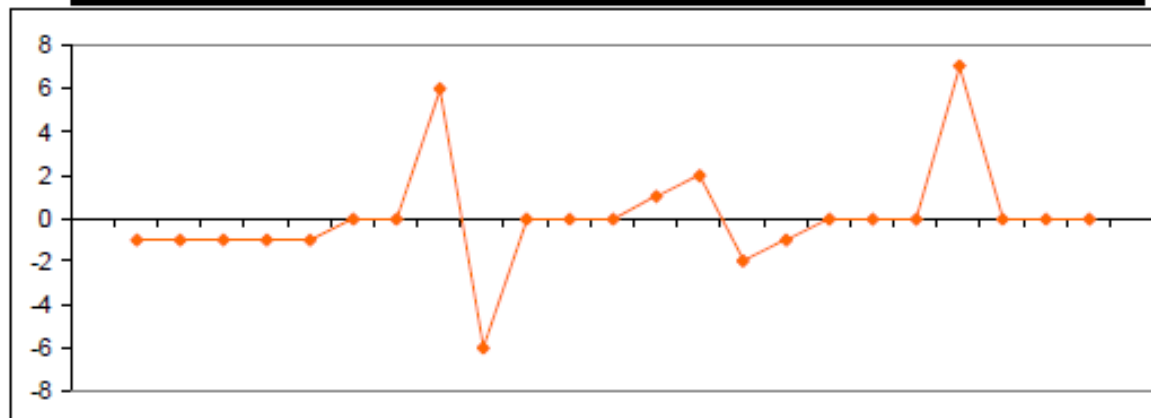
$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

- It is just the difference between subsequent values and measures the rate of change of the function



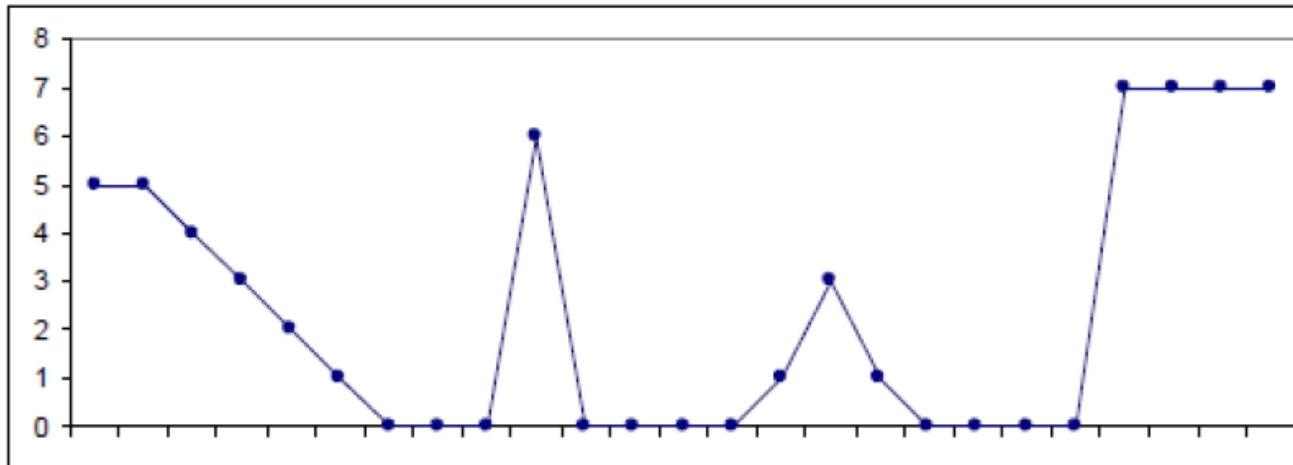
5	5	4	3	2	1	0	0	0	6	0	0	0	0	1	3	1	0	0	0	0	7	7	7	7
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

	0	-1	-1	-1	-1	-1	0	0	6	-6	0	0	0	0	1	2	-2	-1	0	0	0	7	0	0	0
--	---	----	----	----	----	----	---	---	---	----	---	---	---	---	---	---	----	----	---	---	---	---	---	---	---



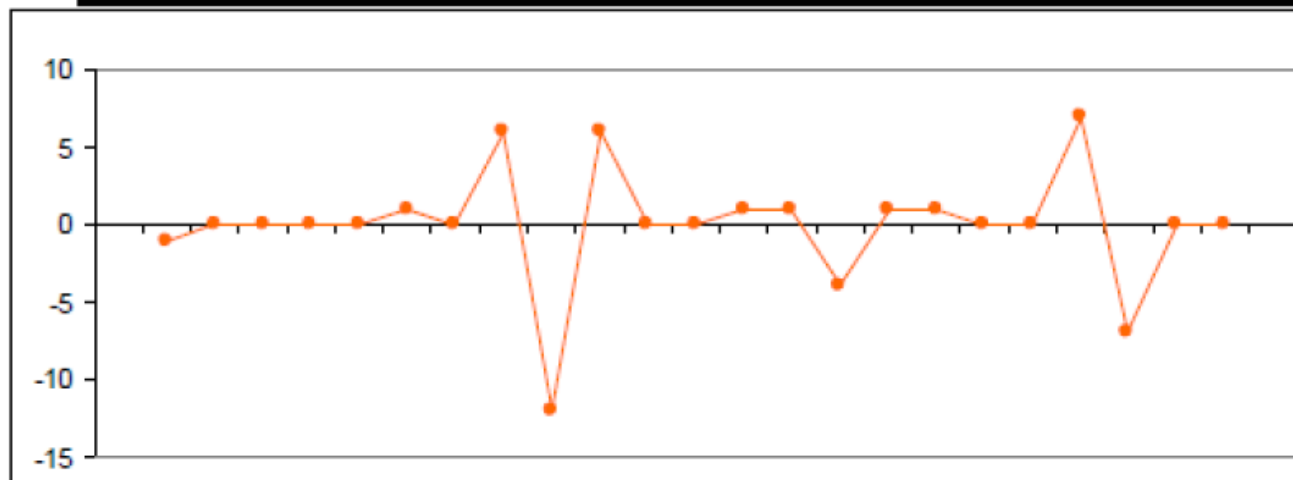
- Discrete approximation of the 2nd derivative:

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

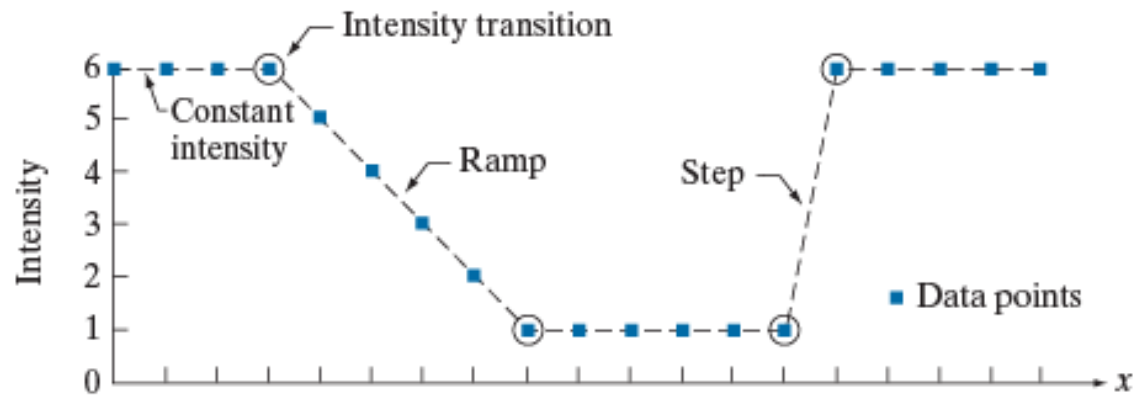


5	5	4	3	2	1	0	0	0	6	0	0	0	0	1	3	1	0	0	0	0	7	7	7	7
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

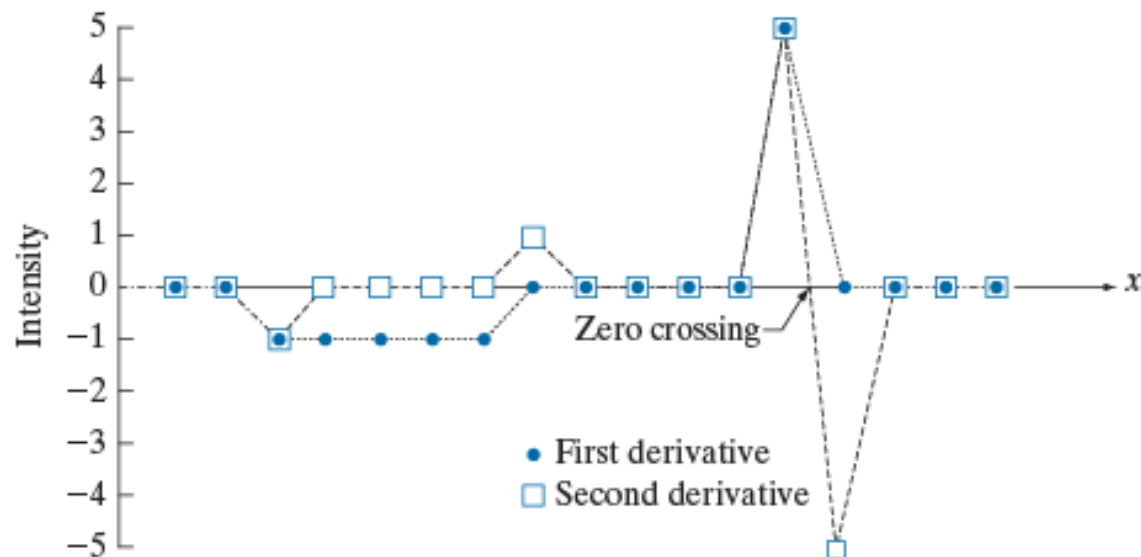
-1	0	0	0	0	1	0	6	-12	6	0	0	1	1	-4	1	1	0	0	7	-7	0	0
----	---	---	---	---	---	---	---	-----	---	---	---	---	---	----	---	---	---	---	---	----	---	---



- Edges in images typically behave like the 'ramps' we examined. The first derivative is constant and produces thick zones at the edges.
- The second derivative gives non-zero response only at the beginning and end of the edge, while being zero in the middle.



Values of scan line	6	6	6	6	5	4	3	2	1	1	1	1	1	1	6	6	6	6	6	$\rightarrow x$
1st derivative	0	0	-1	-1	-1	-1	-1	0	0	0	0	0	0	5	0	0	0	0	0	
2nd derivative	0	0	-1	0	0	0	0	1	0	0	0	0	0	5	-5	0	0	0	0	



Using Second Derivatives For Image Enhancement

- A common sharpening filter is the **Laplacian**
 - Isotropic
 - Rotation invariant: Rotating the image and applying the filter is the same as applying the filter and then rotating the image.
 - In other words, the Laplacian of a rotated image is the rotated Laplacian of the original image.
 - One of the simplest sharpening filters
 - We will look at a digital implementation

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

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

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

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

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

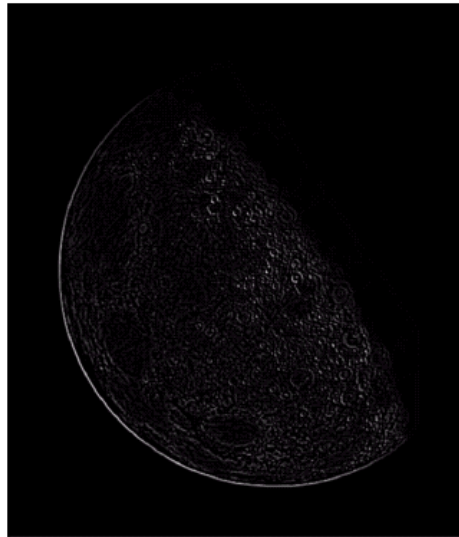
0	1	0
1	-4	1
0	1	0

The Laplacian (cont...)

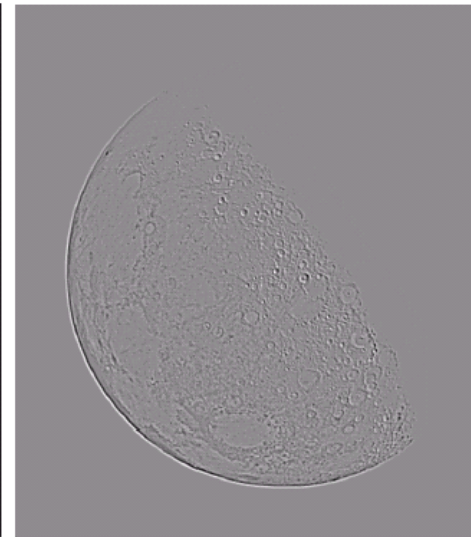
- Applying the Laplacian to an image we get a new image that highlights edges and other discontinuities



Original
Image



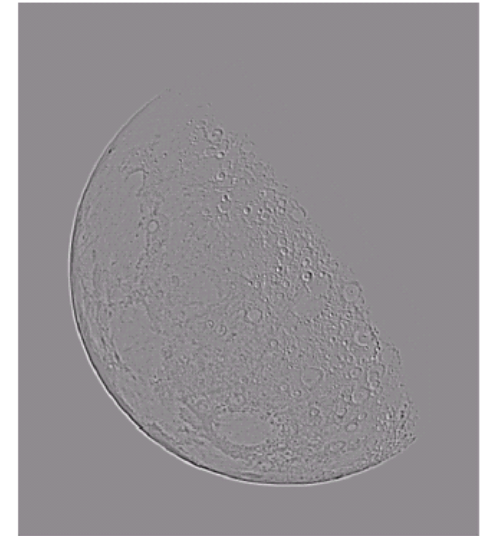
Laplacian
Filtered Image



Laplacian
Filtered Image
Scaled for Display

- The result of a Laplacian filtering is not an enhanced image
- We have to do more work
- Subtract the Laplacian result from the original image to generate our final sharpened enhanced image

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



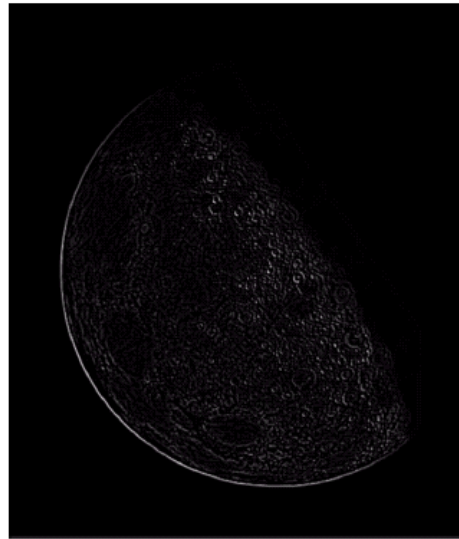
Laplacian
Filtered Image
Scaled for Display

Laplacian Image Enhancement



Original
Image

-



Laplacian
Filtered Image

=



Sharpened
Image

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

Laplacian Image Enhancement



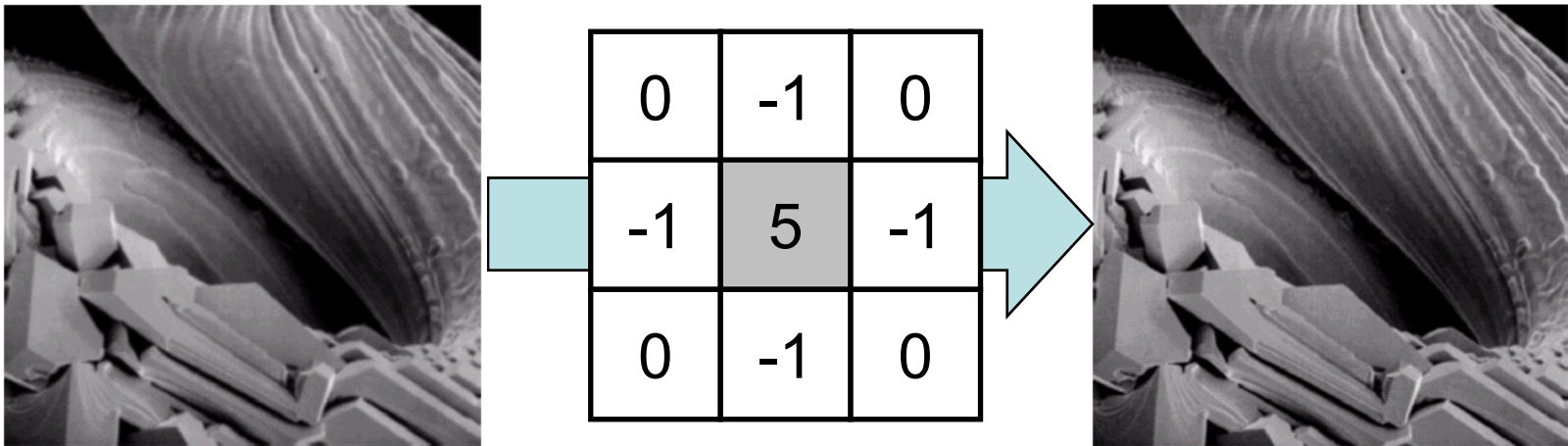
Simplified Image Enhancement

- The entire enhancement can be combined into a single filtering operation:

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

Simplified Image Enhancement (cont...)

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



Variants On The Simple Laplacian

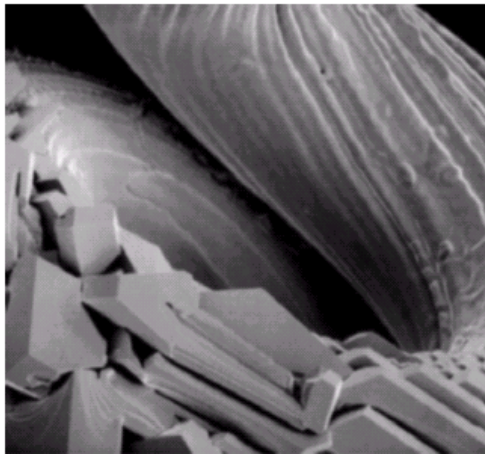
- There are lots of slightly different versions of the Laplacian that can be used:

0	1	0
1	-4	1
0	1	0

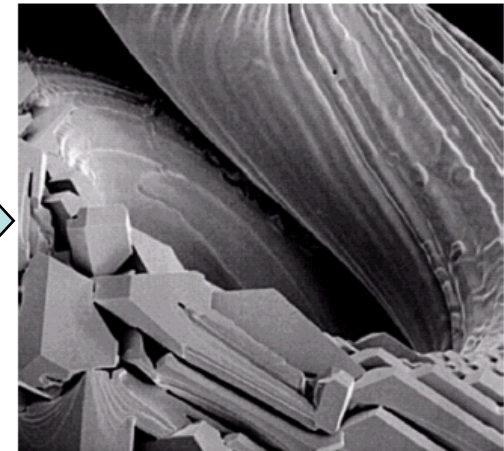
Standard
Laplacian

1	1	1
1	-8	1
1	1	1

Variant of
Laplacian



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



- Used by the printing industry
- Subtracts an unsharped (smooth) image from the original image $f(x,y)$.

–Blur the image

$$b(x,y)=\text{Blur}\{f(x,y)\}$$

–Subtract the blurred image from the original (the result is called the *mask*)

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

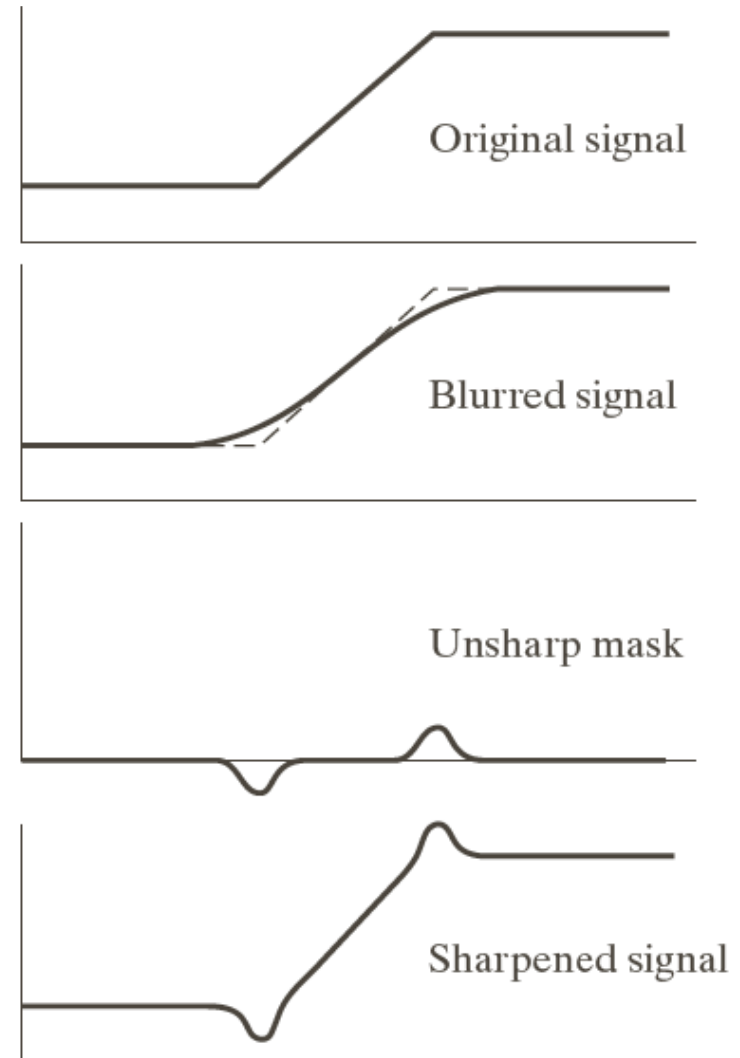
–Add the mask to the original

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

Unsharp masking (cont...)

Sharpening mechanism

If $k > 1$, the process is referred to as **highboost filtering**




Unsharp masking (cont...)

Original image

A dark gray rectangular box containing the text "DIP-XE" in a light gray, sans-serif font.

Blurred image
(Gaussian 5x5, $\sigma=3$)

A dark gray rectangular box containing the text "DIP-XE" in a light gray, sans-serif font. The text is slightly blurred compared to the original image.

Mask

A light gray rectangular box containing the text "DIP-XE" in a light gray, sans-serif font. The text is very faint and blurry.

Unsharp masking ($k=1$)

A dark gray rectangular box containing the text "DIP-XE" in a light gray, sans-serif font. The text is sharper than the blurred image but still has some softness.

Highboost filtering ($k=4.5$)

A dark gray rectangular box containing the text "DIP-XE" in a light gray, sans-serif font. The text is very sharp and clear, with high contrast.

Using First Derivatives For Image Enhancement

$$\nabla f = \begin{bmatrix} G_x & G_y \end{bmatrix}^T = \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \end{bmatrix}^T$$

- Although the derivatives are linear operators, the gradient magnitude is not.
- Also, the partial derivatives are not rotation invariant (isotropic).
- The magnitude of the gradient vector is isotropic.

$$||\nabla f|| = \sqrt{G_x^2 + G_y^2}$$

Using First Derivatives For Image Enhancement (cont...)

- In some applications it is more computationally efficient to approximate:

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

- This expression preserves relative changes in intensity but it is not isotropic.
- Isotropy is preserved only for a limited number of rotational increments which depend on the filter masks (e.g. 90 deg.).

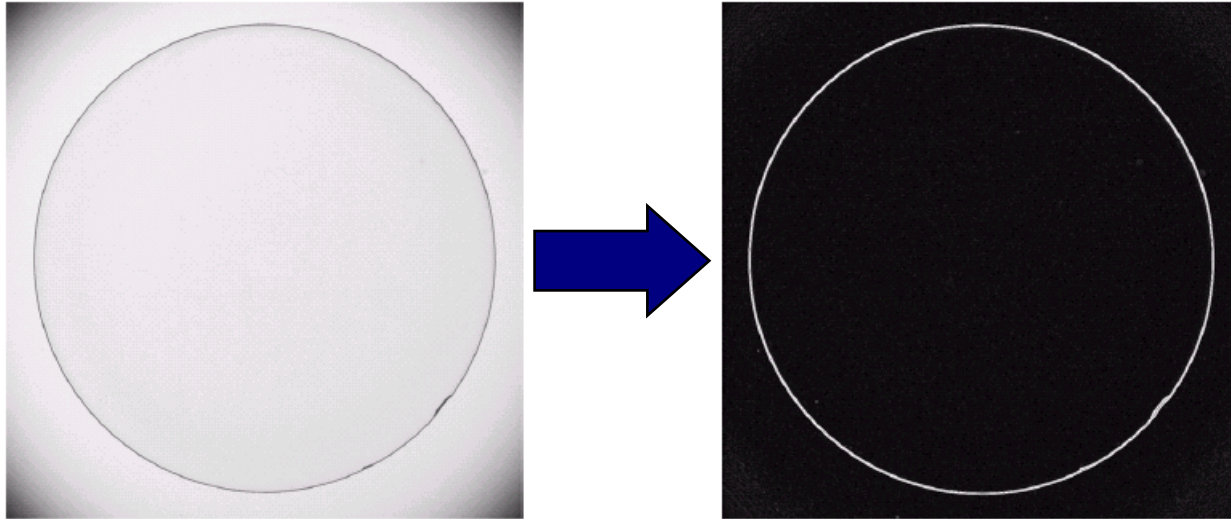
- **Sobel operators** introduce the idea of differentiating by giving more importance to the center point:

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

- Note that the coefficients sum to 0 to give a 0 response at areas of constant intensity.

Sobel operator Example



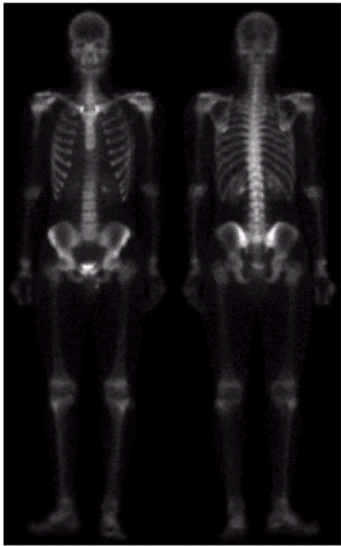
- Sobel gradient aids to eliminate constant or slowly varying shades of gray and assist automatic inspection.
- It also enhances small discontinuities in a flat gray field.
- General comments
 - 1st derivatives tend to produce more thick edges
 - 2nd order derivatives have better response to detail (thin edge)
 - 2nd order derivatives produce double response to edges.

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...)



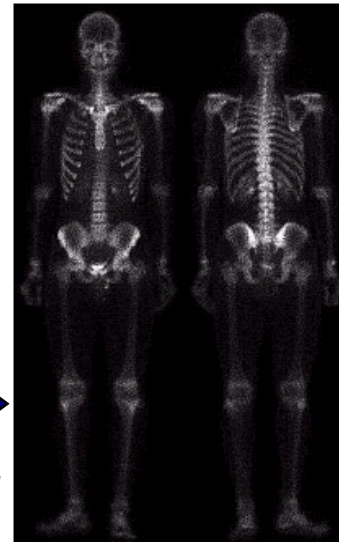
(a)

Laplacian filter of
bone scan (a)



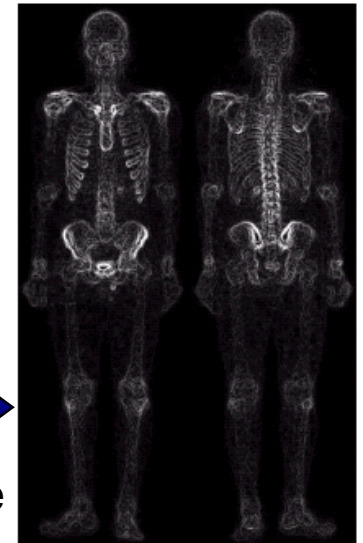
(b)

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



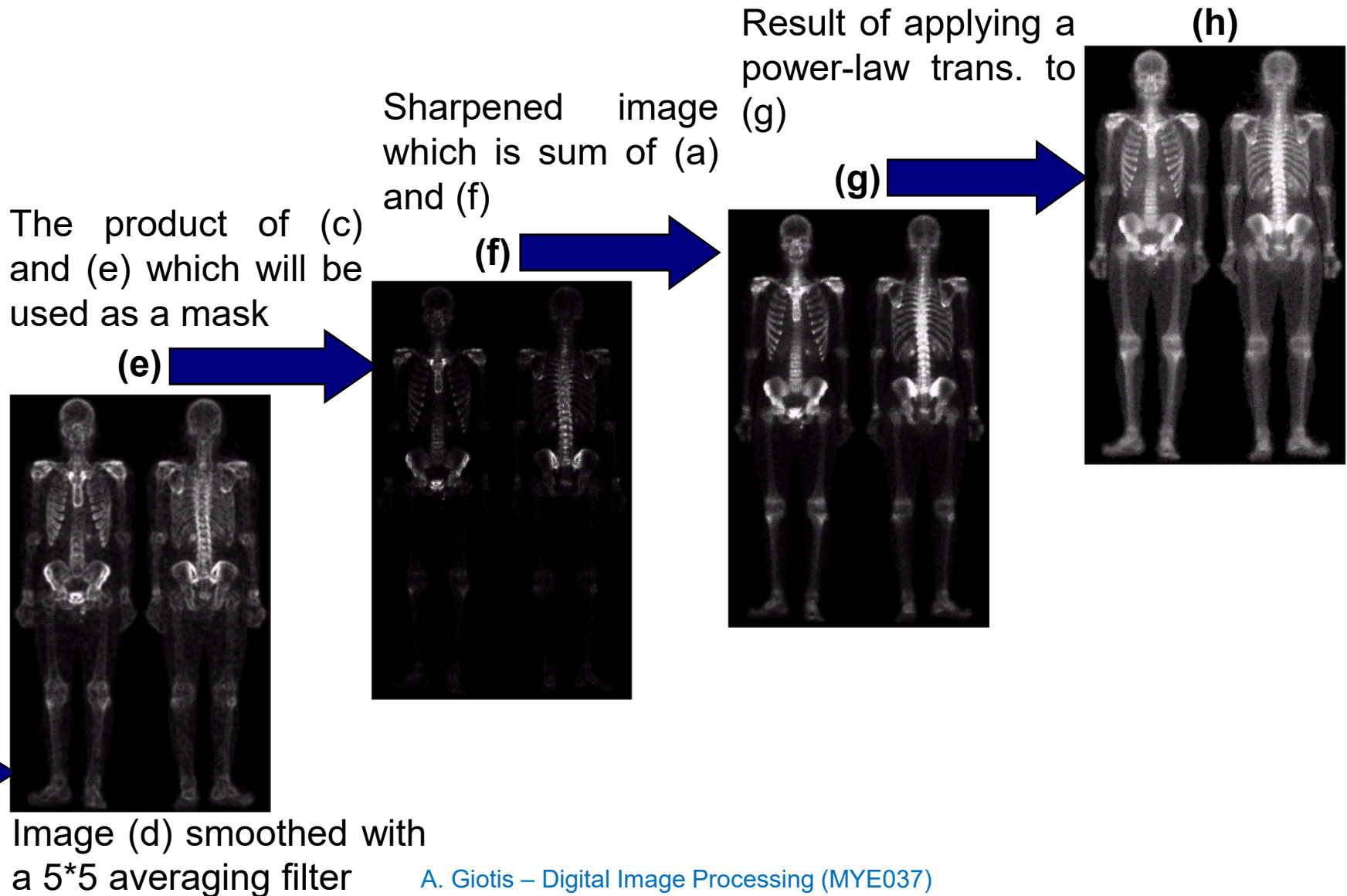
(c)

Sobel filter of bone
scan (a)



(d)

Combining Spatial Enhancement Methods (cont...)



Combining Spatial Enhancement Methods (cont...)

Compare the original and final images



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