BBM413 Fundamentals of Image Processing

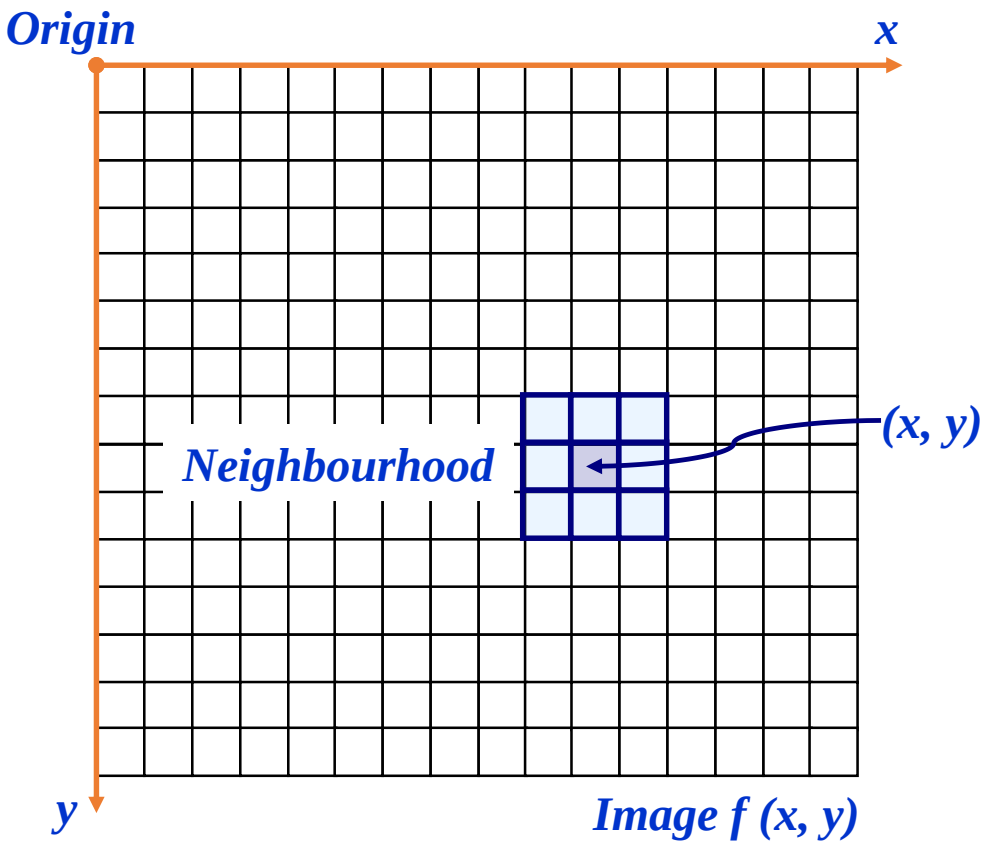
Spatial Filtering

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
    - 1st derivative filters
    - 2nd derivative filters
  - Combining filtering techniques

Neighbourhood Operations

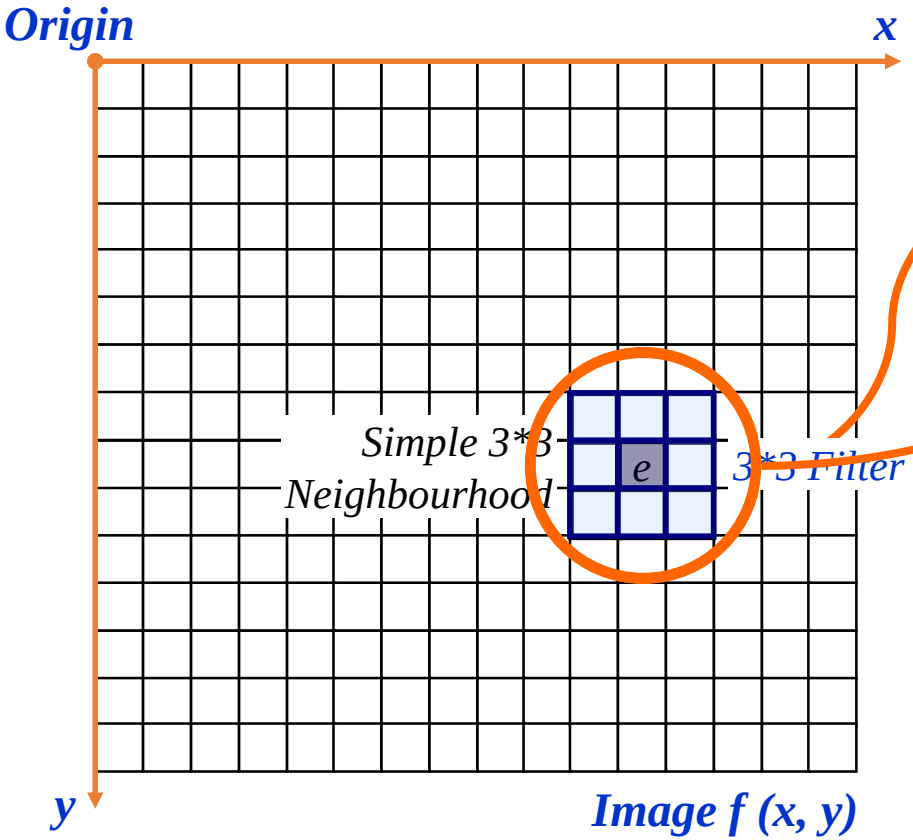
- Neighbourhood operations simply operate on a larger neighbourhood of pixels than point operations
- Neighbourhoods are mostly a square around a central pixel
- Any size rectangle and any shape filter are possible



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

The Spatial Filtering Process



a	b	c
d	e	f
g	h	i

Original Image Pixels

*

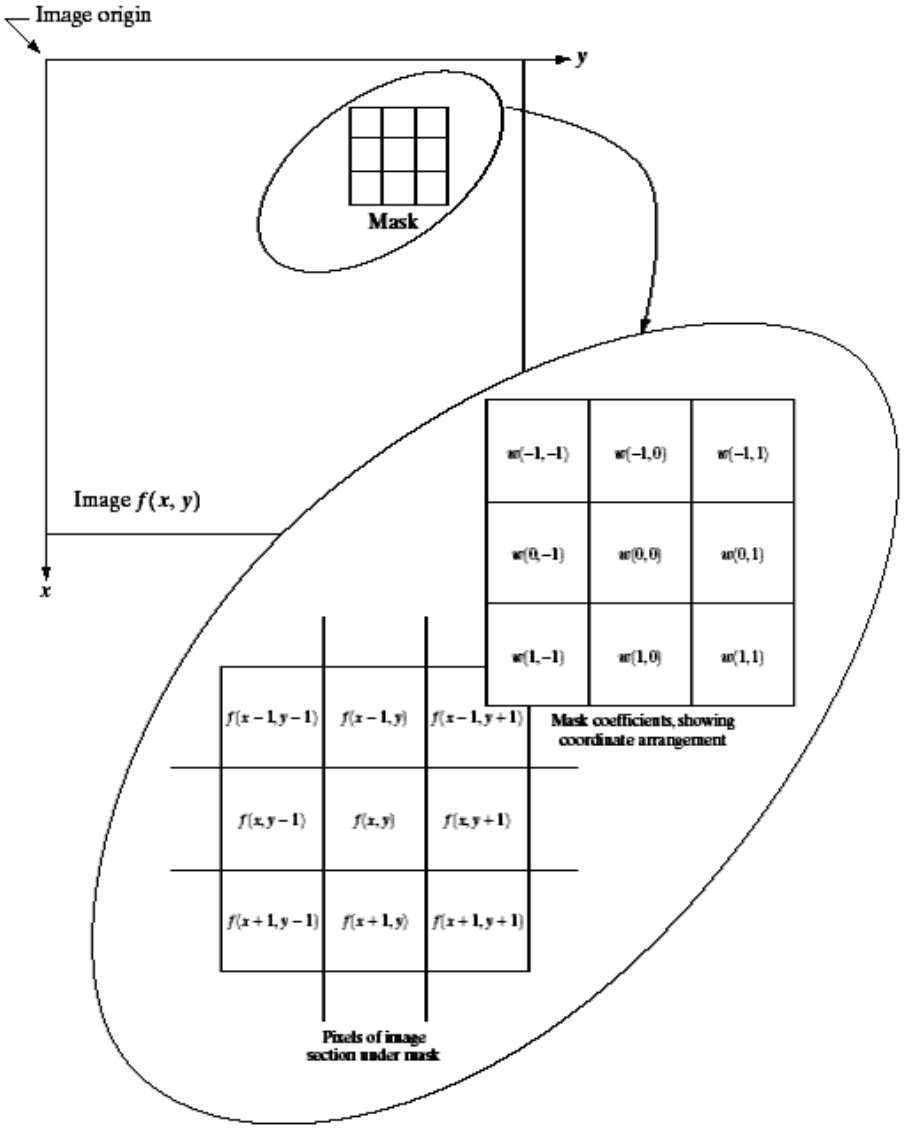
r	s	t
u	v	w
x	y	z

Filter

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

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

Spatial Filtering: Equation Form



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

Filtering can be given in equation form as shown above

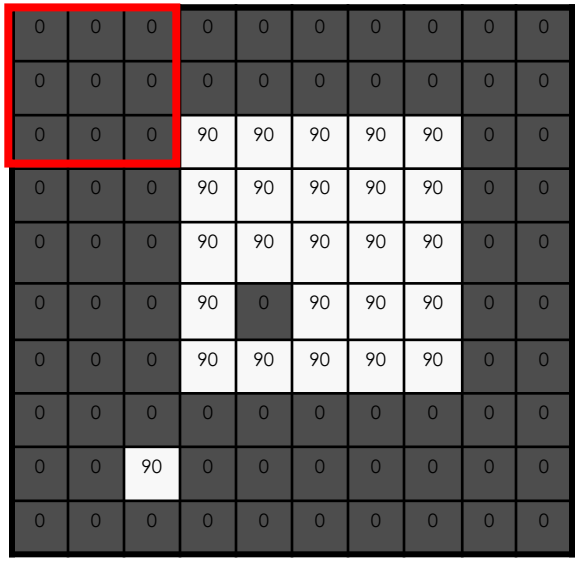
Notations are based on the image shown to the left

Image filtering

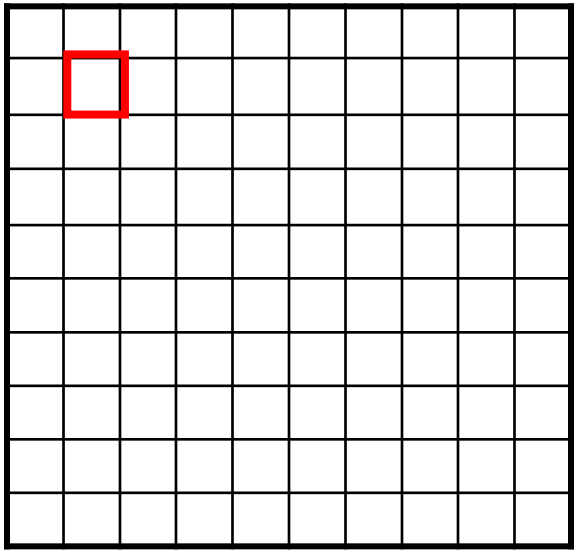
$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$



$h[.,.]$



$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

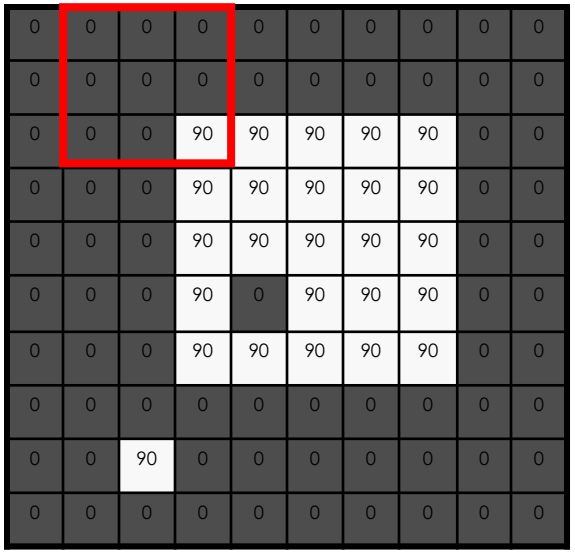
Credit: S. Seitz

Image filtering

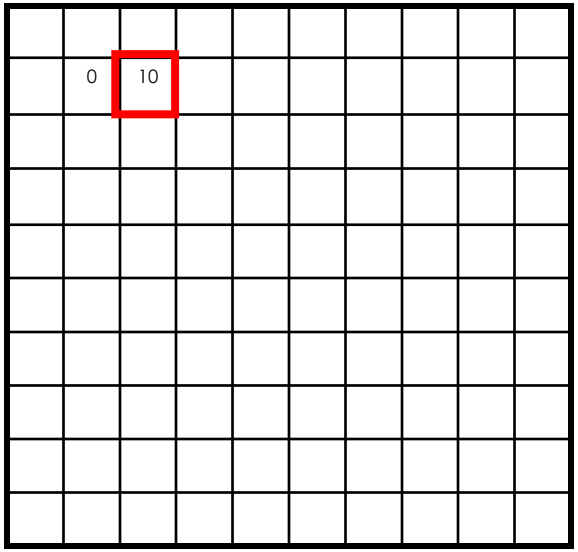
$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$



$h[.,.]$



$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

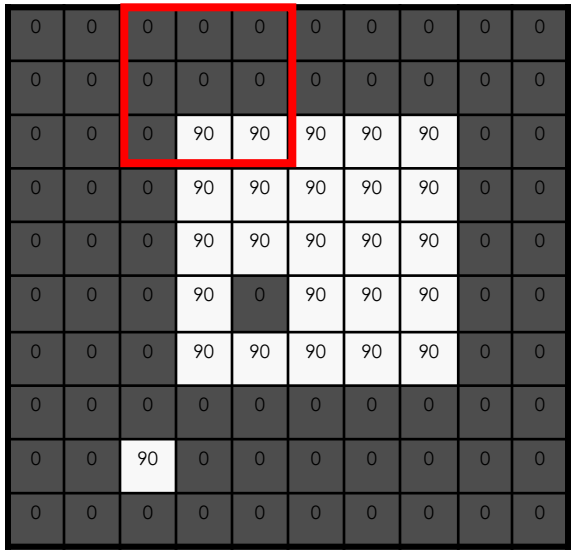
Credit: S. Seitz

Image filtering

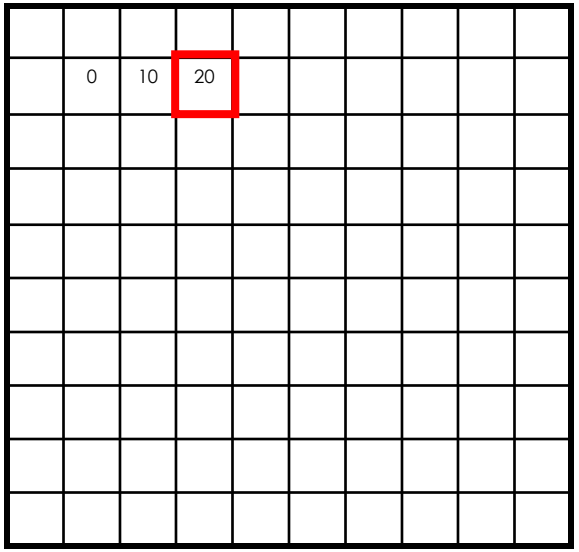
$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$



$h[.,.]$



$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

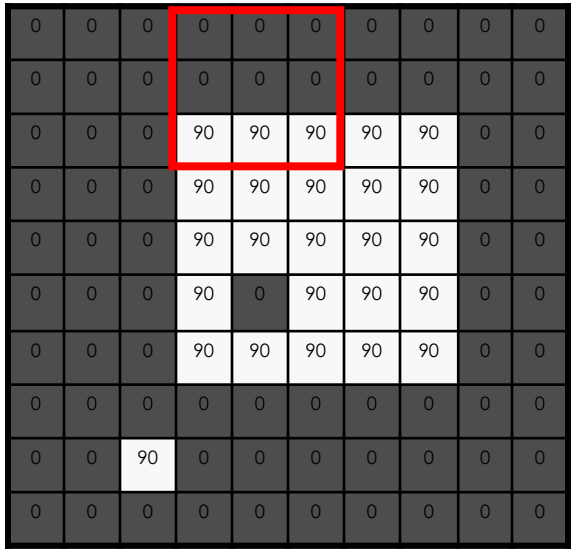
Credit: S. Seitz

Image filtering

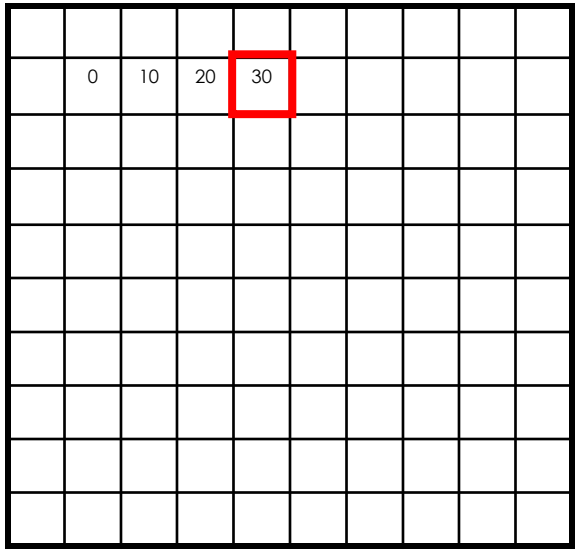
$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$



$h[.,.]$



$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

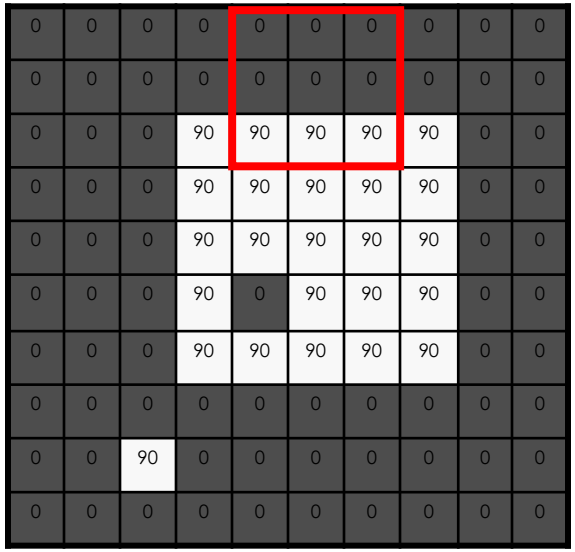
Credit: S. Seitz

Image filtering

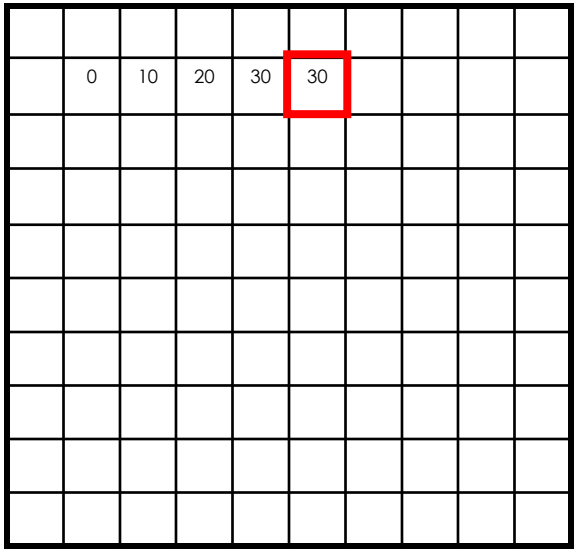
$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$



$h[.,.]$



$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

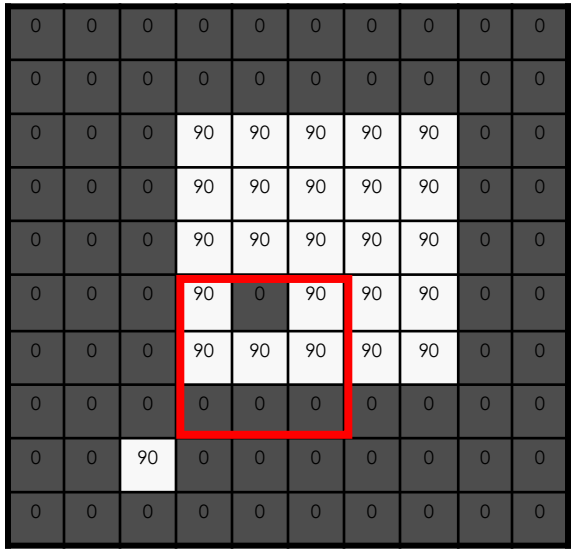
Credit: S. Seitz

Image filtering

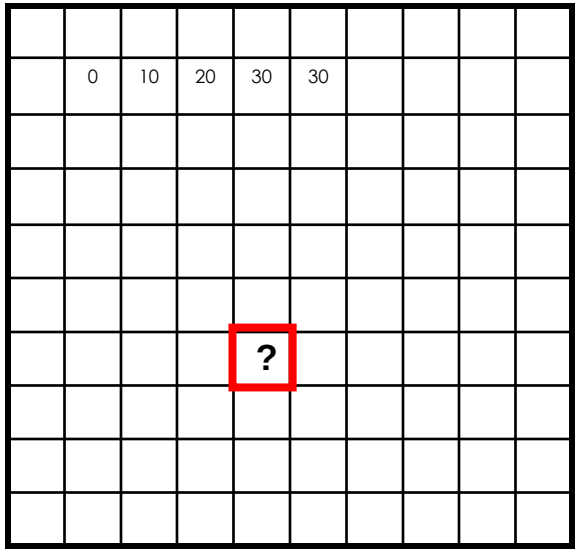
$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$



$h[.,.]$



$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

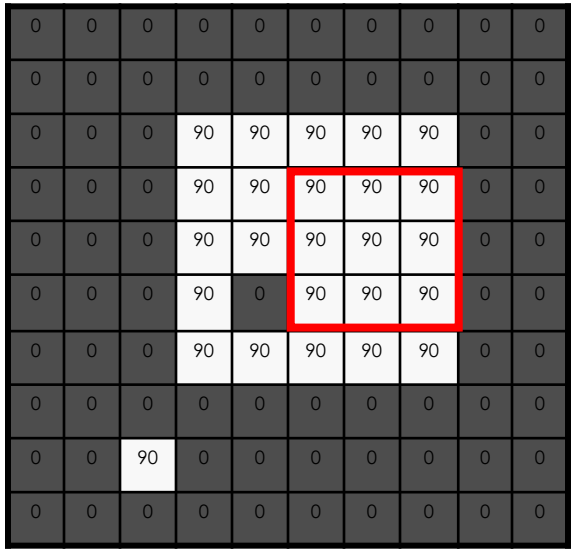
Credit: S. Seitz

Image filtering

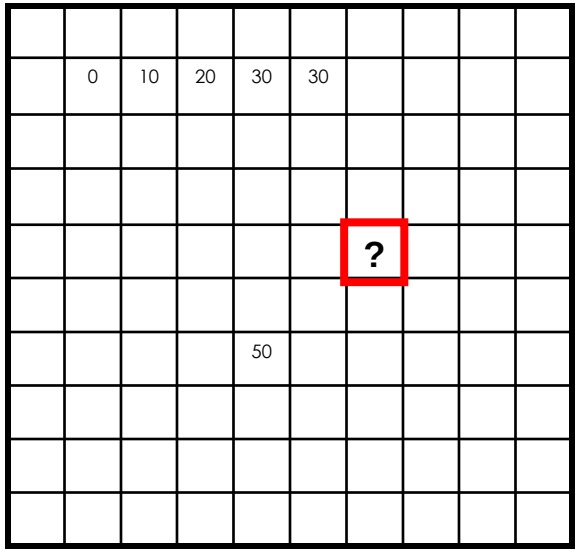
$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$



$h[.,.]$



$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

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

$h[.,.]$

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

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

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

$h[.,.]$

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

Informally, what does the filter do?

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

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

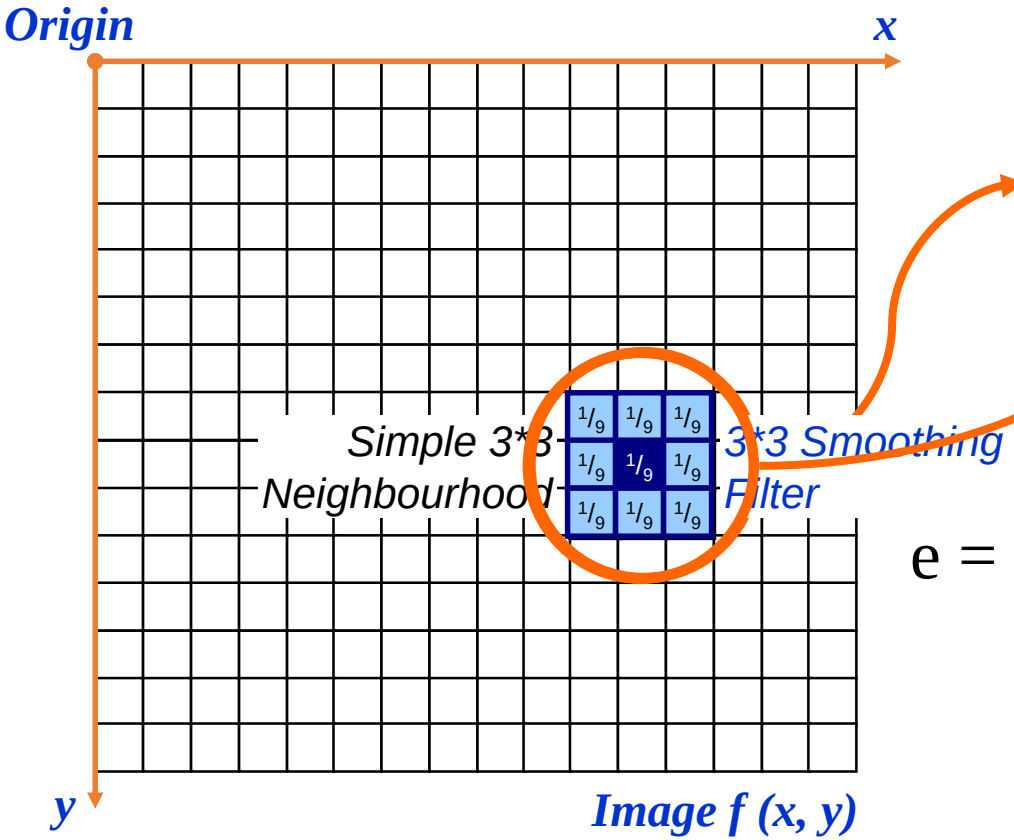
$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$

Simple averaging filter

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

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

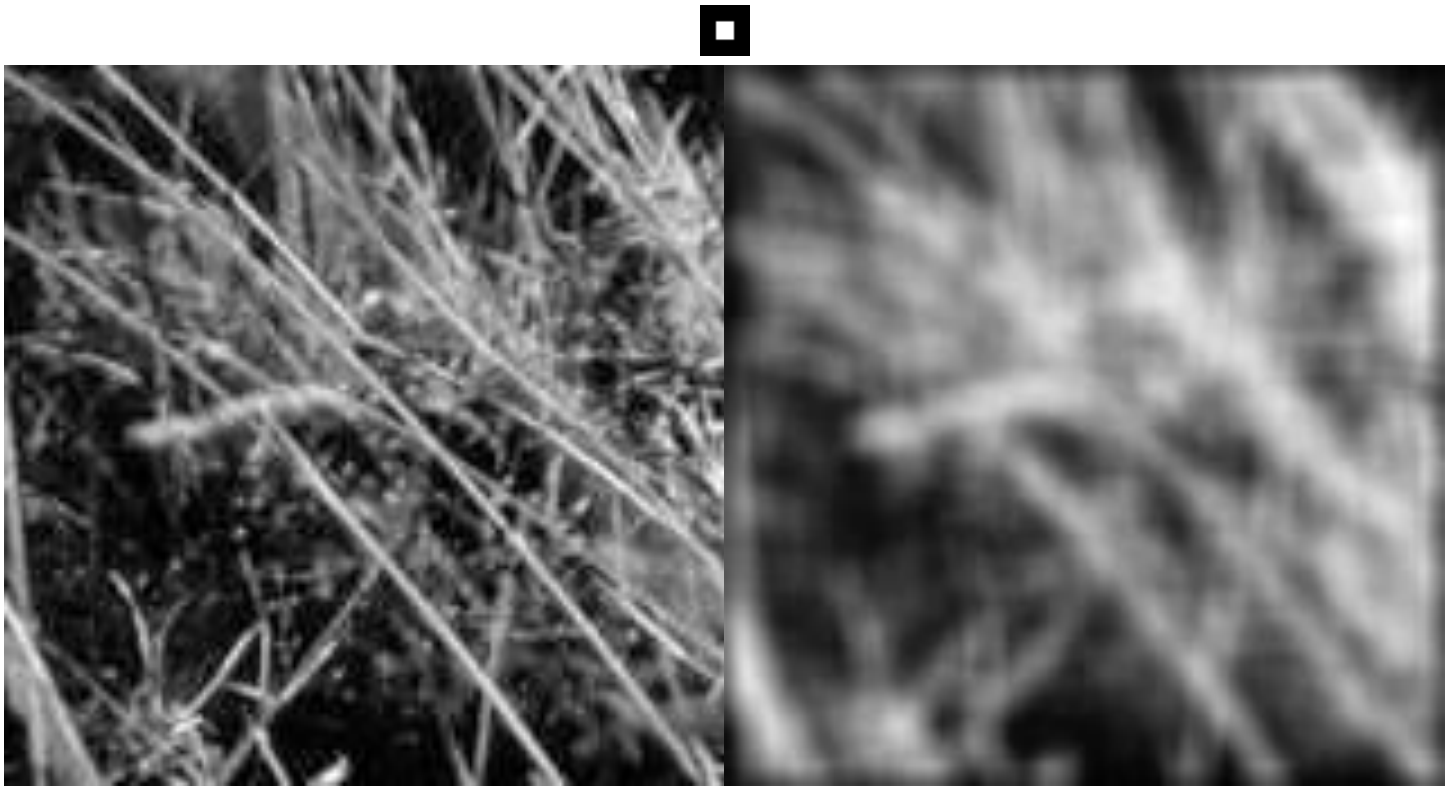
Filter

$$e = \tfrac{1}{9}*106 + \tfrac{1}{9}*104 + \tfrac{1}{9}*100 + \tfrac{1}{9}*108 + \tfrac{1}{9}*99 + \tfrac{1}{9}*98 + \tfrac{1}{9}*95 + \tfrac{1}{9}*90 + \tfrac{1}{9}*85$$

$$= 98.3333$$

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

Smoothing with box filter



Credit: D. Hoiem

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

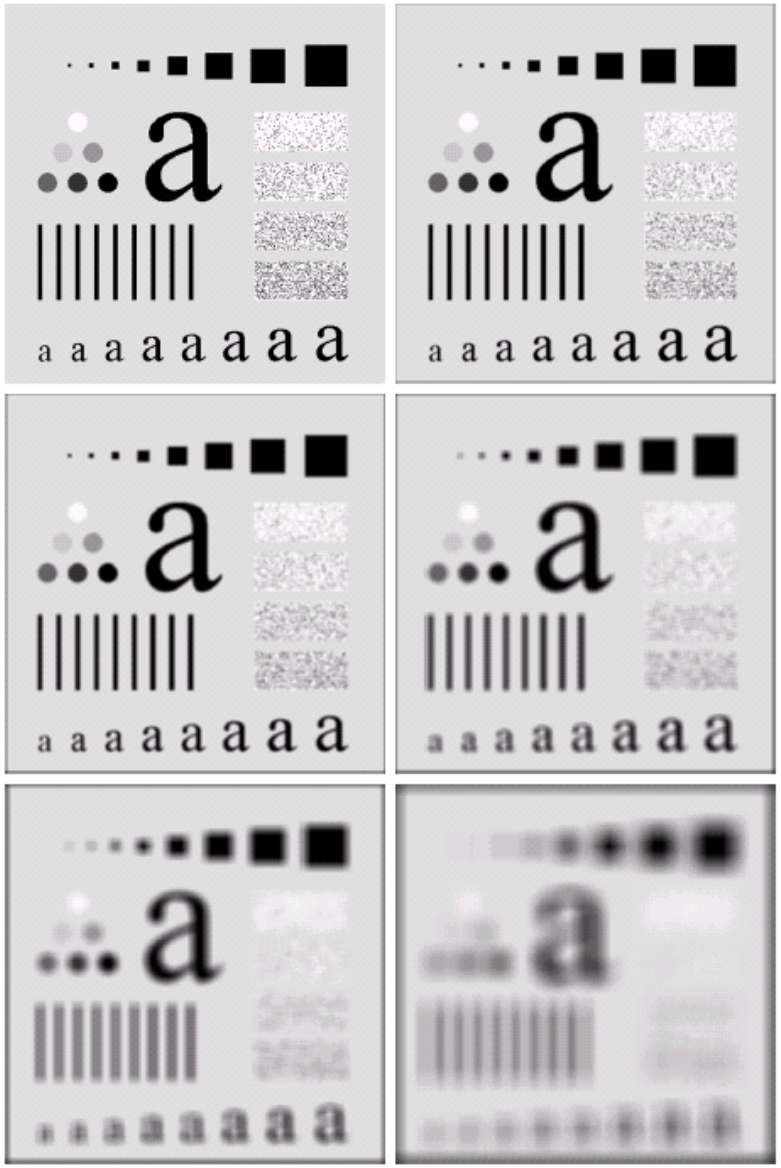
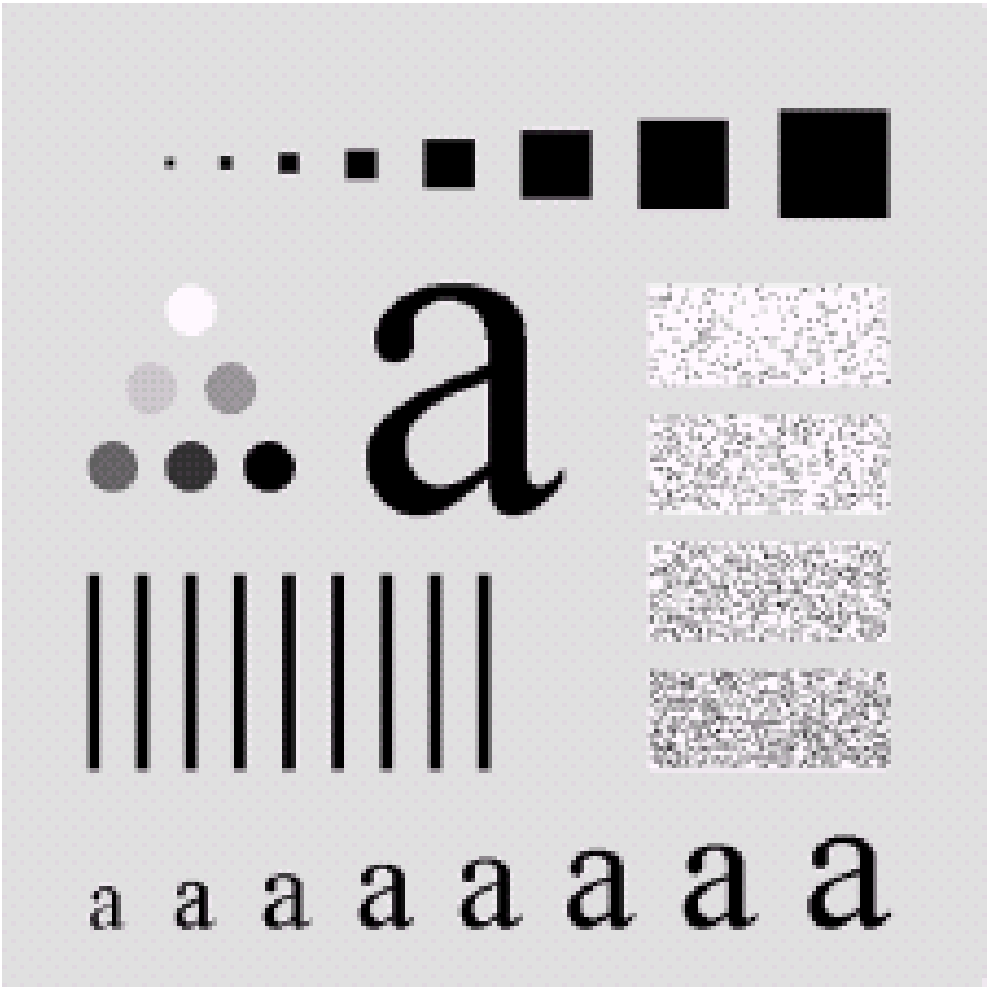
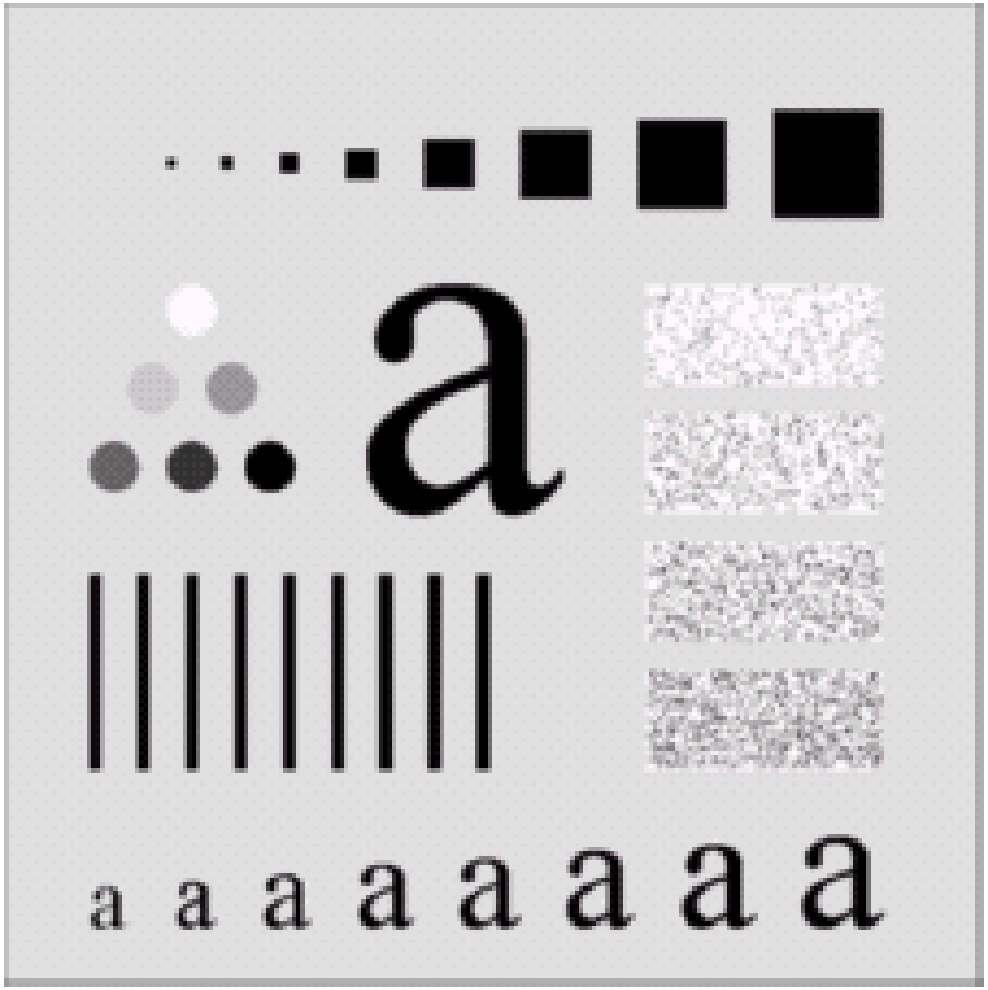


Image Smoothing Example



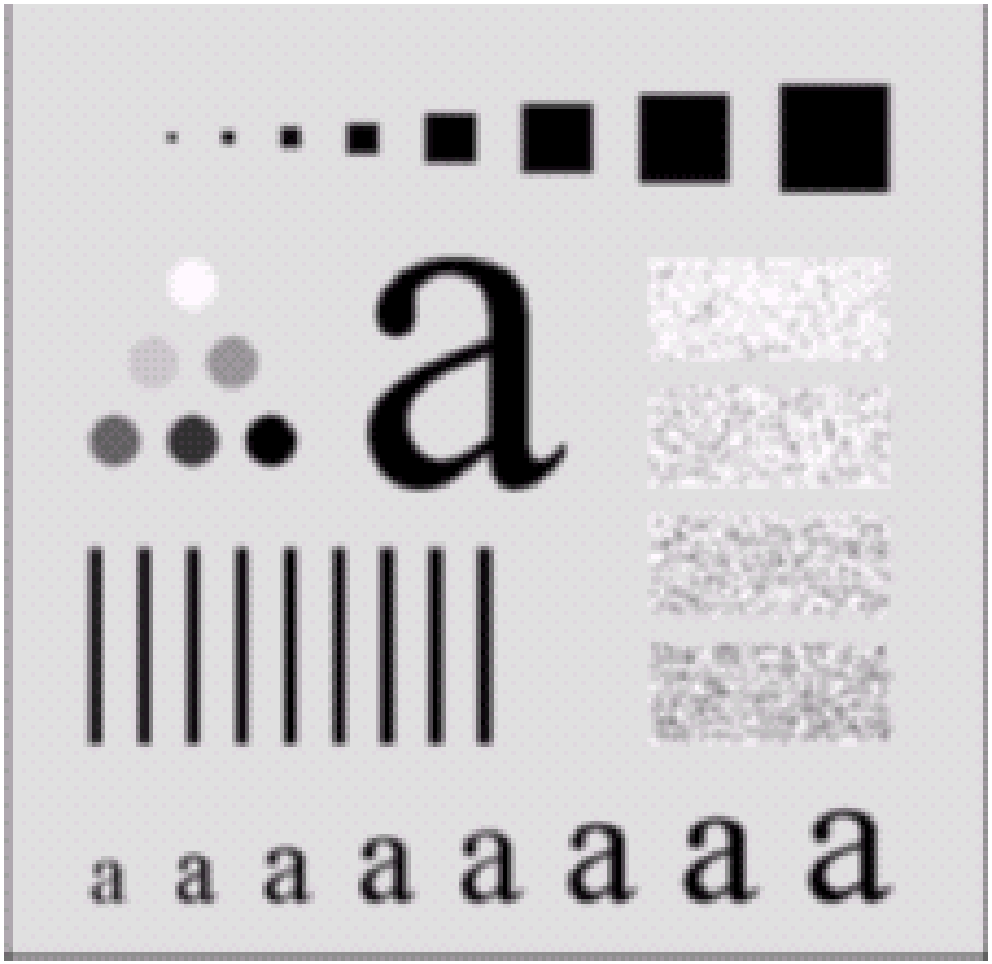
original

Image Smoothing Example



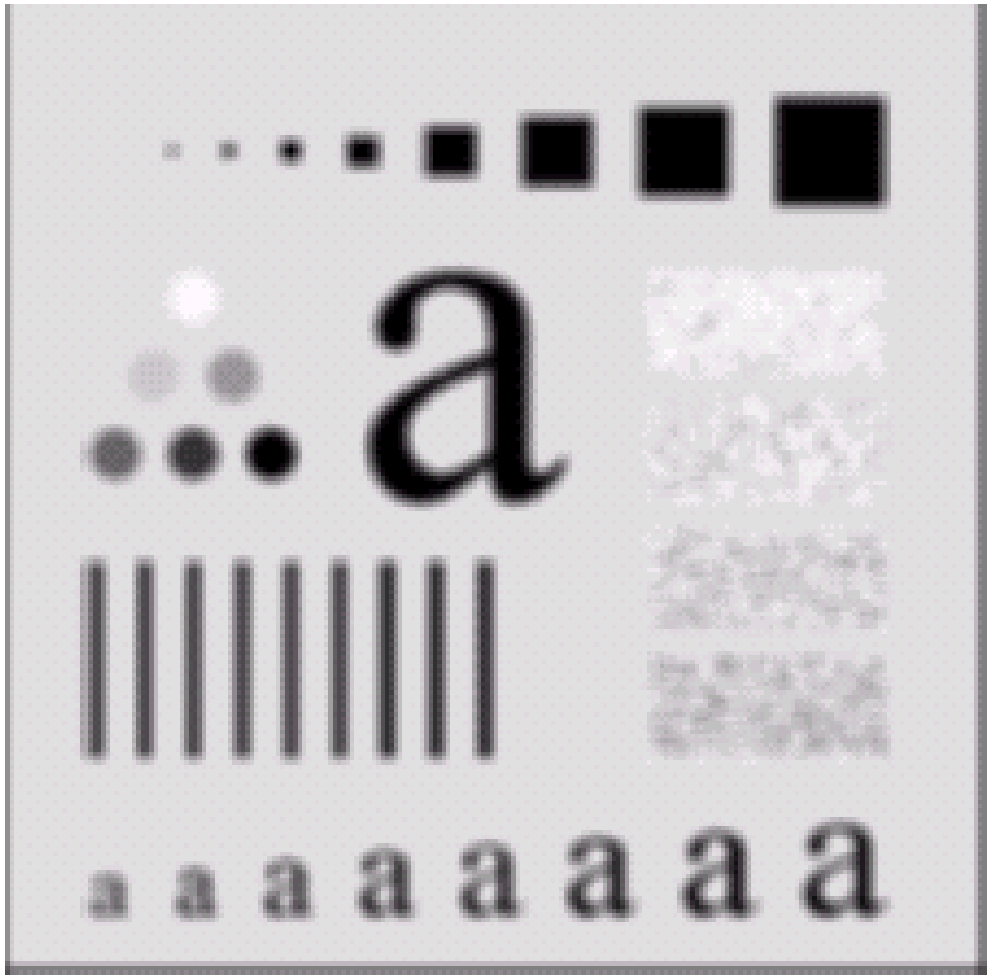
3x3

Image Smoothing Example



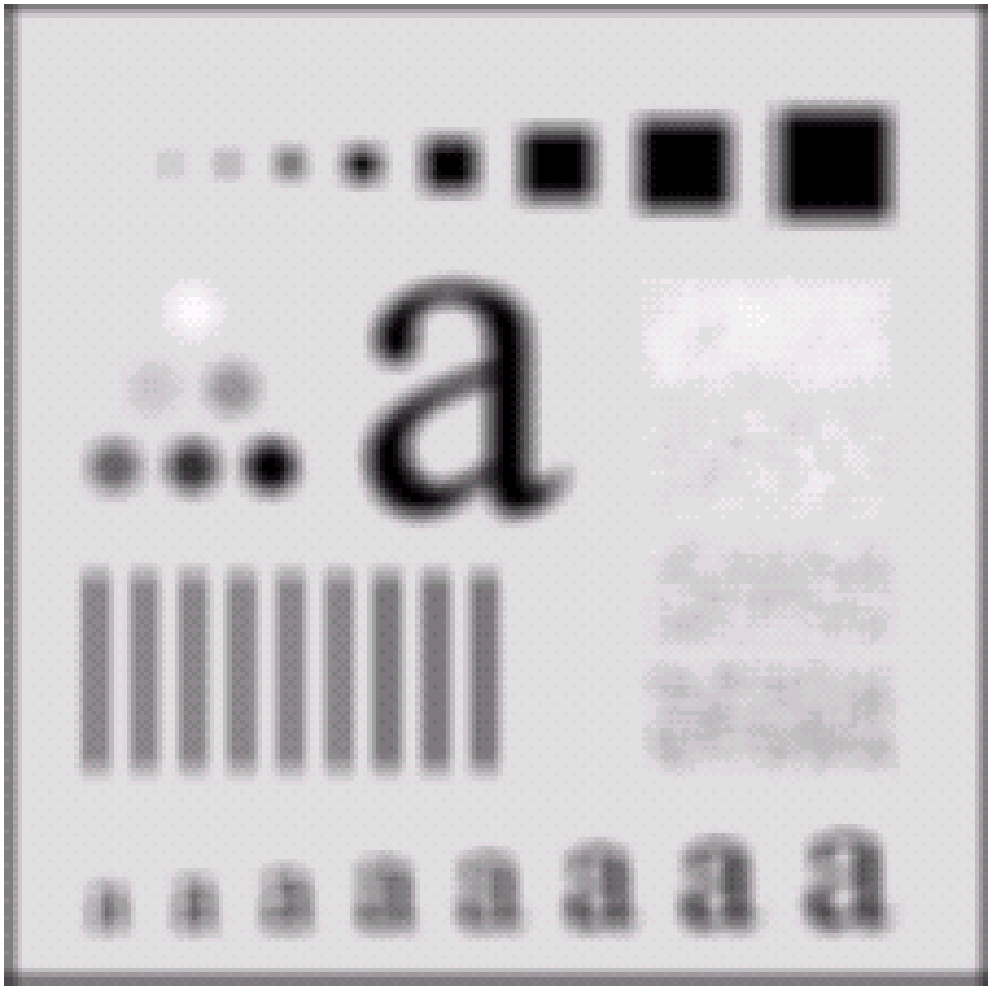
5x5

Image Smoothing Example



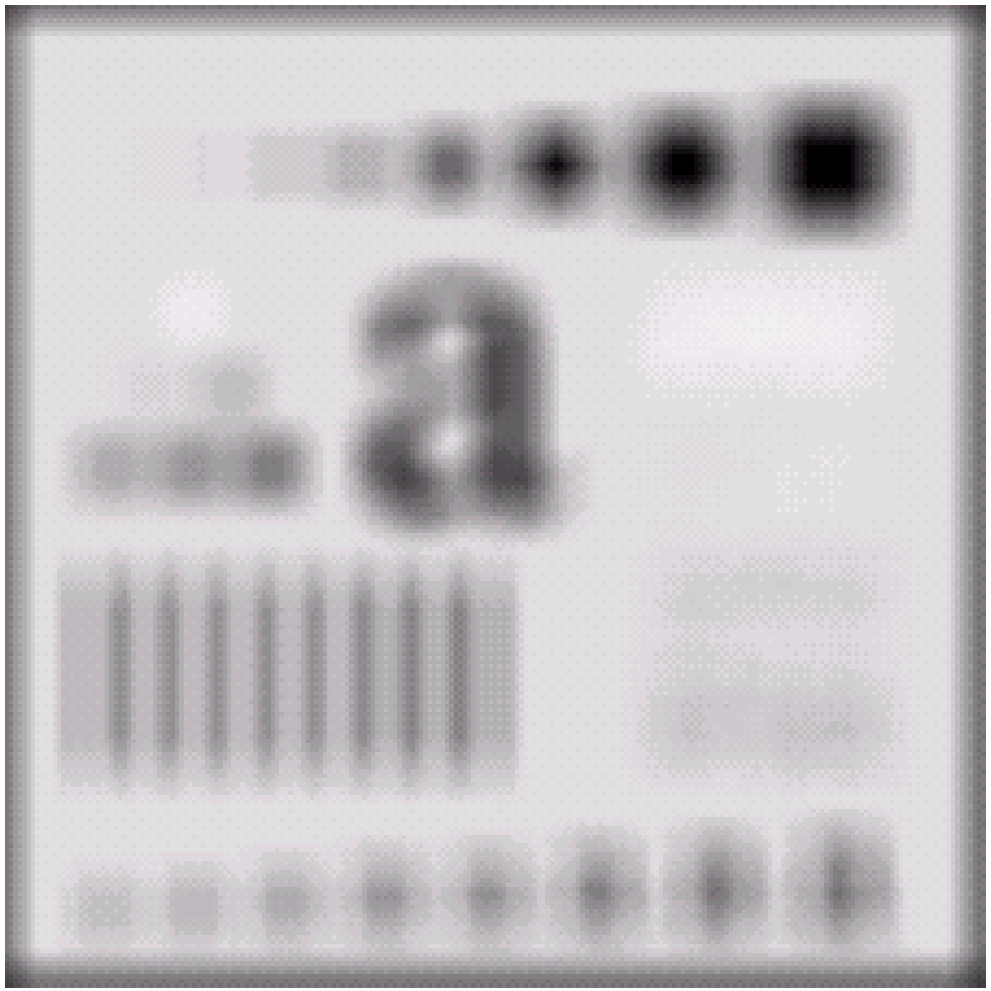
9x9

Image Smoothing Example



15x15

Image Smoothing Example



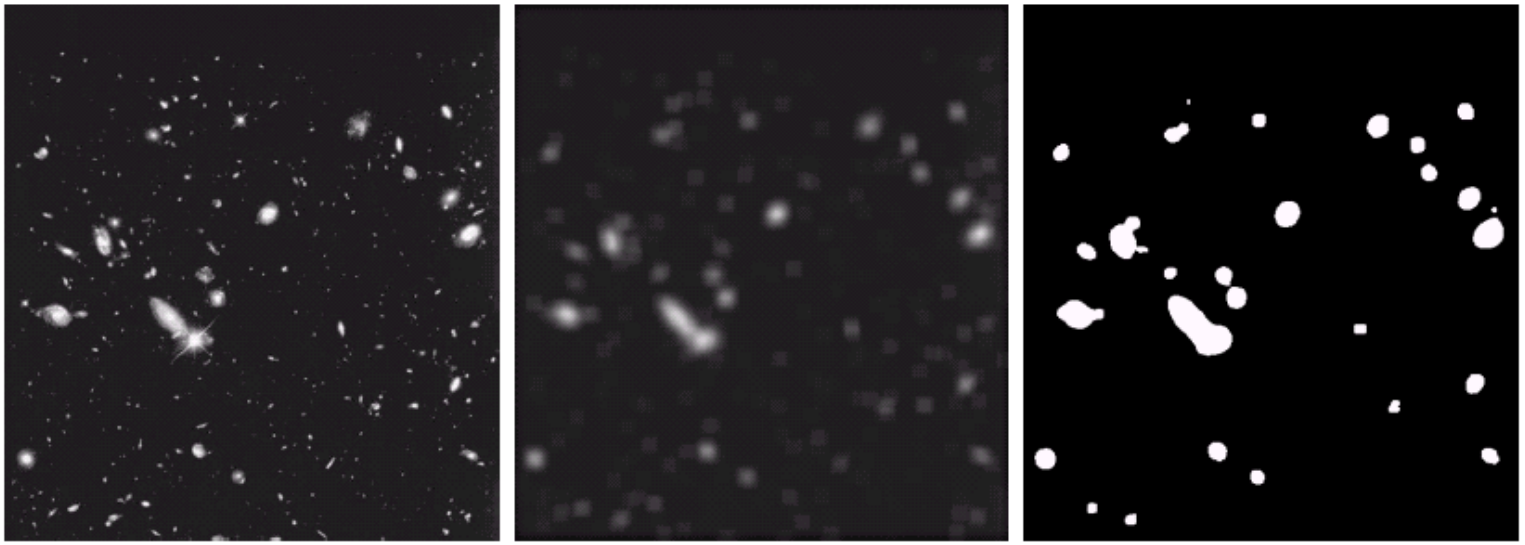
35x35

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

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.)



Original Image **Smoothed Image** **Thresholded Image**

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Weighted Smoothing Filters

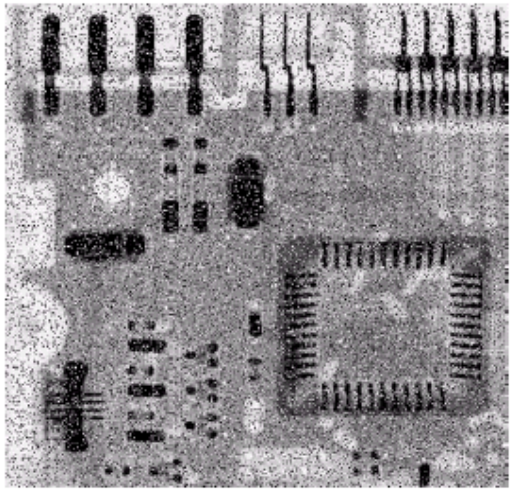
•More effective smoothing filters can be generated by allowing different pixels in the neighbourhood different weights in the averaging function

- 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

Averaging Filter Vs. Median Filter Example



Original Image With Noise

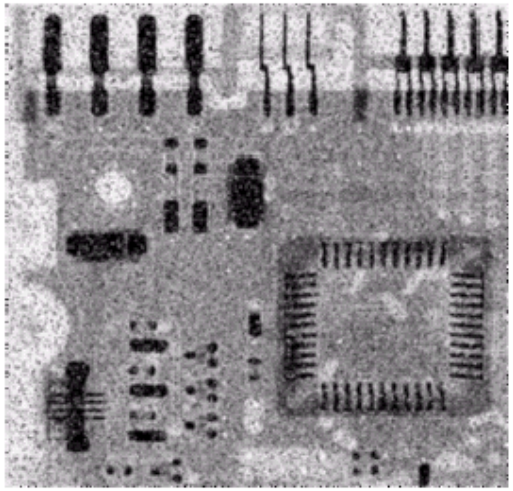


Image After Averaging Filter 3x3

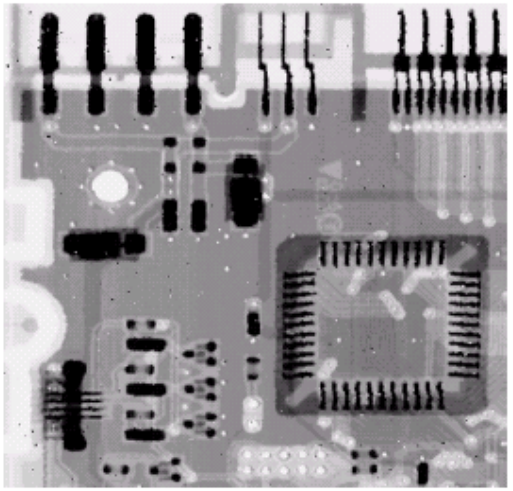
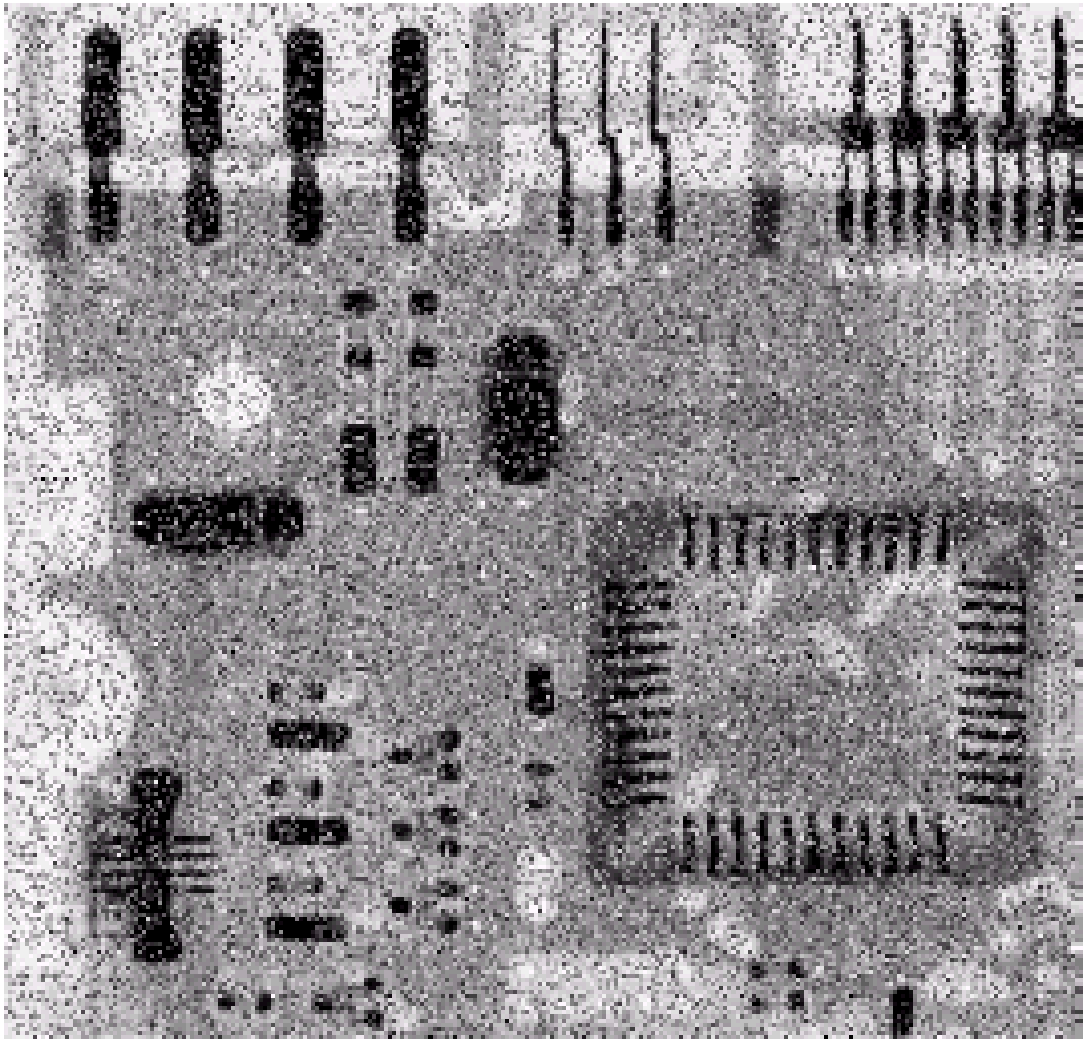


Image After Median Filter 3x3

- Filtering is often used to remove noise from images
- Sometimes a median filter works better than an averaging filter

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Averaging Filter Vs. Median Filter Example



Original Image With Noise

Averaging Filter Vs. Median Filter Example

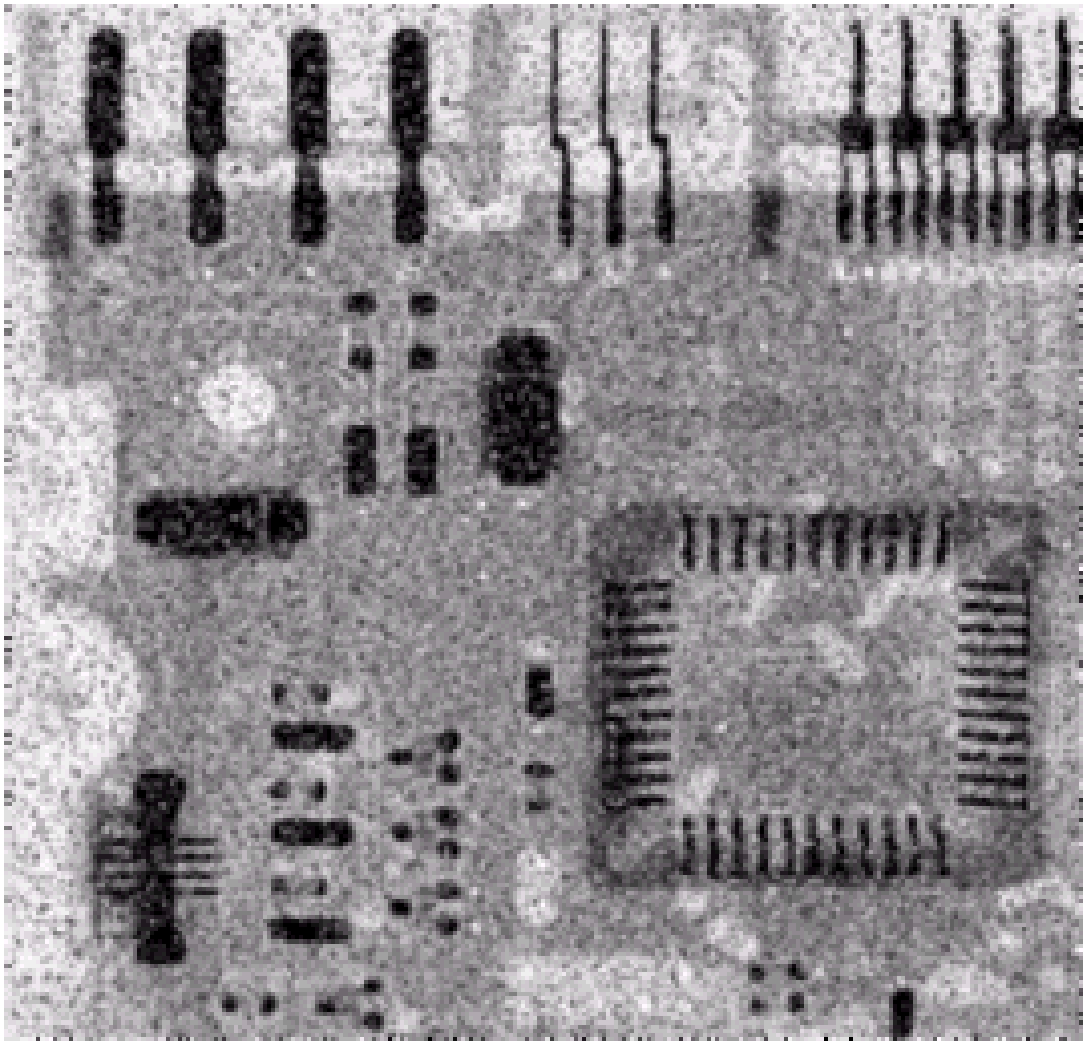


Image After Averaging Filter

Averaging Filter Vs. Median Filter Example

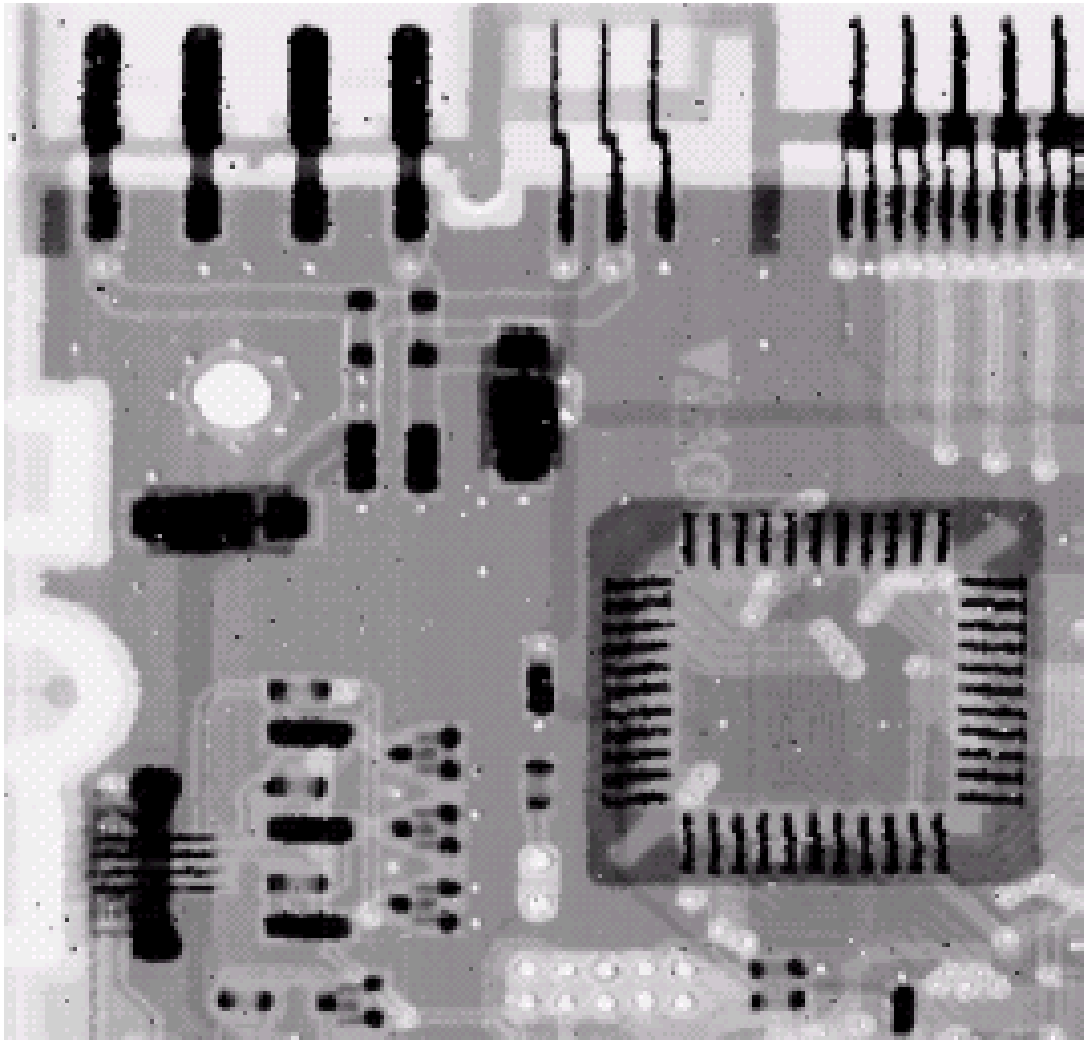
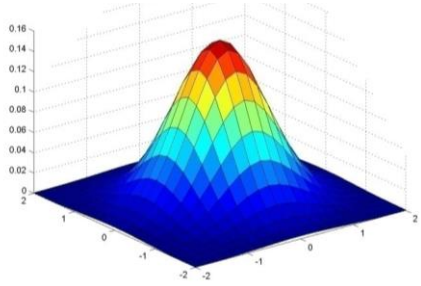
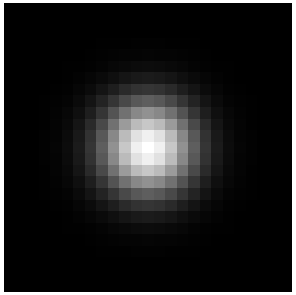


Image After Median Filter

Important filter: Gaussian

- Weight contributions of neighboring pixels by nearness





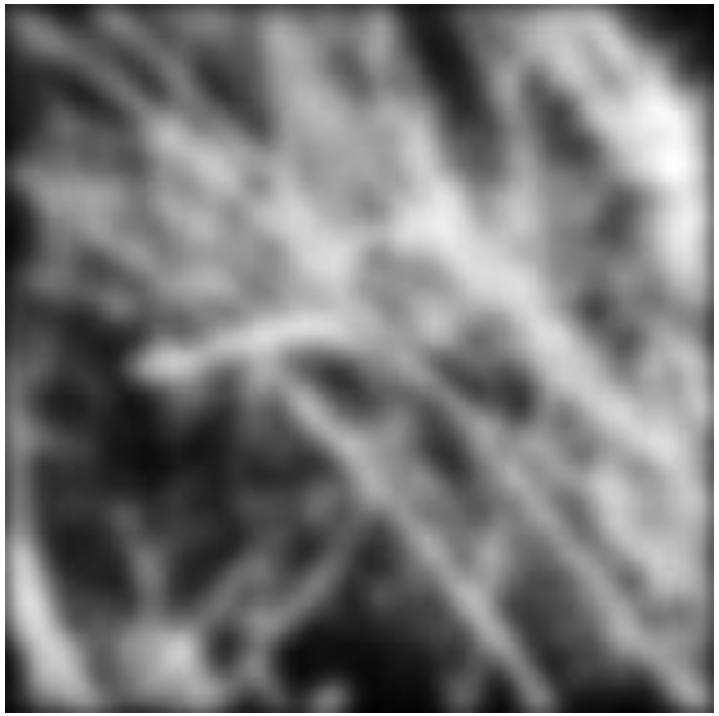
0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

5 x 5, $\sigma = 1$

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

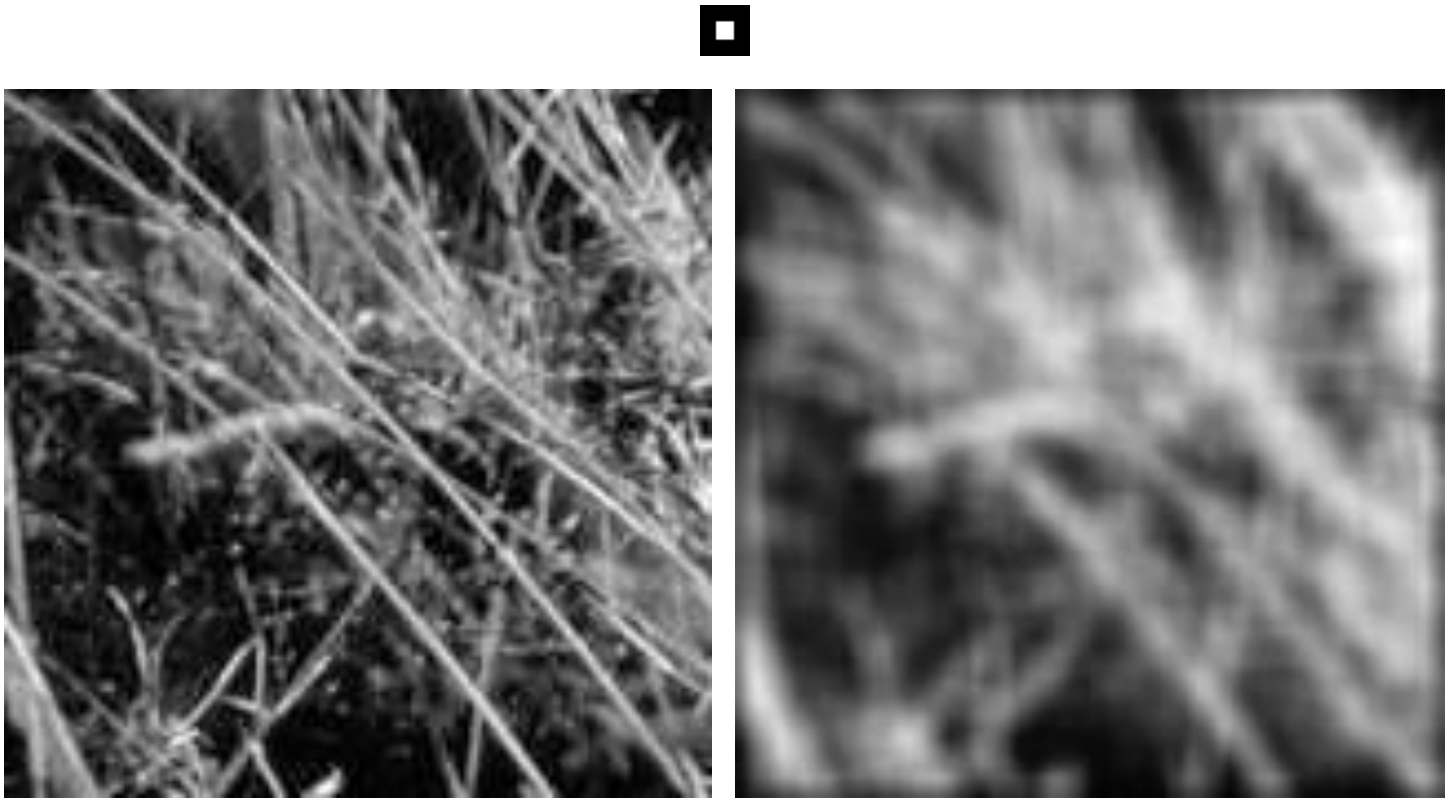
Slide credit: C. Rasmussen

Smoothing with Gaussian filter



Credit: D. Hoiem

Smoothing with Box filter



Credit: D. Hoiem

Gaussian filters

- Remove "high-frequency" components from the image (low-pass filter)
  - Images become more smooth
- Convolution with self is another Gaussian
  - So can smooth with small-width kernel, repeat, and get same result as larger-width kernel would have
  - Convolving two times with Gaussian kernel of width σ is same as convolving once with kernel of width $\sigma\sqrt{2}$
- *Separable* kernel
  - Factors into product of two 1D Gaussians

Source: K. Grauman

Separability of the Gaussian filter

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$= \left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{x^2}{2\sigma^2}}\right)\left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{y^2}{2\sigma^2}}\right)$$

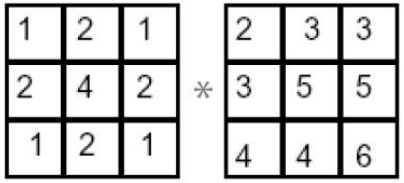
The 2D Gaussian can be expressed as the product of two functions, one a function of x and the other a function of y

In this case, the two functions are the (identical) 1D Gaussian

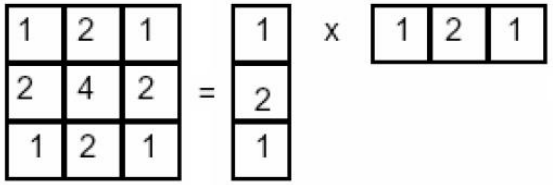
Source: D. Lowe

Separability example

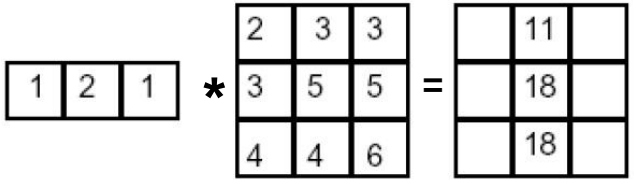
2D filtering
(center location only)



The filter factors
into a product of 1D
filters:



Perform filtering
along rows:

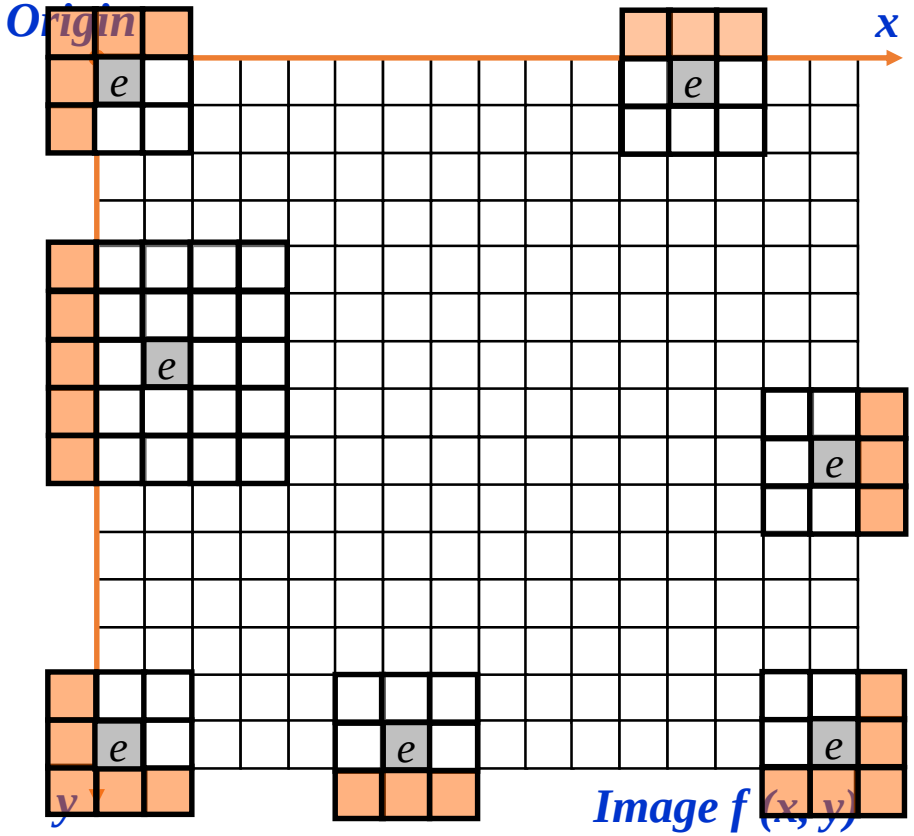


Followed by filtering
along the remaining column:

Source: K. Grauman

Strange Things Happen At The Edges!

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

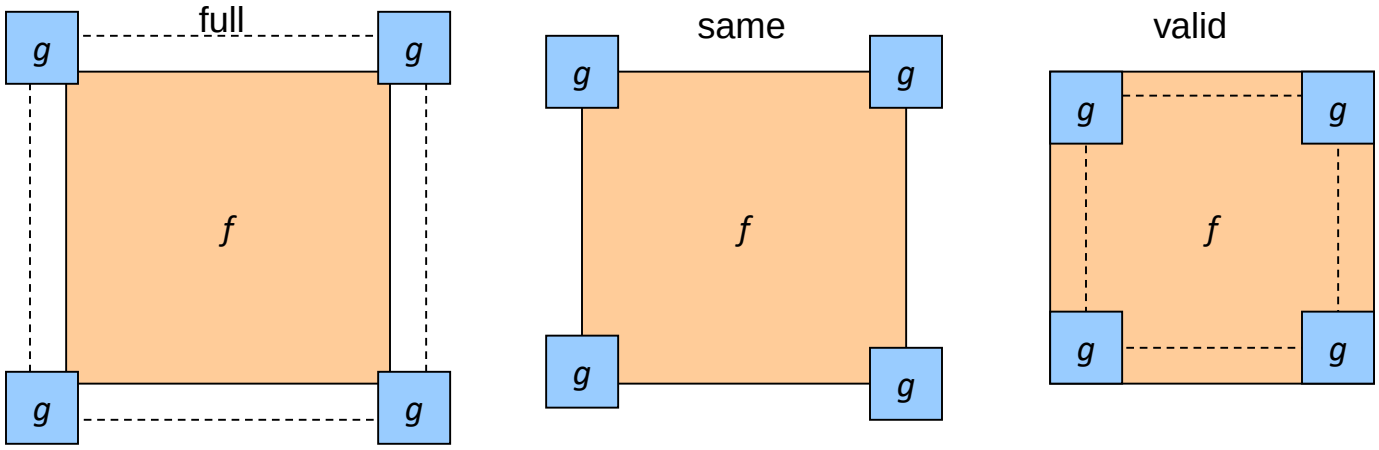


Strange Things Happen At The Edges! (cont…)

- 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

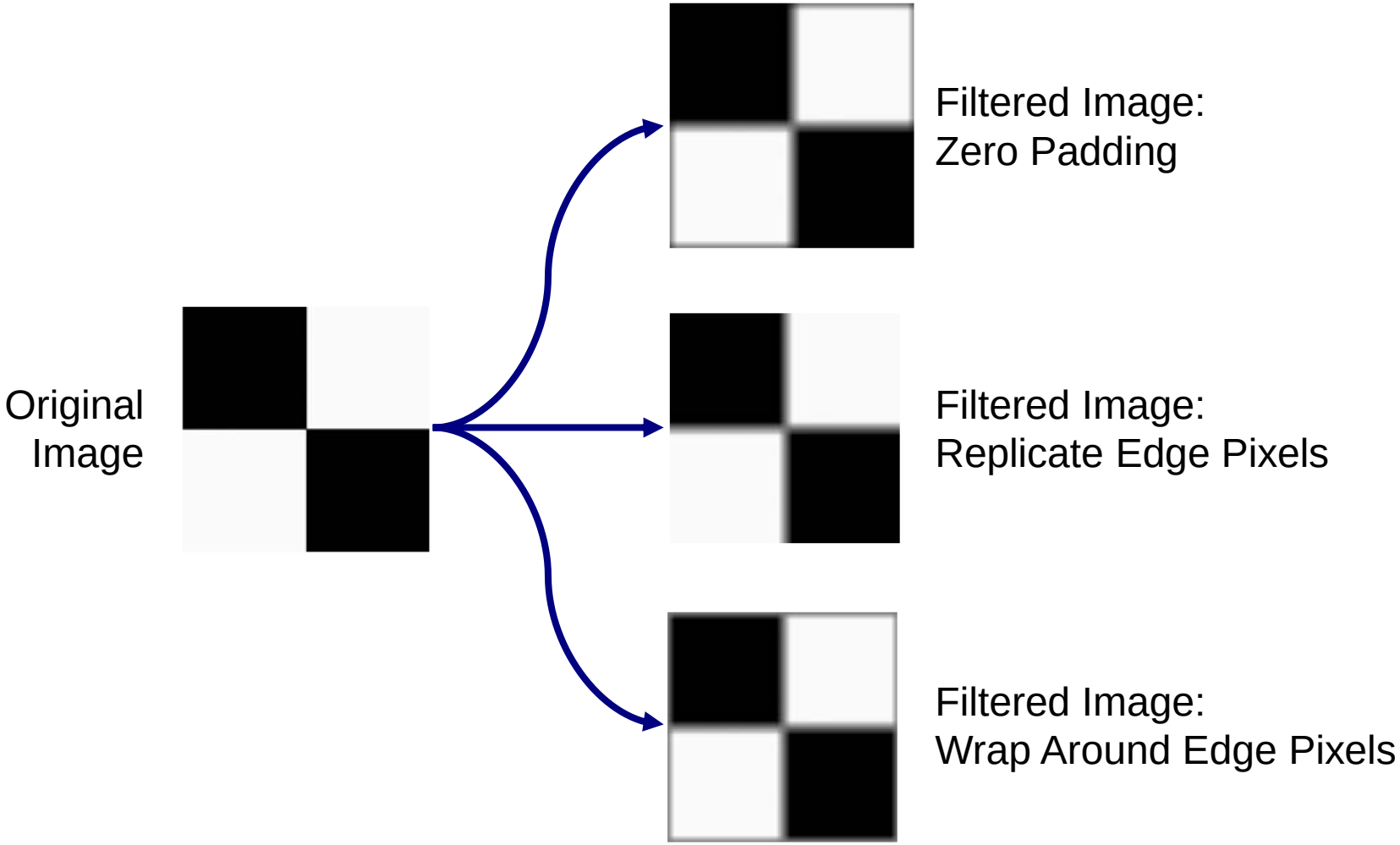
Practical matters

- What is the size of the output?
- Python: `convolve2d(g, f, mode)`
  - *mode* = ‘full’: output size is sum of sizes of f and g
  - *mode* = ‘same’: output size is same as f
  - *mode* = ‘valid’: output size is difference of sizes of f and g



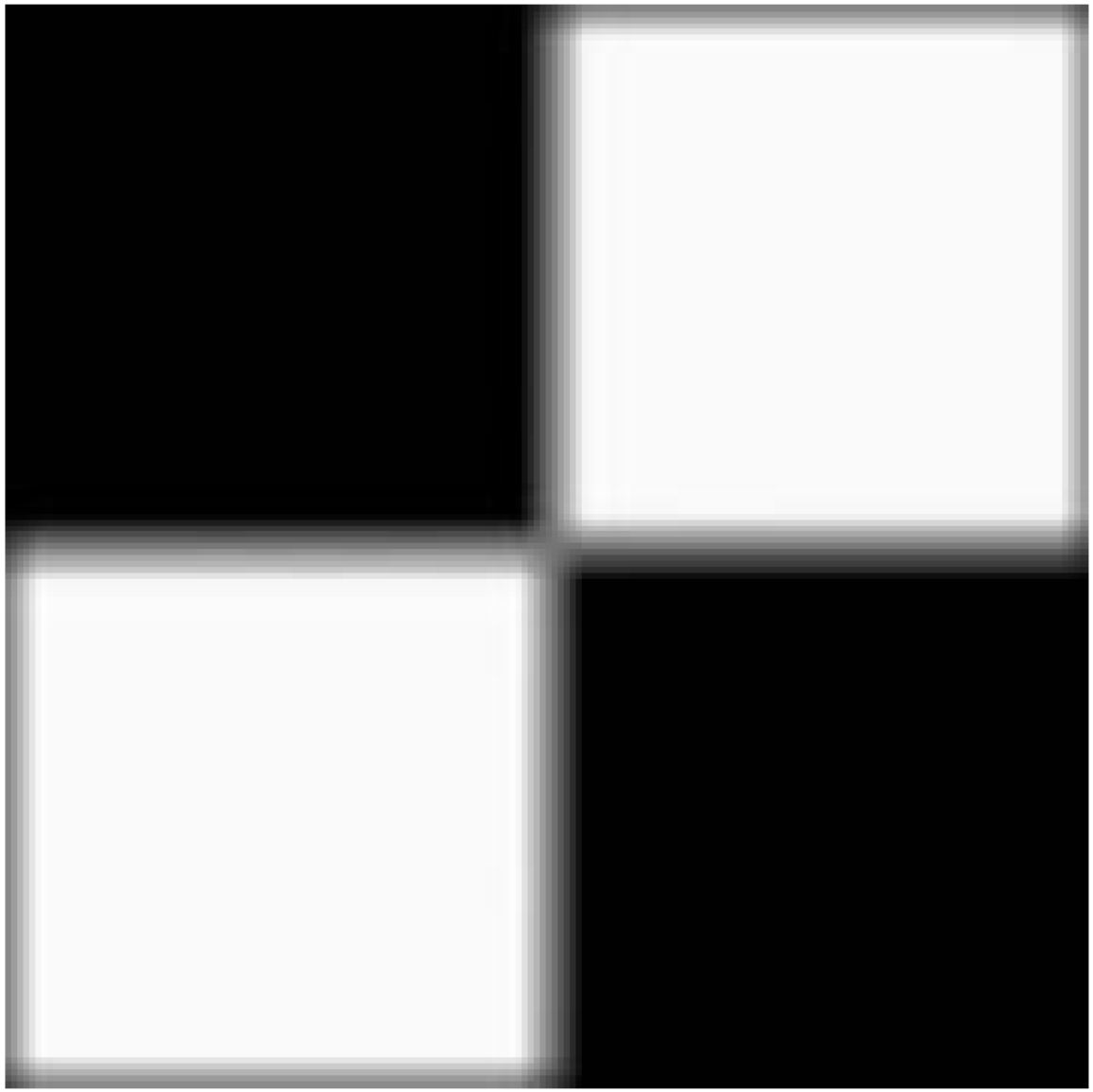
Credit: D. Hoiem

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



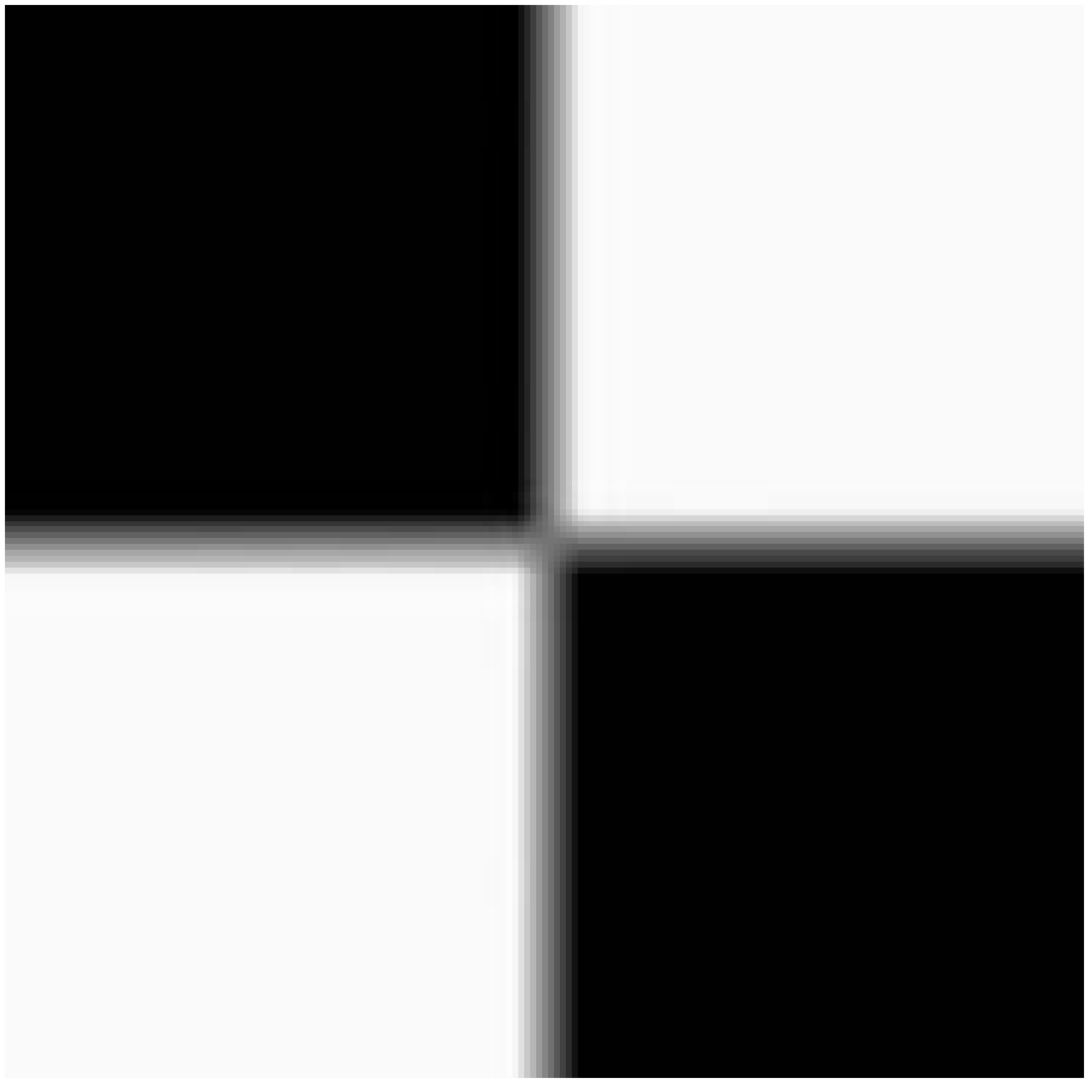
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Strange Things Happen At The Edges! (cont…)



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Strange Things Happen At The Edges! (cont…)



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Strange Things Happen At The Edges! (cont…)

Correlation & Convolution

- The filtering we have been talking about so far is referred to as c*orrelation* with the filter itself referred to as the *correlation kernel*

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

- *Convolution* is a similar operation, with just one subtle difference

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

- For symmetric filters it makes no difference

a	b	c
d	e	e
f	g	h

Original Image Pixels

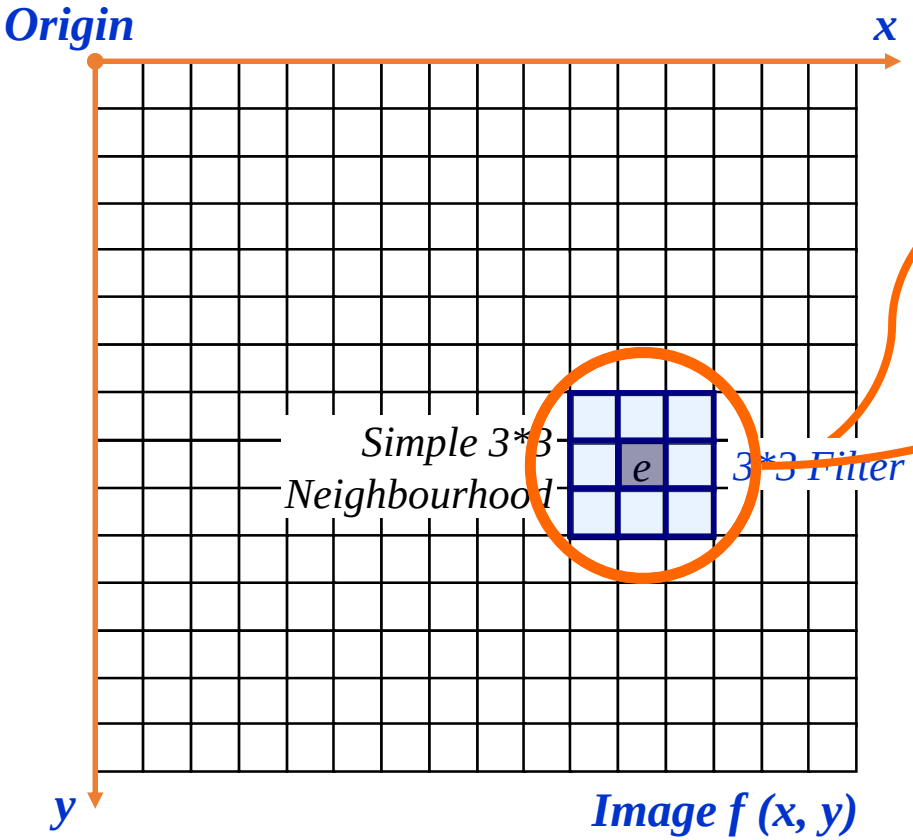
$*$

r	s	t
u	v	w
x	y	z

Filter

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

Spatial Filtering Refresher



a	*b*	*c*
d	*e*	*f*
g	*h*	*i*

Original Image Pixels

*

r	*s*	*t*
u	*v*	*w*
x	*y*	*z*

Filter

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

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

Sharpening Spatial Filters

Previously we have looked at smoothing filters which remove fine detail

Sharpening spatial filters seek to highlight fine detail

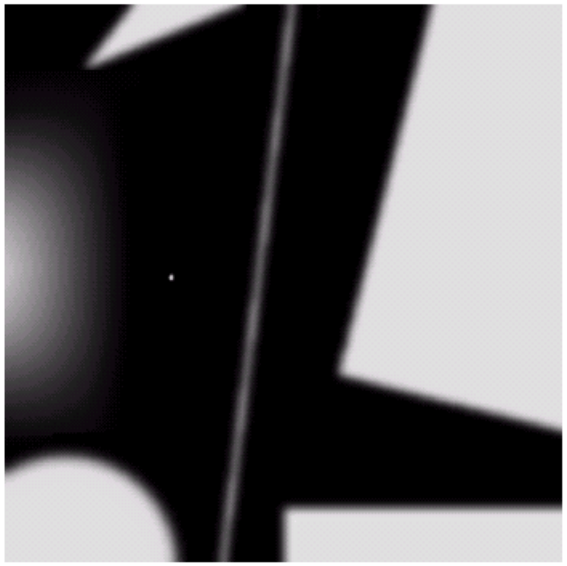
- Remove blurring from images
- Highlight edges

Sharpening filters are based on *spatial differentiation*

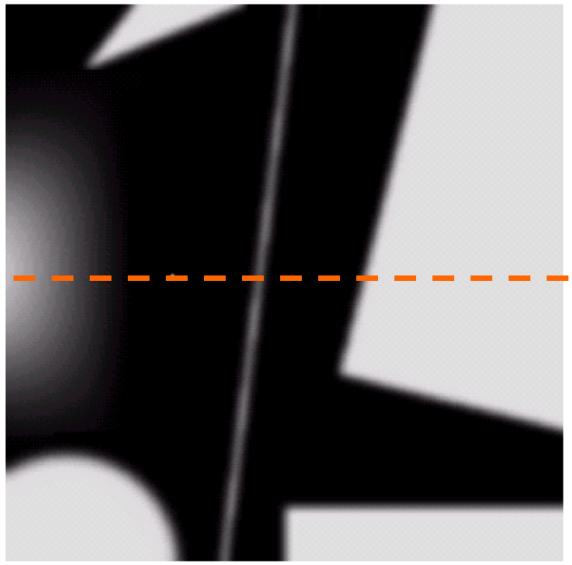
Spatial Differentiation

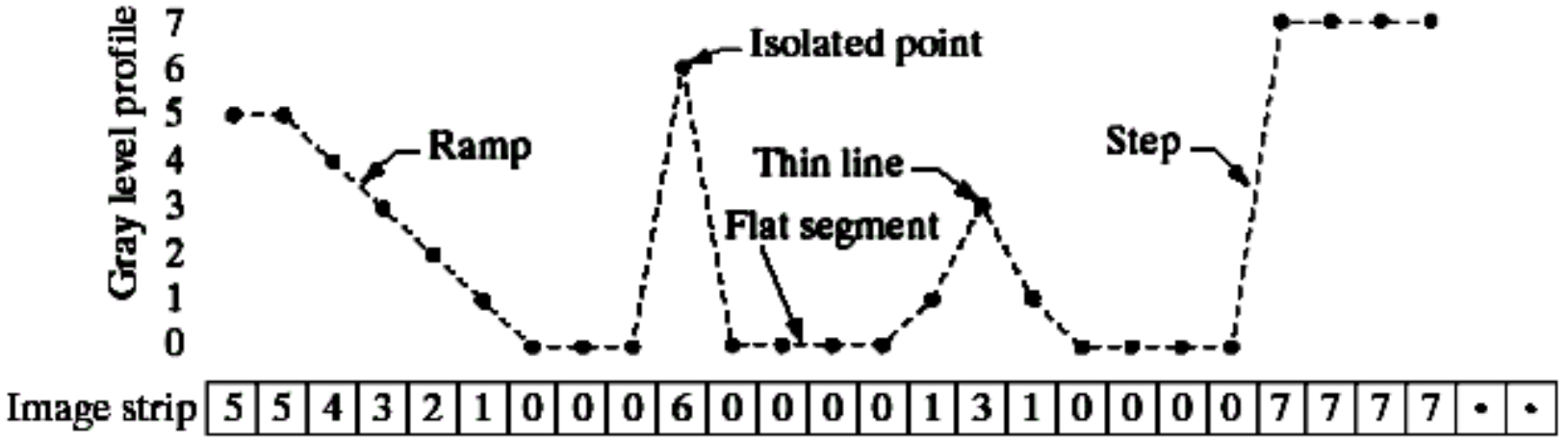
Differentiation measures the *rate of change* of a function

Let's consider a simple 1 dimensional example



Spatial Differentiation





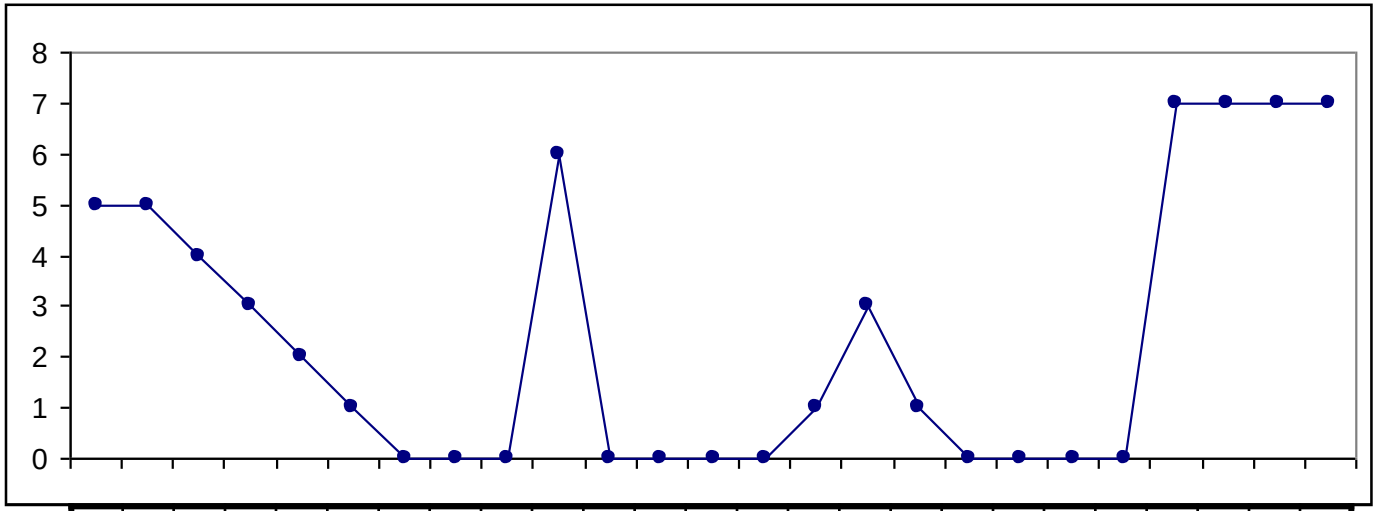
1st Derivative

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

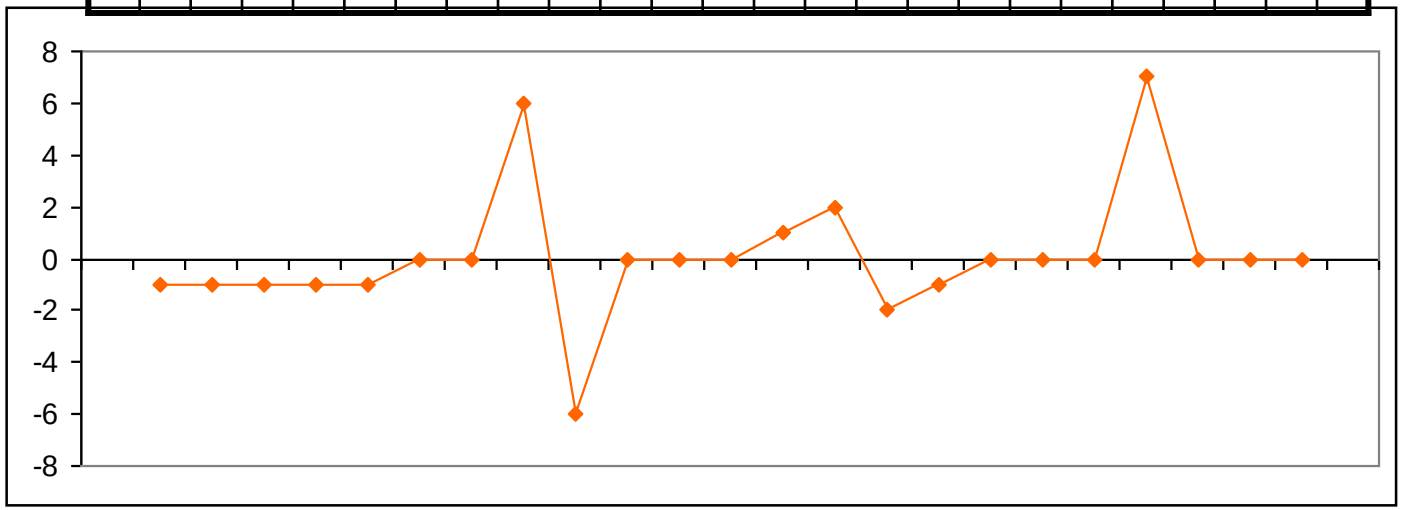
$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

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

1st Derivative (cont…)



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



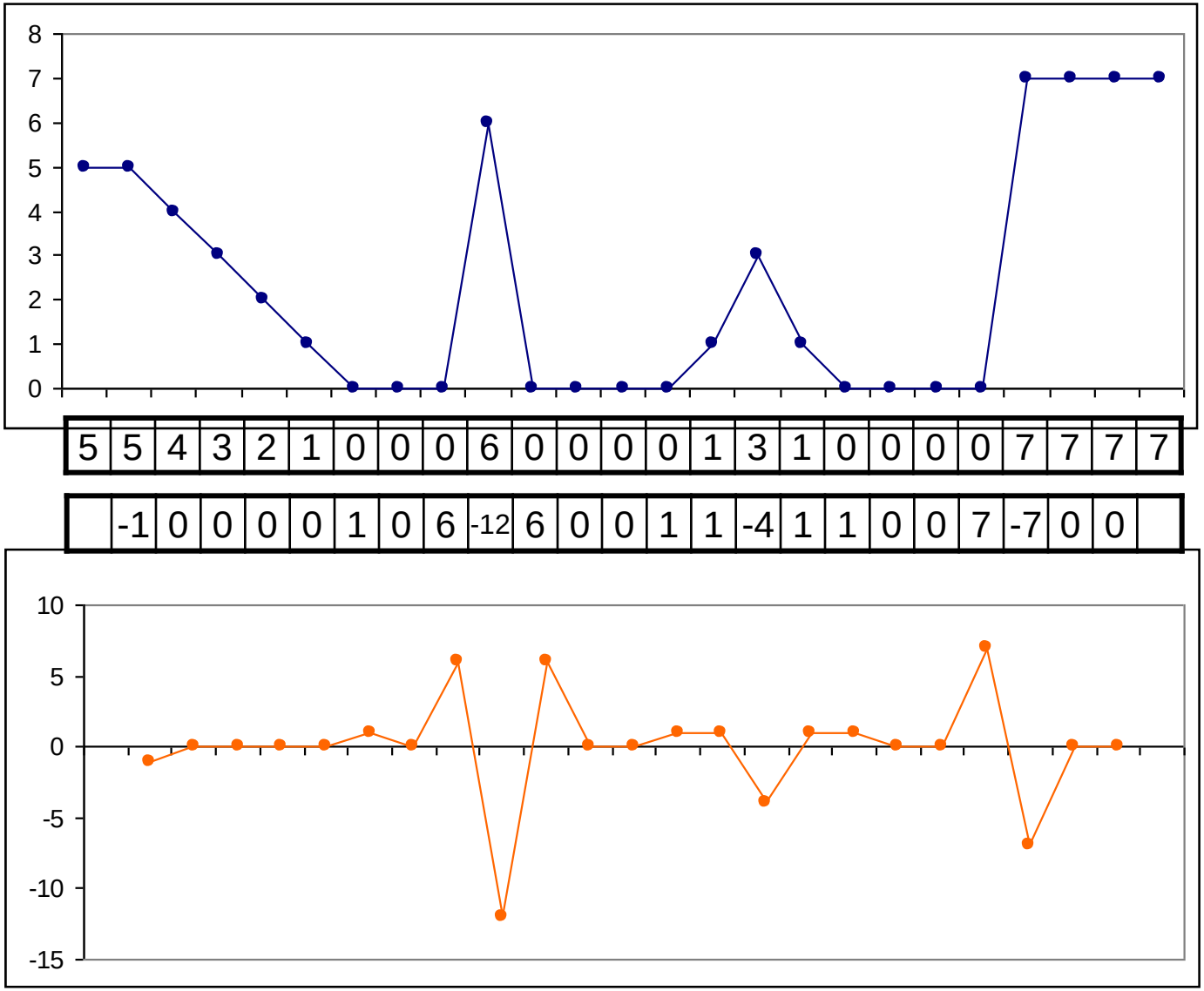
2nd Derivative

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

2nd Derivative (cont…)



Using Second Derivatives For Image Enhancement

The 2nd derivative is more useful for image enhancement than the 1st derivative

- Stronger response to fine detail
- Simpler implementation
- We will come back to the 1st order derivative later on

The first sharpening filter we will look at 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 2nd 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…)

So, the Laplacian can be given as follows:

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

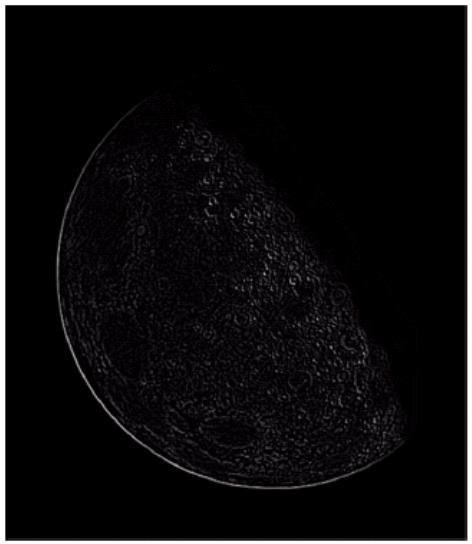
We can easily build a filter based on this

0	1	0
1	-4	1
0	1	0

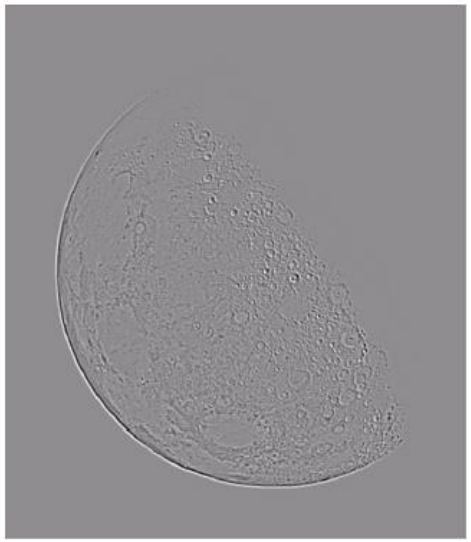
The Laplacian (cont…)

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

Original Image



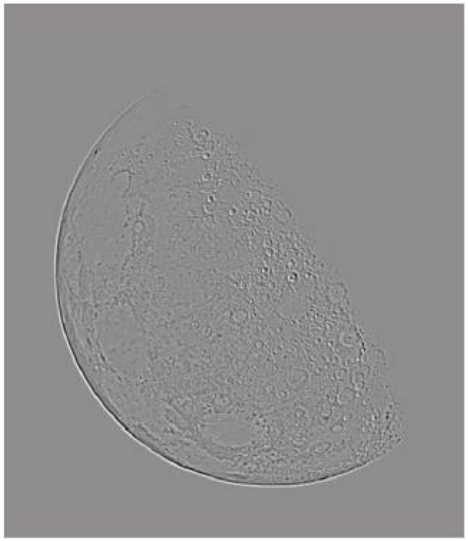
Laplacian Filtered Image



Laplacian Filtered Image Scaled for Display

But That Is Not Very Enhanced!

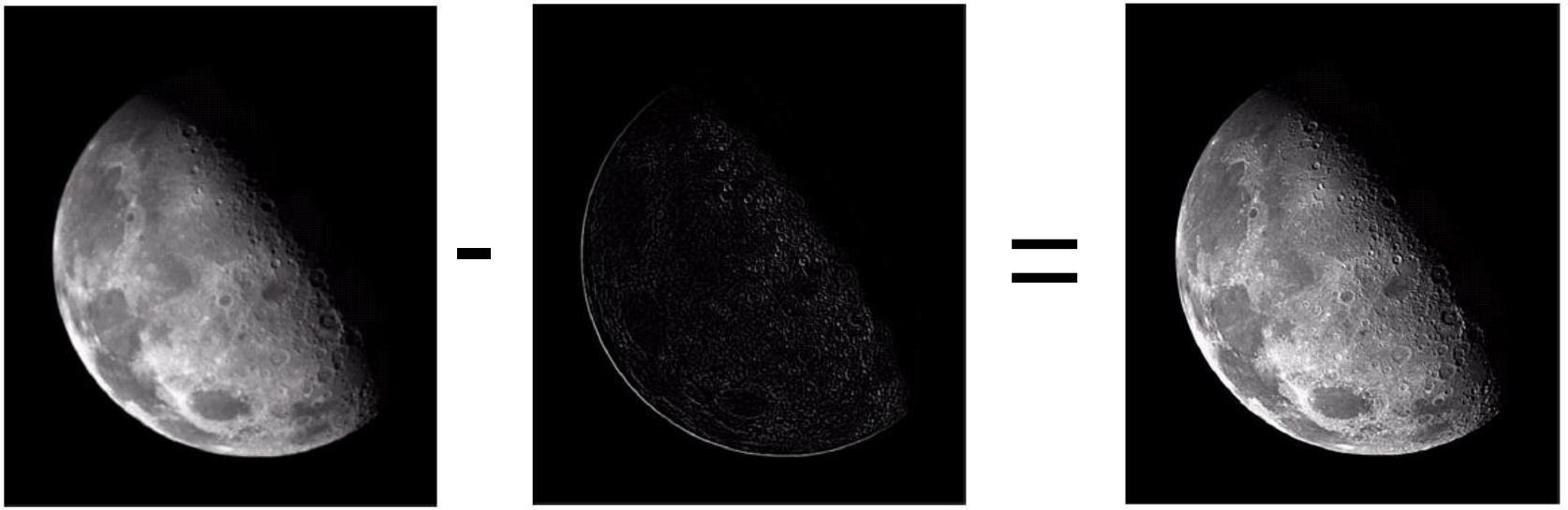
- The result of a Laplacian filtering is not an enhanced image
- We have to do more work in order to get our final image
- Subtract the Laplacian result from the original image to generate our final sharpened enhanced image



Laplacian
Filtered Image
Scaled for Display

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

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

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

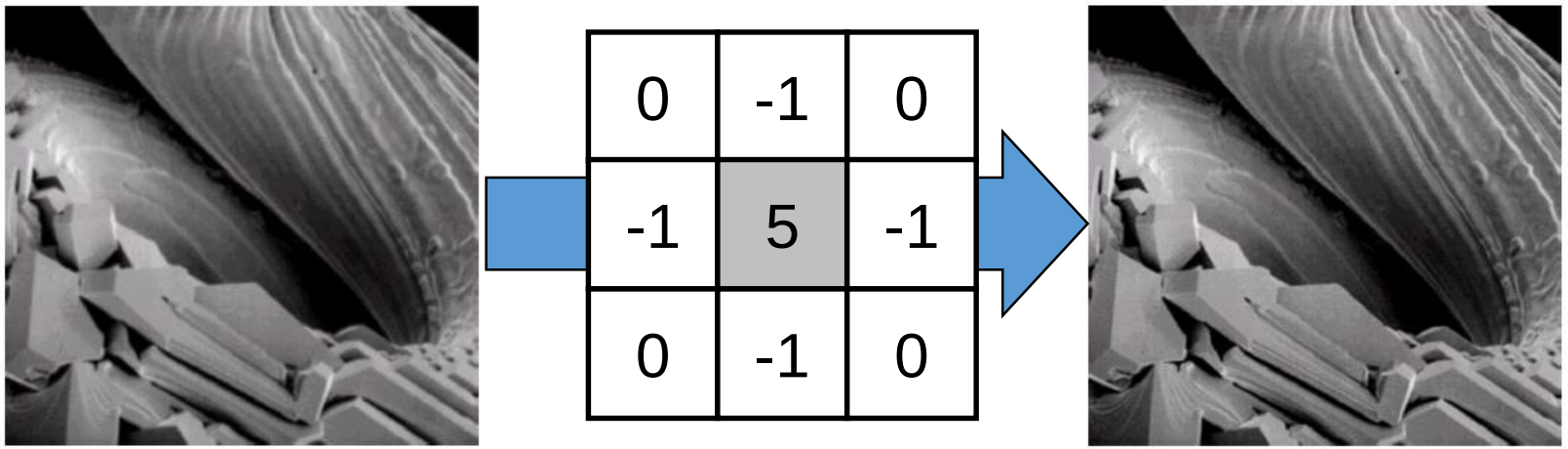
Simplified Image Enhancement

The entire enhancement can be combined into a single filtering operation

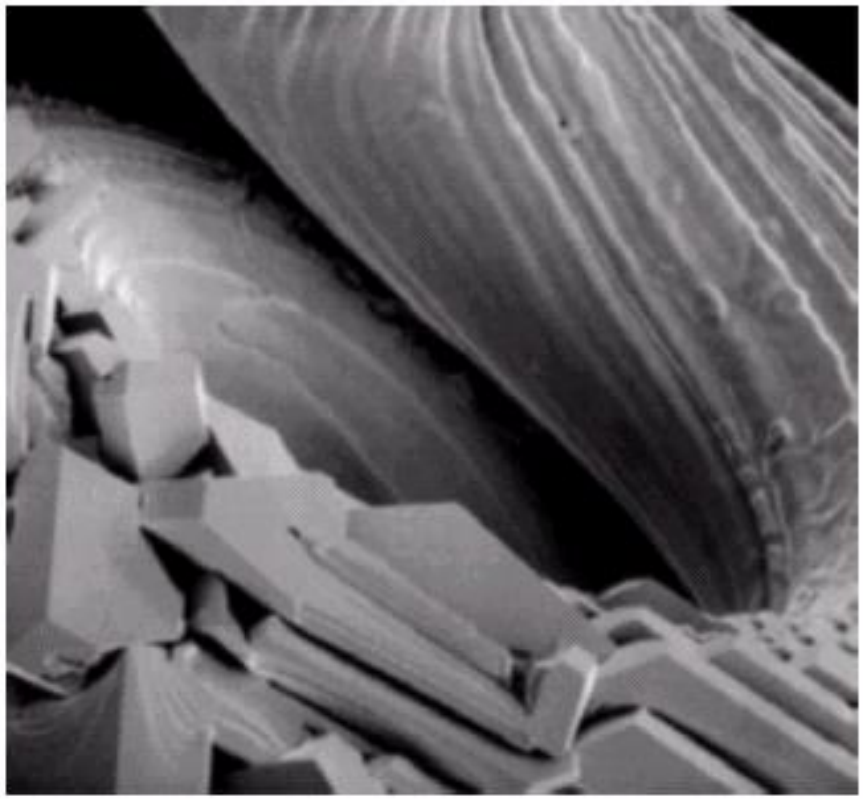
$$
\begin{aligned}
g(x,y) &= f(x,y) - \nabla^2 f \\
&= f(x,y) - [f(x+1,y) + f(x-1,y) \\
&\qquad + f(x,y+1) + f(x,y-1) \\
&\qquad - 4f(x,y)] \\
&= 5f(x,y) - f(x+1,y) - f(x-1,y) \\
&\qquad - f(x,y+1) - f(x,y-1)
\end{aligned}
$$

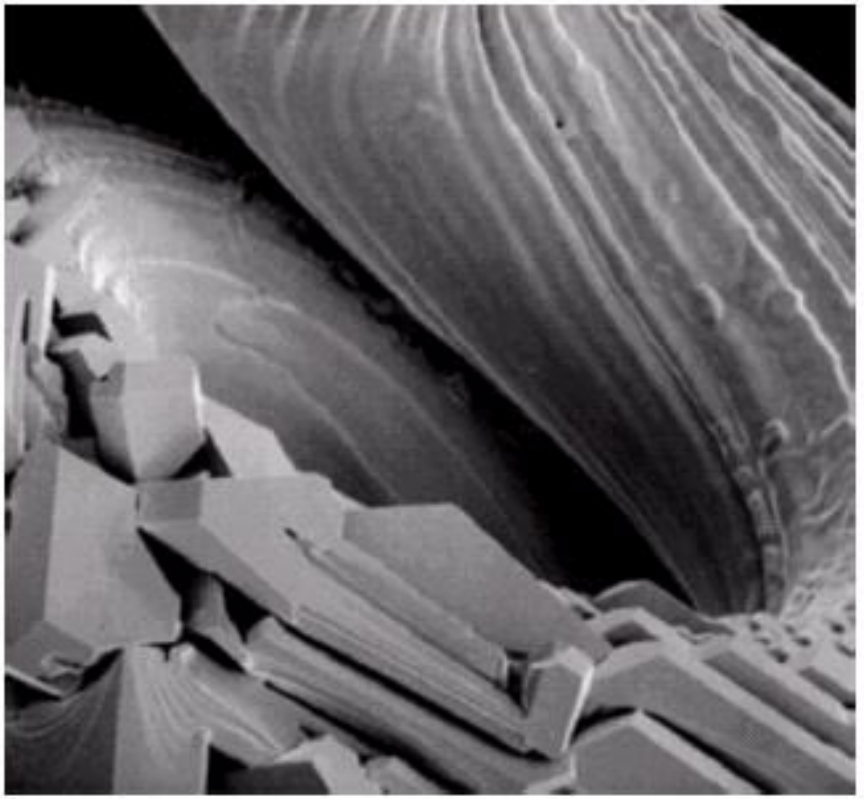
Simplified Image Enhancement (cont…)

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



Simplified Image Enhancement (cont…)





Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Variants On The Simple Laplacian

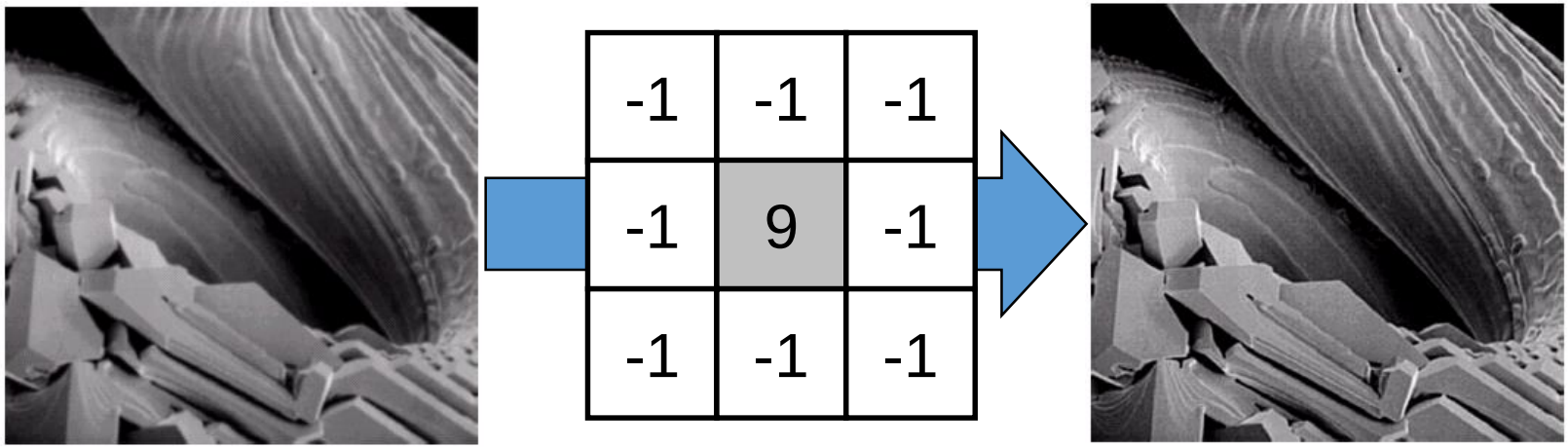
There are lots of slightly different versions of the Laplacian that can be used:

0	1	0
1	-4	1
0	1	0

Simple Laplacian

1	1	1
1	-8	1
1	1	1

Variant of Laplacian



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:

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

1st Derivative Filtering (cont…)

The magnitude of this vector is given by:

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

For practical reasons this can be simplified as:

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

1st Derivative Filtering (cont…)

There is some debate as to how best to calculate these gradients but we will use for Sobel Filters:

$$\nabla f \approx \left|\left(z_7 + 2z_8 + z_9\right) - \left(z_1 + 2z_2 + z_3\right)\right|$$
$$+ \left|\left(z_3 + 2z_6 + z_9\right) - \left(z_1 + 2z_4 + z_7\right)\right|$$

which is based on these coordinates

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

Sobel Operators

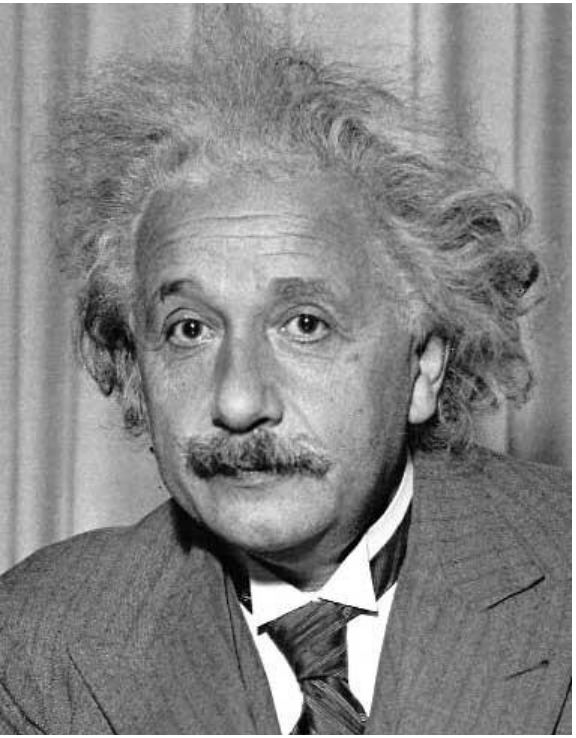
- Based on the previous equations we can derive the *Sobel Operators*

- To filter an image it is filtered using both operators the results of which are added together

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

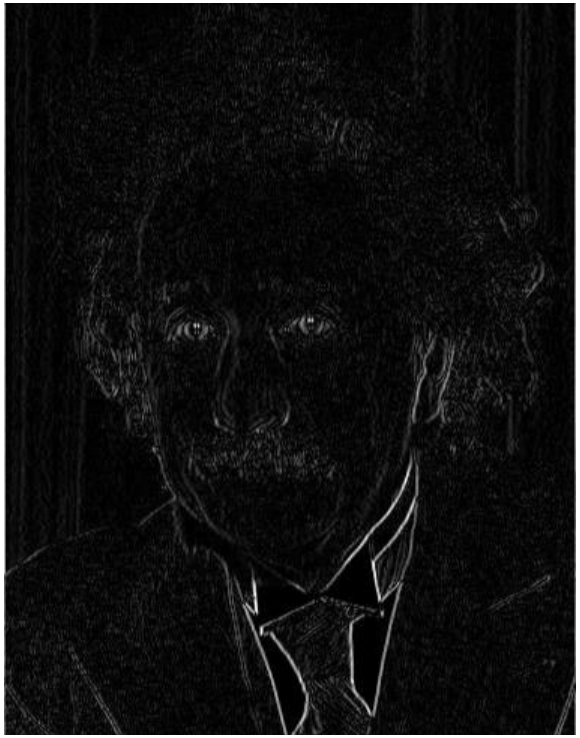
-1	0	1
-2	0	2
-1	0	1

Other filters



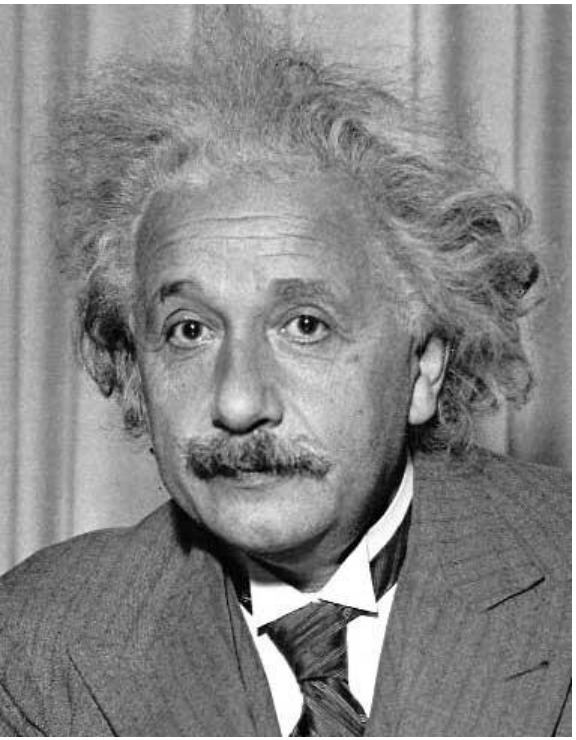
1	0	-1
2	0	-2
1	0	-1

Sobel



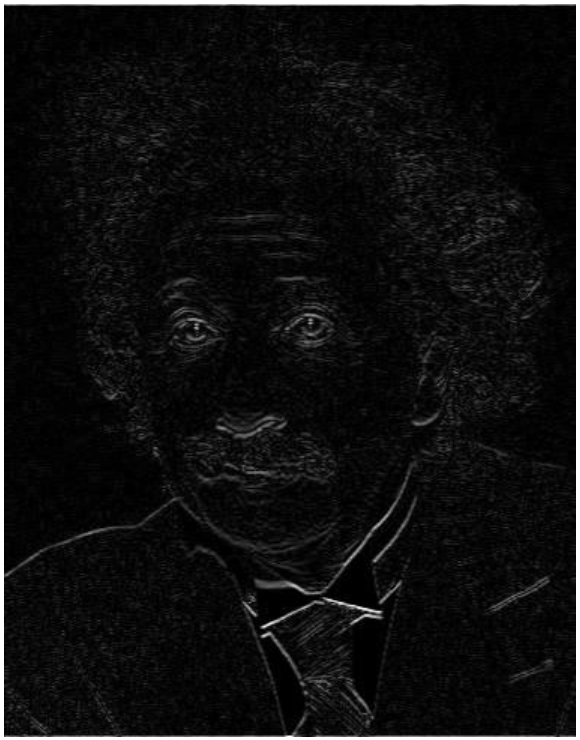
Vertical Edge
(absolute value)

Other filters



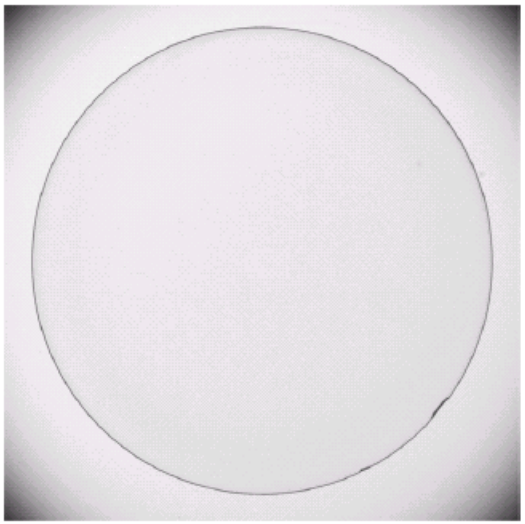
1	2	1
0	0	0
-1	-2	-1

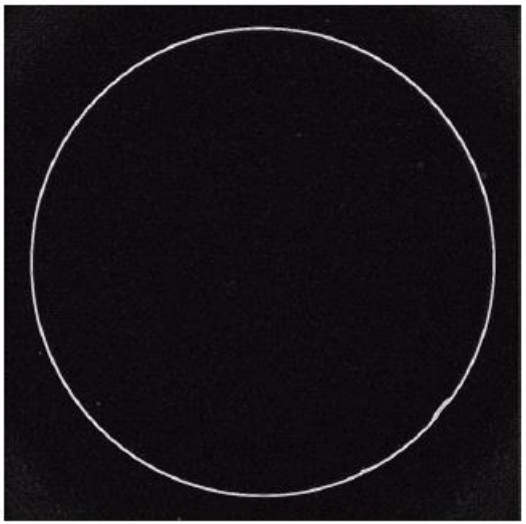
Sobel



Horizontal Edge
(absolute value)

Sobel Example





An image of a contact lens which is enhanced in order to make defects (at four and five o'clock in the image) more obvious

Sobel filters are typically used for edge detection

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

1^{st} & 2^{nd} Derivatives

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

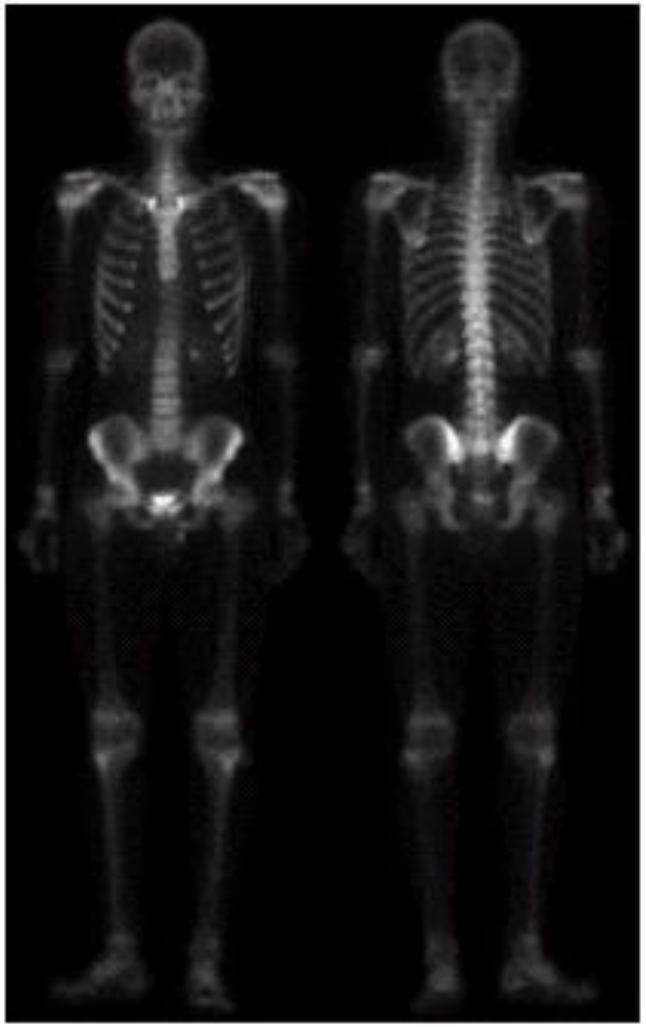
- 1^{st} order derivatives generally produce thicker edges
- 2^{nd} order derivatives have a stronger response to fine detail e.g. thin lines
- 1^{st} order derivatives have stronger response to grey level step
- 2^{nd} order derivatives produce a double response at step changes in grey level

Combining Spatial Enhancement Methods

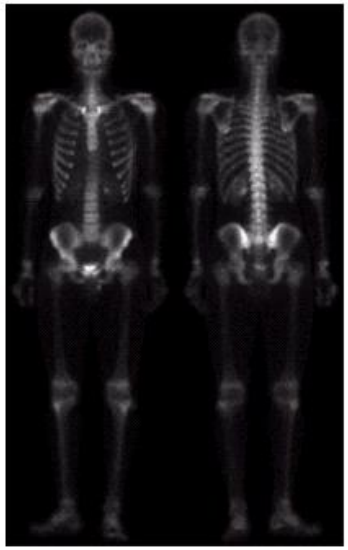
Successful image enhancement is typically not achieved using a single operation

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

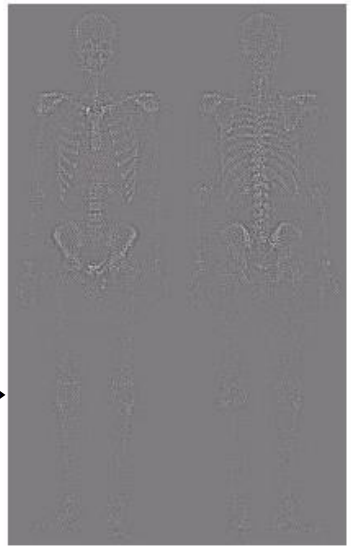


Combining Spatial Enhancement Methods (cont…)



(a)

Laplacian filter of bone scan (a)

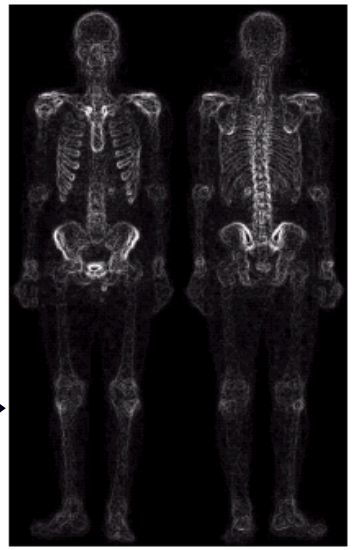


(b)

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

(c)

Sobel filter of bone scan (a)



(d)

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Combining Spatial Enhancement Methods (cont…)

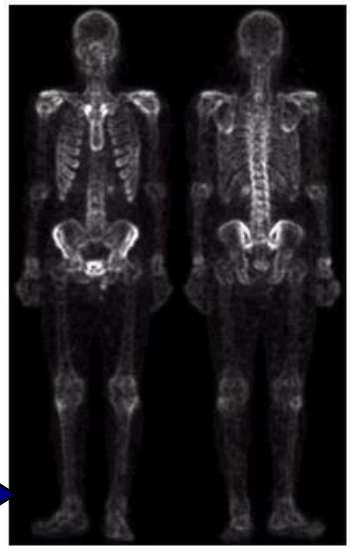
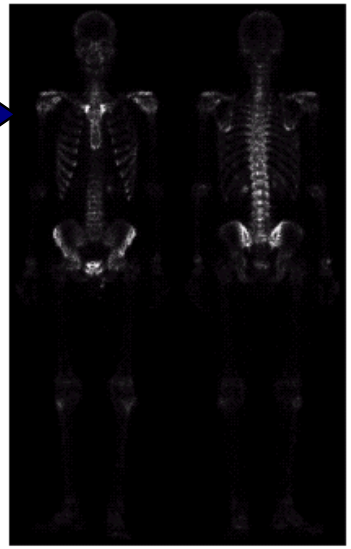


Image (d) smoothed with a 5*5 averaging filter

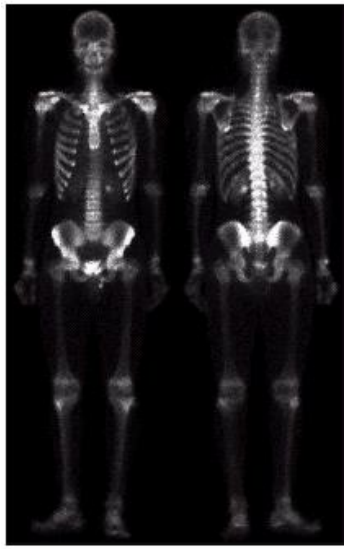
(e)

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



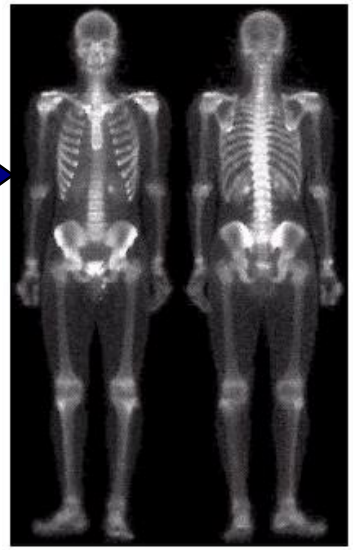
(f)

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



(g)

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

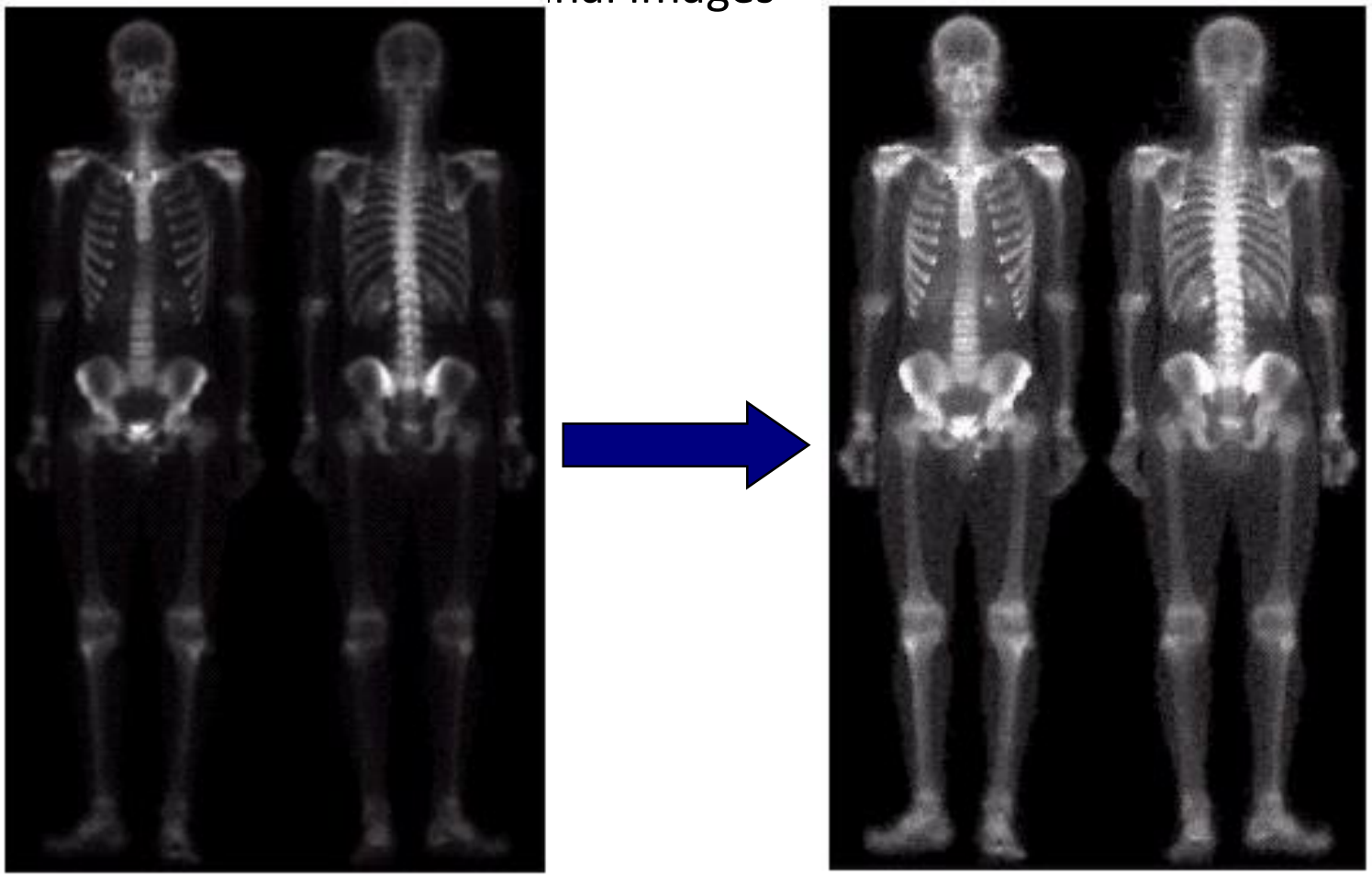


(h)

Image from Gonzalez & Woods, Digital Image Processing (2002)

Combining Spatial Enhancement Methods (cont…)

Compare the original and final images



Summary

- In this lecture we looked at:
  - Neighbourhood operations
  - What is spatial filtering?
  - Smoothing operations
  - What happens at the edges?
  - Correlation and convolution
  - Spatial filtering refresher
  - Sharpening filters
    - 1^{st} derivative filters
    - 2^{nd} derivative filters
  - Combining filtering techniques