Spatial Domain

Procedures that operate directly on pixels.

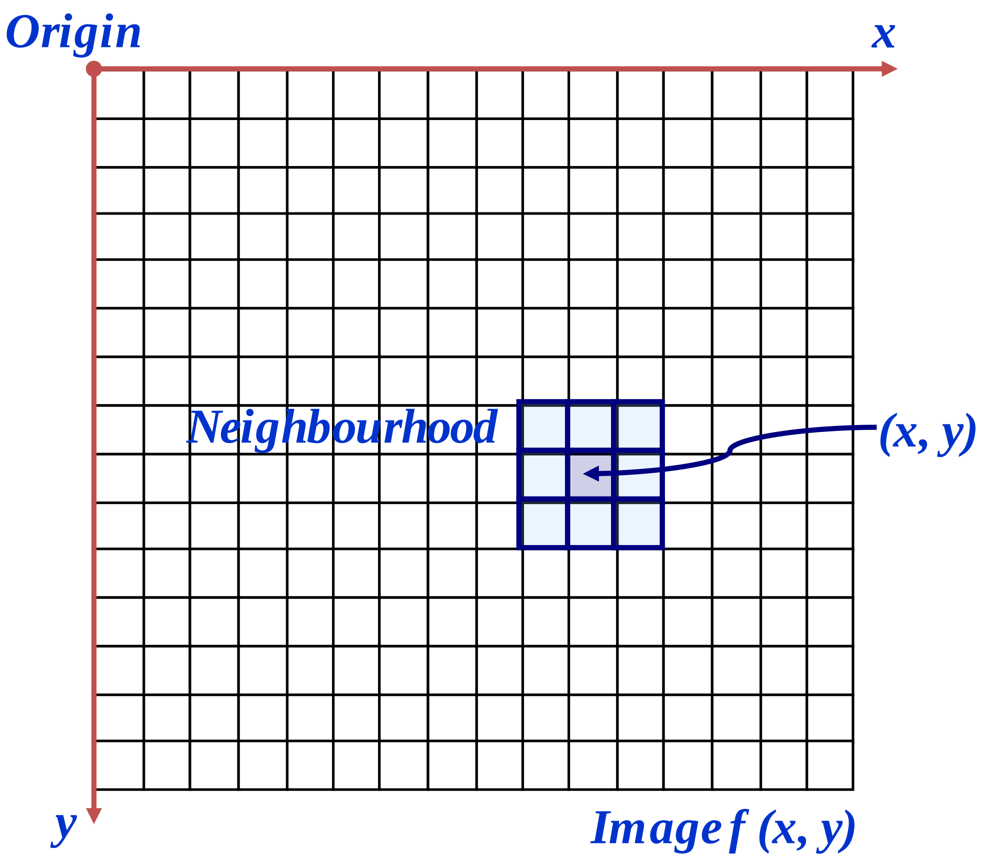
g(x,y) = T[f(x,y)]

where

f(x,y) is the input image

g(x,y) is the processed image

T which is called ***spactial filter*** is an operator on ***f*** defined over some neighborhood of ***(x,y)***



Example: Average

- Neighborhood is a square of size 3 x 3 (for example)
- T is defined as "the **average intensity** of the neighborhood."
- Assuming that the origin of the neighborhood is at its center, for the point (100,150), the result, g(100, 150), is the sum of f(100, 150) and its 8-neighbors, divided by 9
- The origin of the neighborhood is then moved to the next location and the procedure is repeated to generate the next value of the output image *g.*

Spatial filter

- The smallest possible neighborhood is of size 1 x 1.
  - Called **point processing**
  - *g* depends only on the value of *f* at a single point *(x,* y) and
  - *T* becomes an ***intensity transformation function*** (also called *gray-level* or *mapping).*
  - *S = T(r) ,* where *s* and *r* are the intensity of *g* and *f,* respectively, at any point *(x,* y).

Fundamentals of Spatial Filtering

- “Filtering" refers to accepting (passing) or rejecting certain frequency components.
- A spatial filter consists of

  (1) a ***neighborhood,*** (typically a small rectangle), and

  (2) a ***predefined operation*** that is performed on the image pixels encompassed by the neighborhood.

- Filtering creates **a new pixel with coordinates equal to the coordinates of the center of the neighborhood**, and whose value is the result of the filtering operation.
  - **Filter, mask, kernel, template or window**
- If the operation performed on the image pixels is linear, then the filter is called a ***linear*** *spatial filter.* Otherwise, the filter is ***nonlinear***.

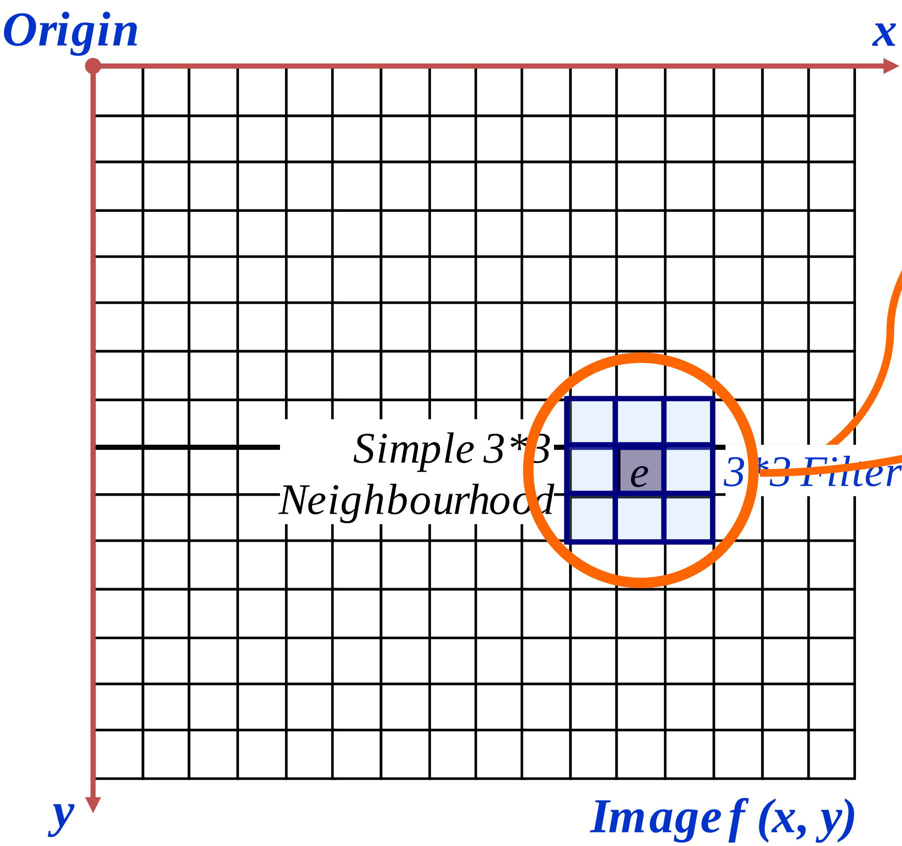
Simple Neighbourhood Operations Example

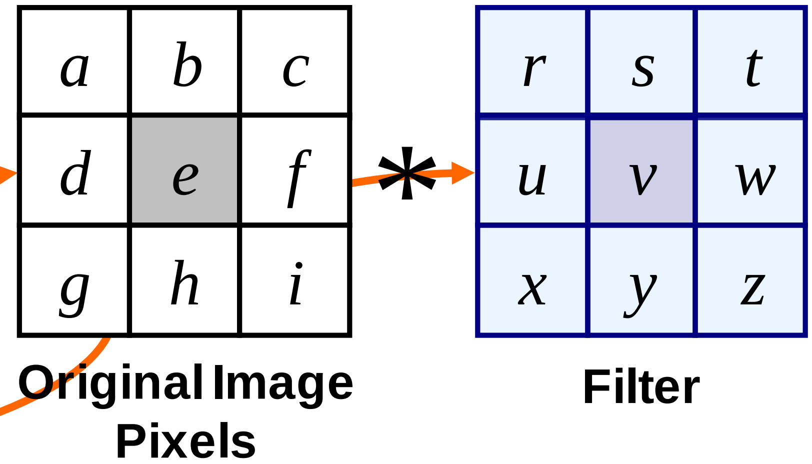
Original Image

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

Enhanced Image

The Spatial Filtering Process

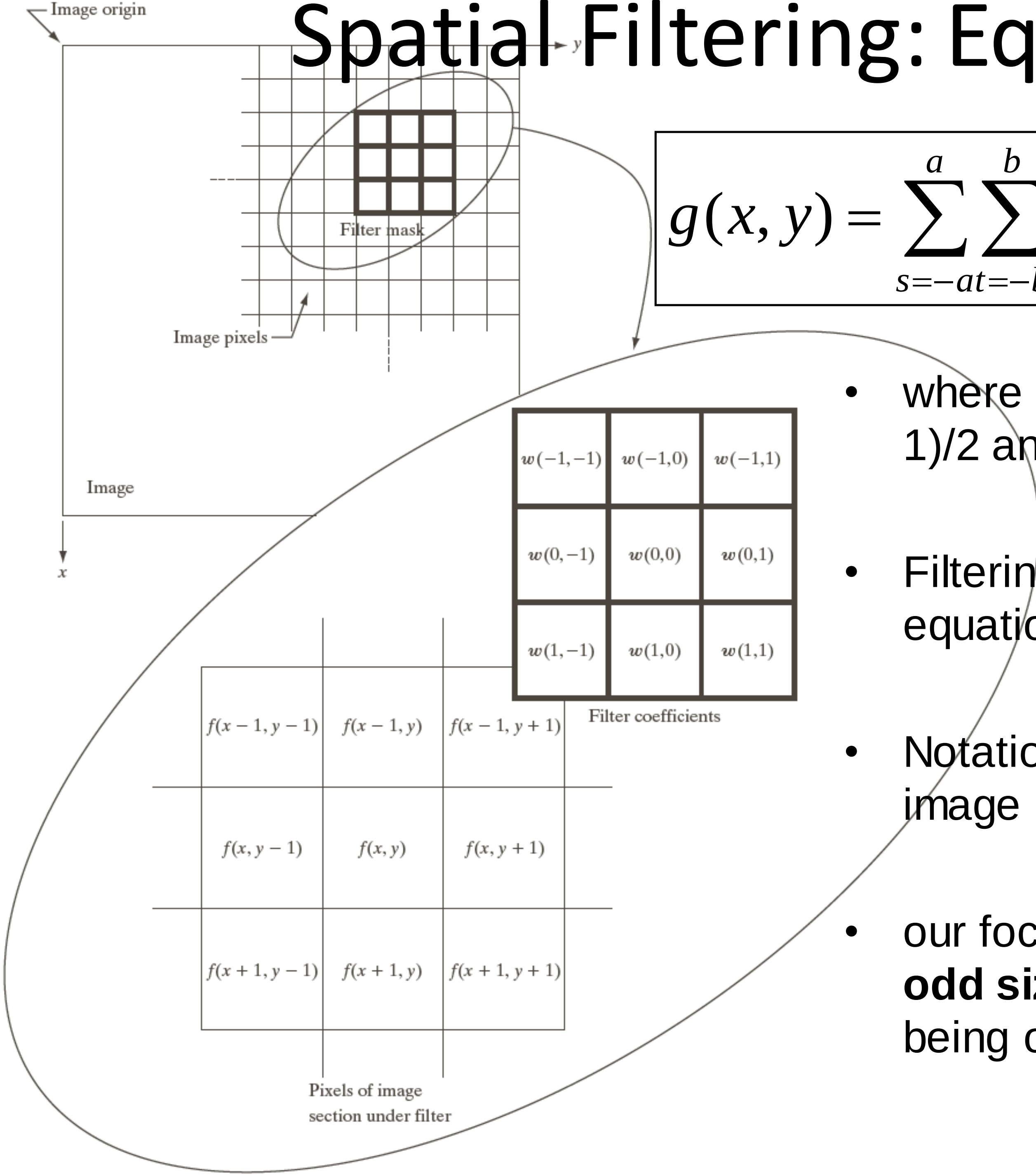




$$e_{processed} = v*e + r*a + s*b + t*c + u*d + w*f + x*g + y*h + z*i$$

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

Spatial Filtering: Equation Form



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

- where a = (m-1)/2 and b = (n-1)/2 and filter mask of size mxn
- Filtering can be given in equation form as shown above
- Notations are based on the image shown to the left
- our focus will be on **filters of odd size**, with the smallest being of size **3 X 3**.

Smoothing Spatial Filters (Linear)

- One of the simplest spatial filtering operations

- Simply average all of the pixels in a neighbourhood around a central value

- Reducing the rapid **pixel-to-pixel variation** in gray values

- **Sharp transitions**
  - Random noise
  - Edges

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

Simple averaging filter

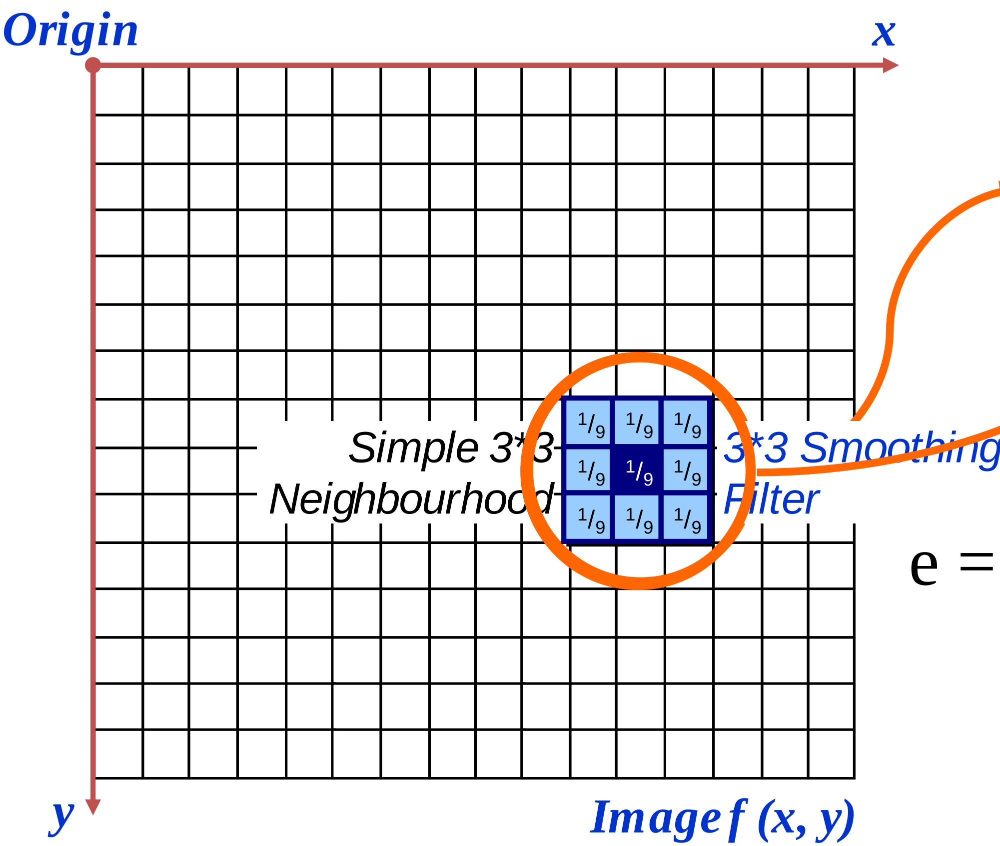
Smoothing Spatial Filters (Linear)

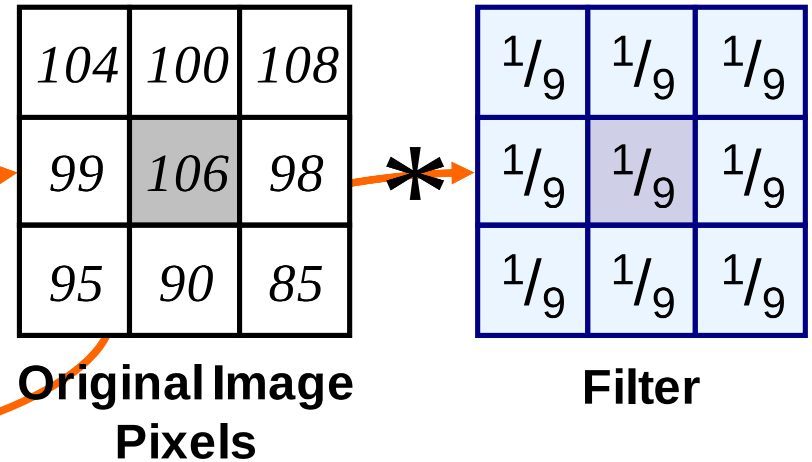
- Effect:
  - **Noise reduction**
  - **Blur edges**

- Blurring (in pre-processing)
    - **Highlighting gross details**
      - By removing the small ones using a big mask
    - **Bridging small gaps**
    - **Losing sharp details**

- **Averaging filter** or **low-pass filter**

Smoothing Spatial Filtering





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

- The above is repeated for every pixel in the original image to generate the smoothed image
- The result is an image with **reduced "sharp" transitions** in intensities.

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

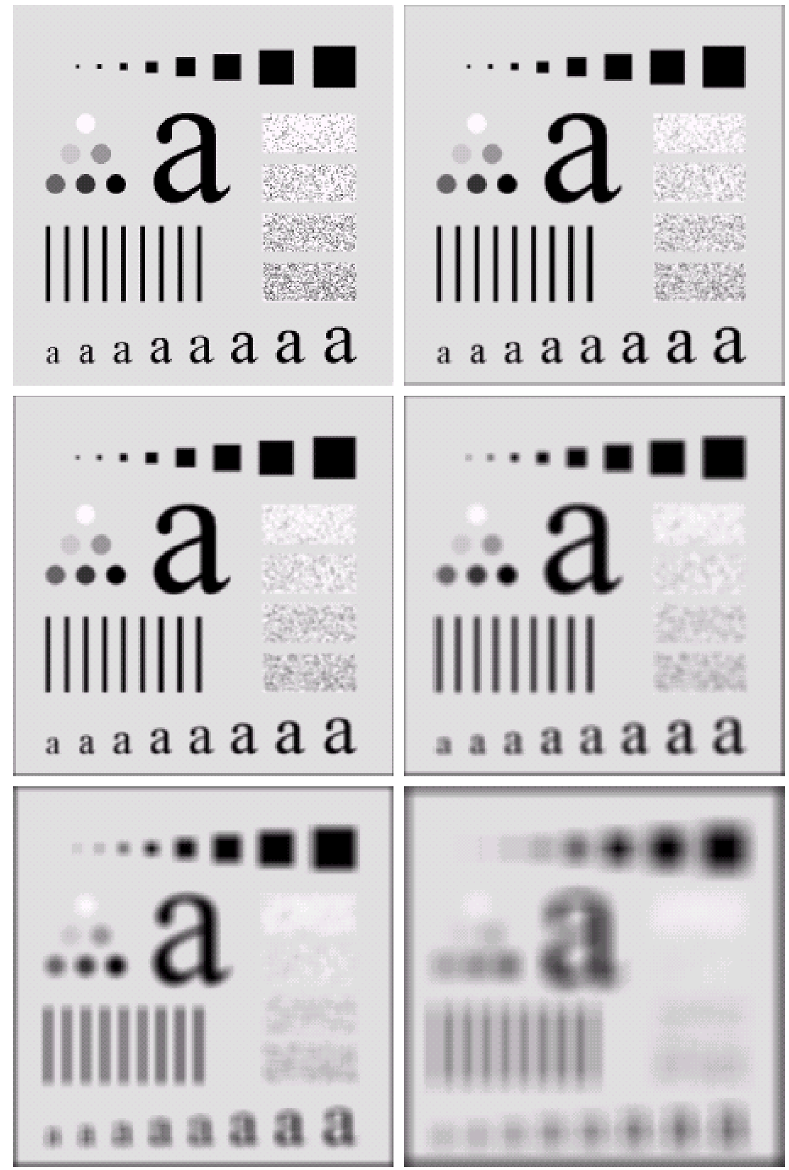


Image Smoothing Example

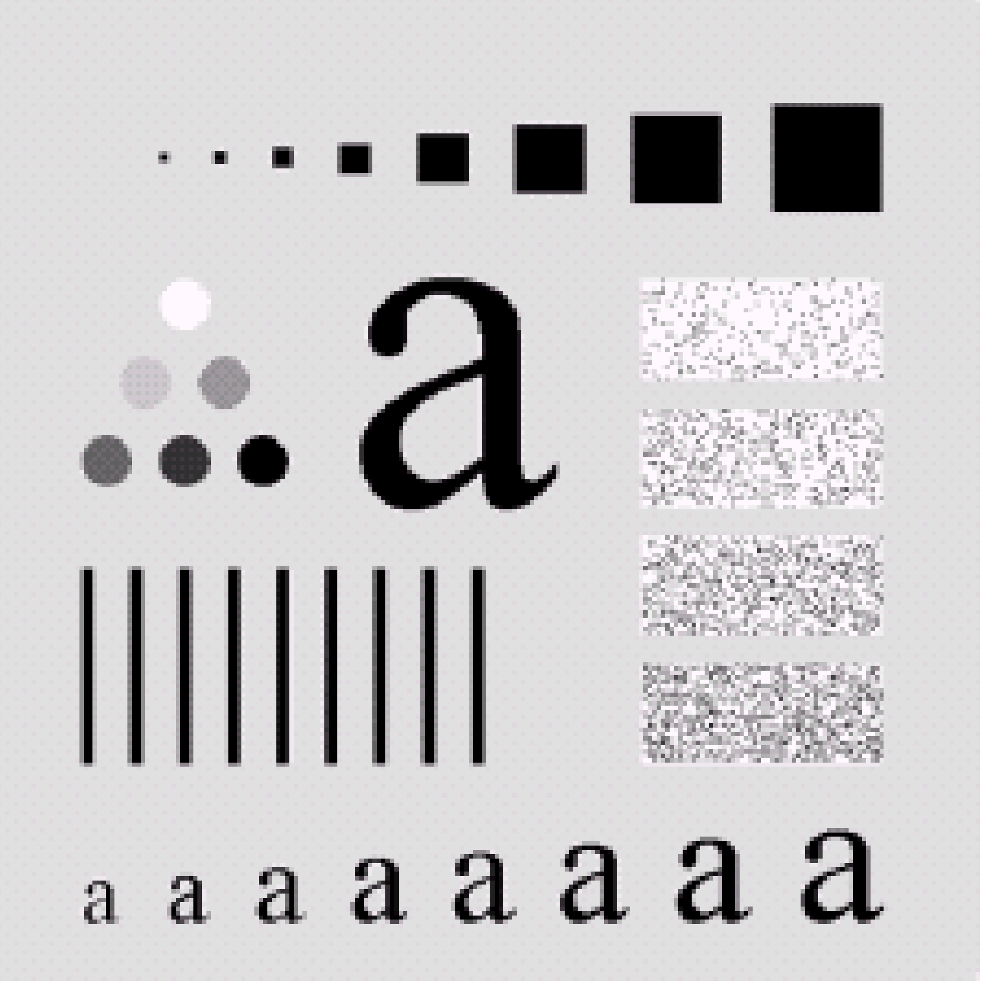


Image Smoothing Example

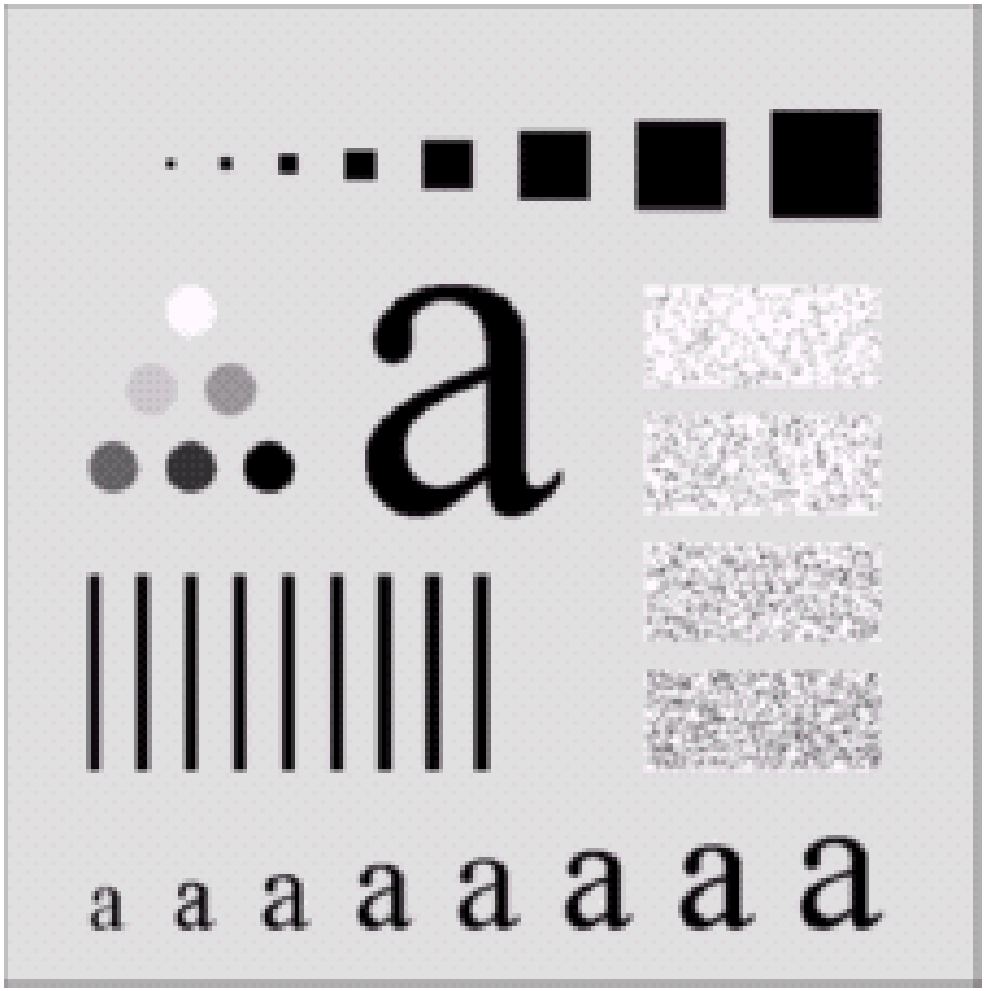


Image Smoothing Example

Image Smoothing Example

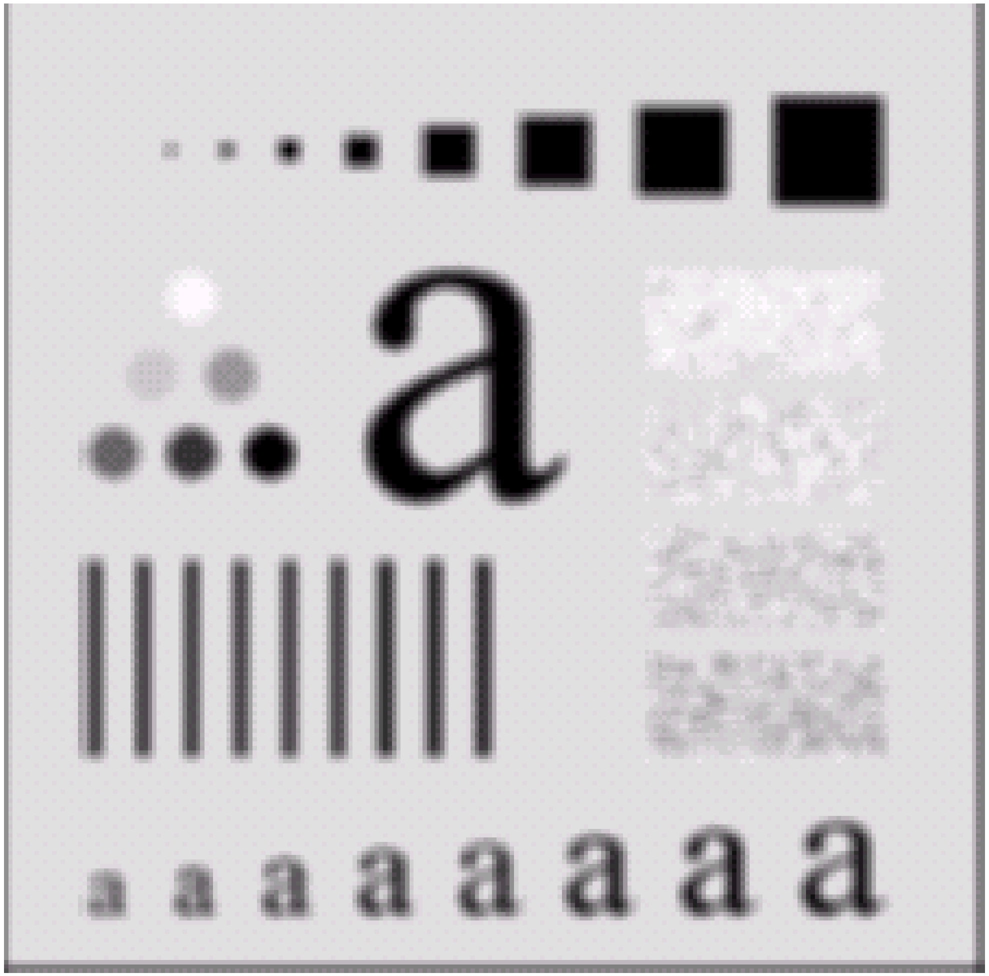


Image Smoothing Example

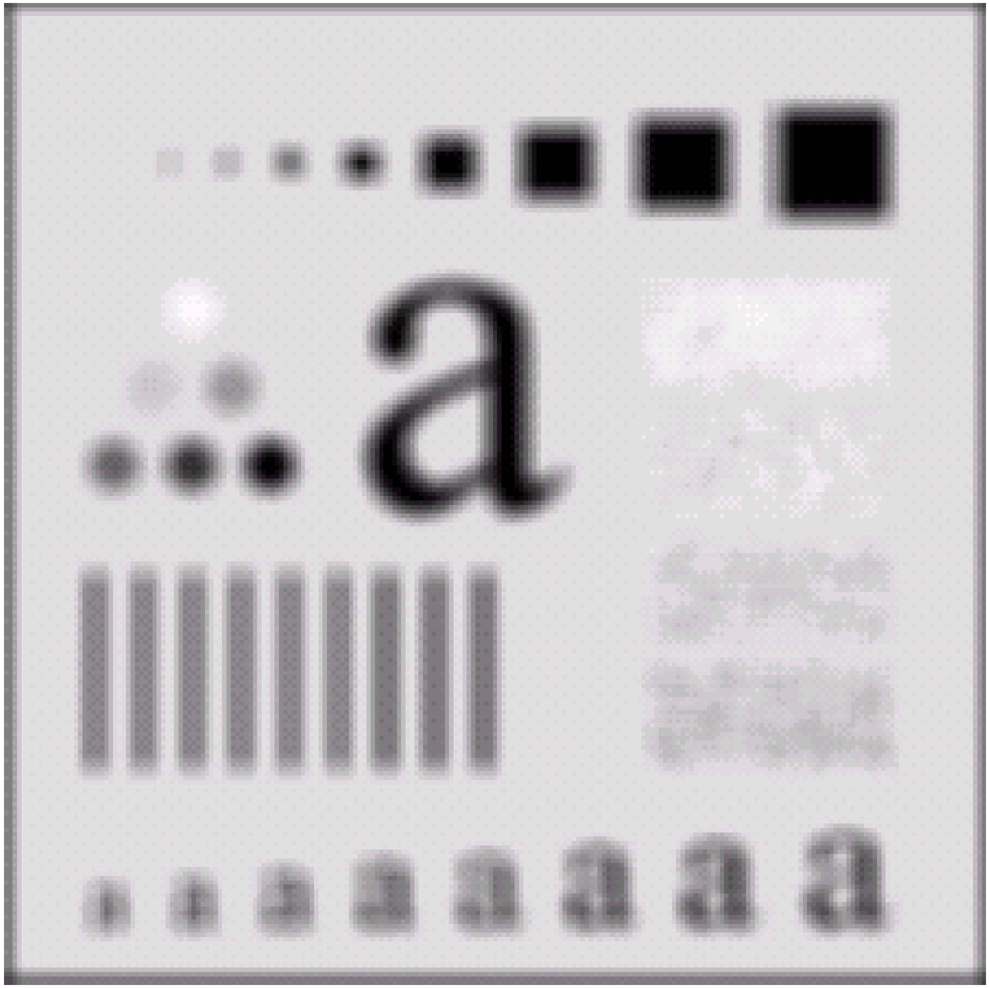
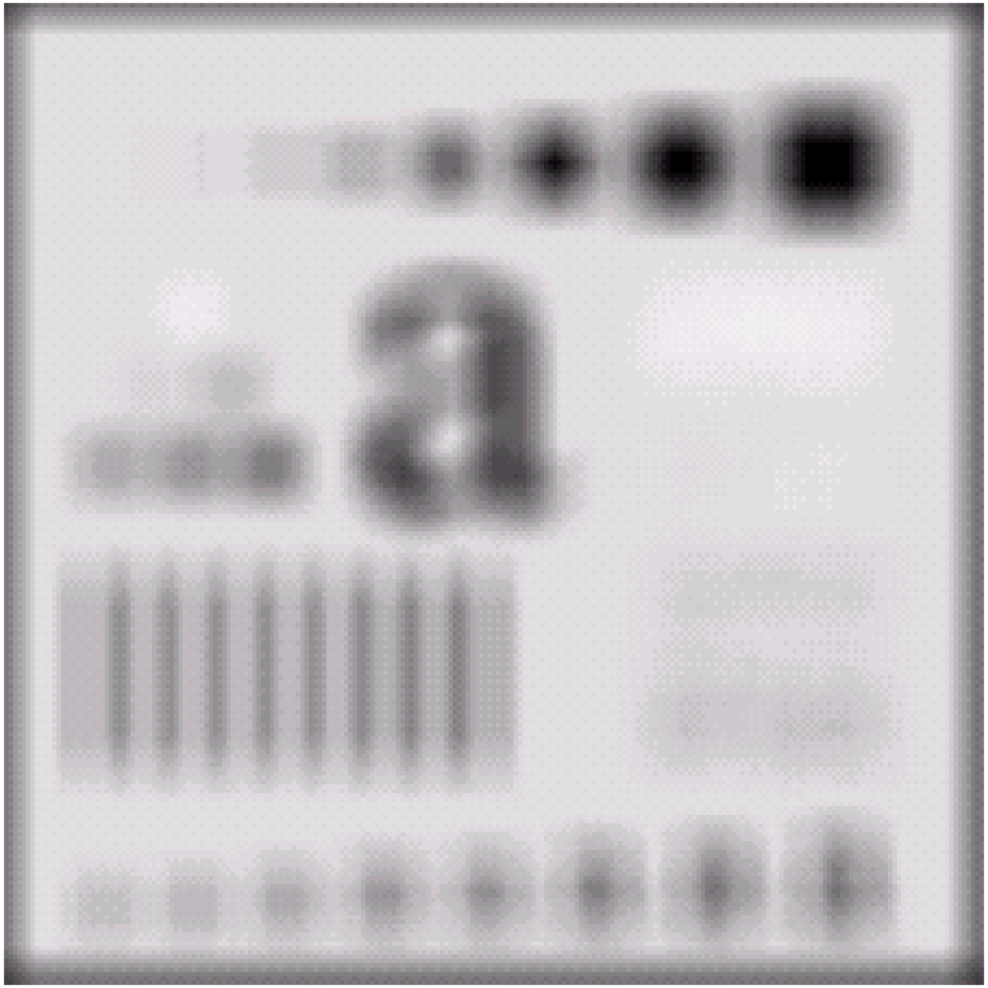
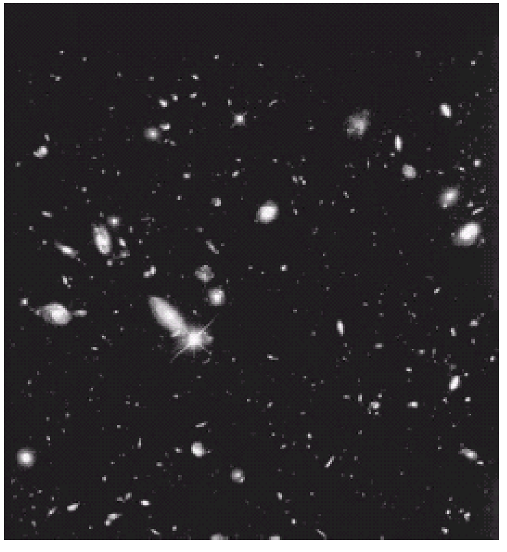


Image Smoothing Example

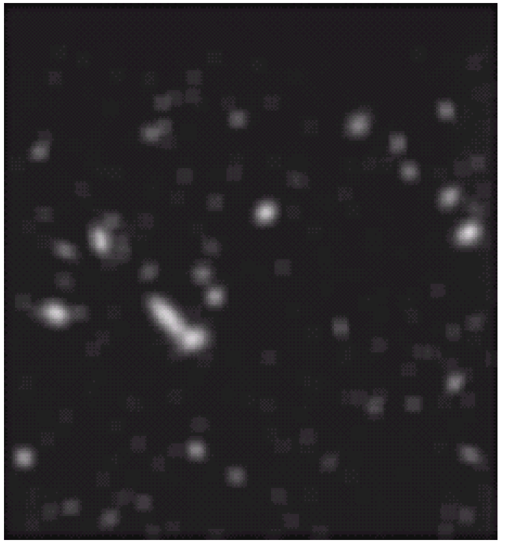


Another Smoothing Example

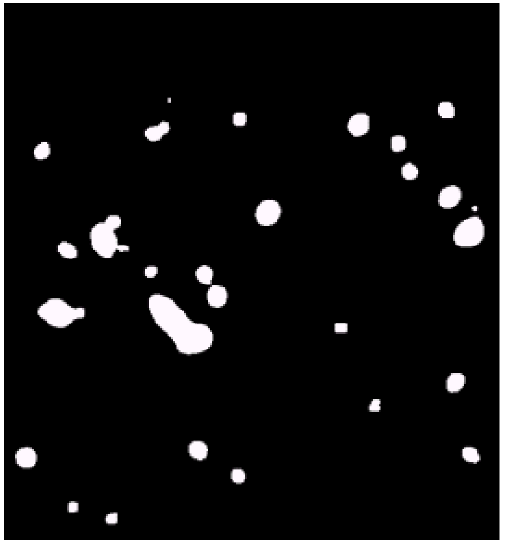
- By smoothing the original image we get rid of lots of the finer detail which leaves only the gross features for thresholding



Original Image



Smoothed Image
with 15x15 averaging filter



Thresholded Image

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

Order-Statistics Filters (Non-linear)

- Based on ordering ordering the pixels within the mask
  - Replace the center pixel with the result of a ranking operation

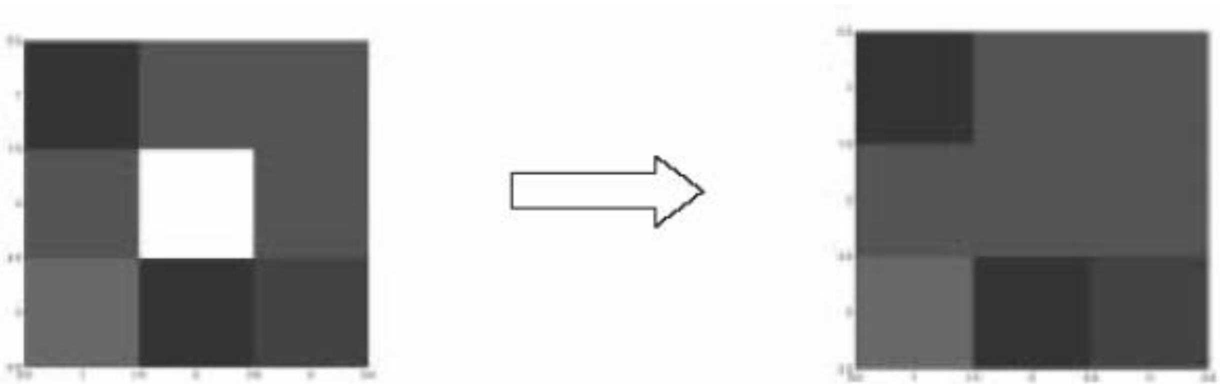
- Examples:
  - Median filter
  - Maximum filter
  - Minimum filter

Median Filter

- Useful for random noise
  - Impulse noise or salt and pepper noise
- Less blurring than averaging filter
- Example:
  - [0,0,1,1,1,1,1,2,7]
  - Replace 7 with 1

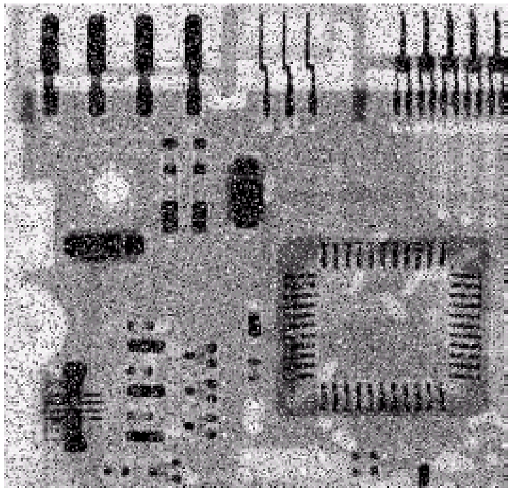
0	1	1
1	7	1
2	0	1

Input image



خفف ال noise

Averaging Filter Vs. Median Filter Example



Original Image With Noise

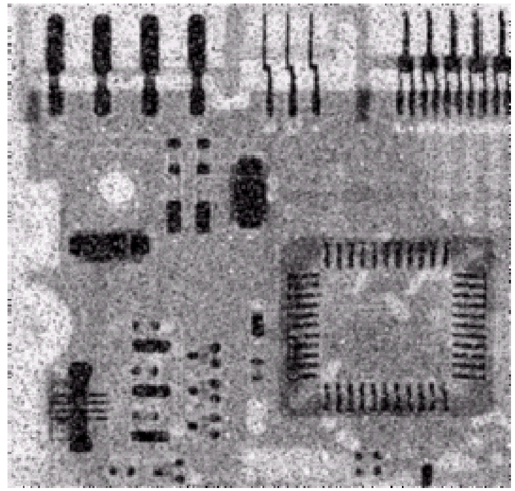


Image After Averaging Filter

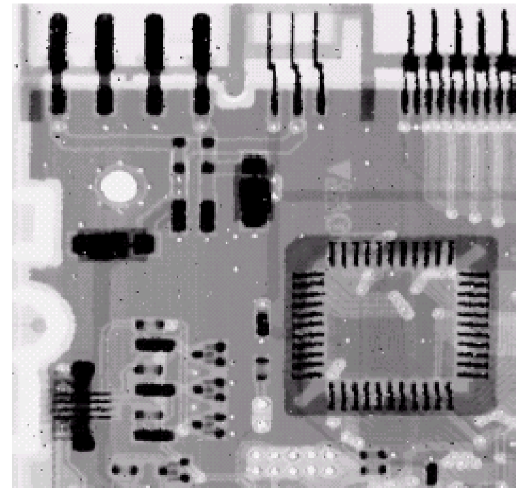
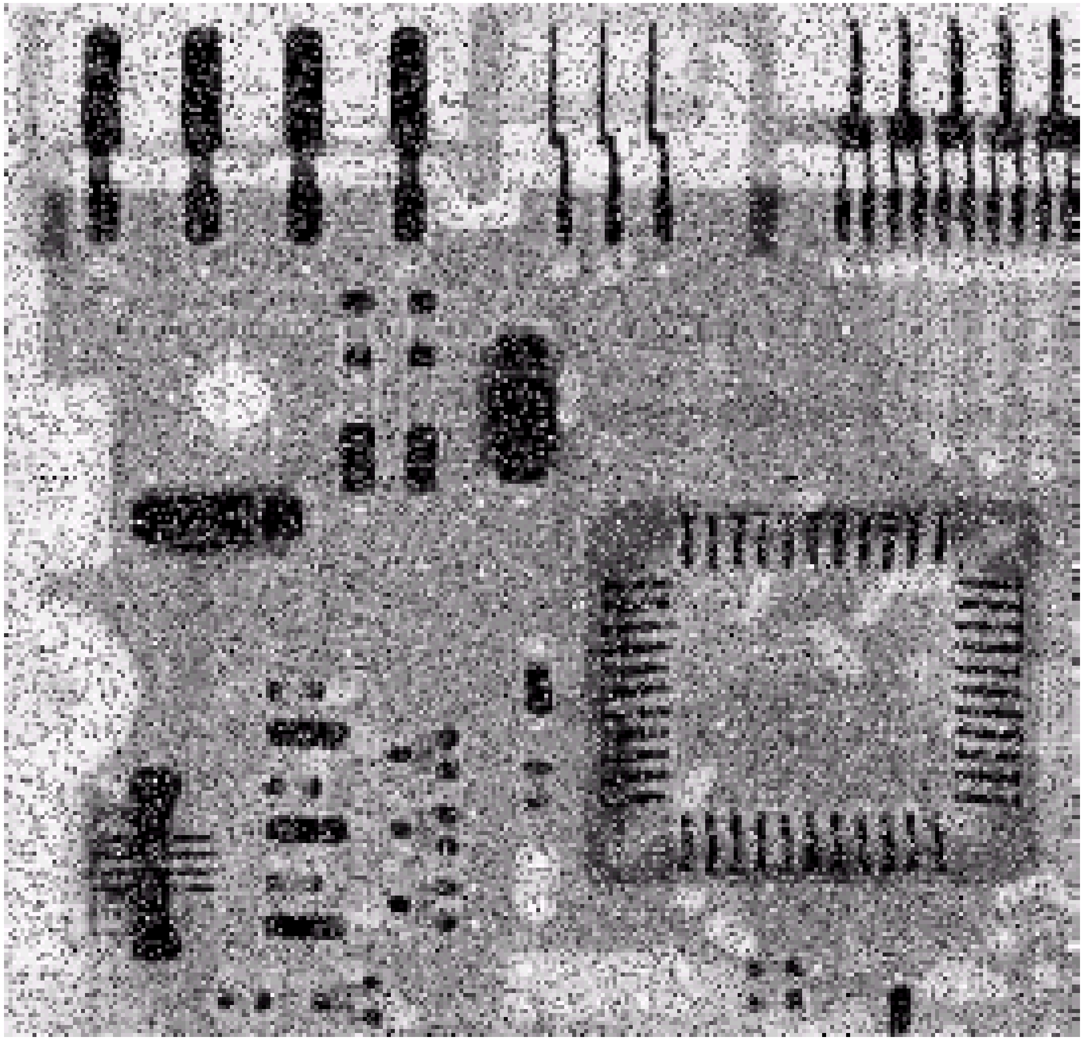


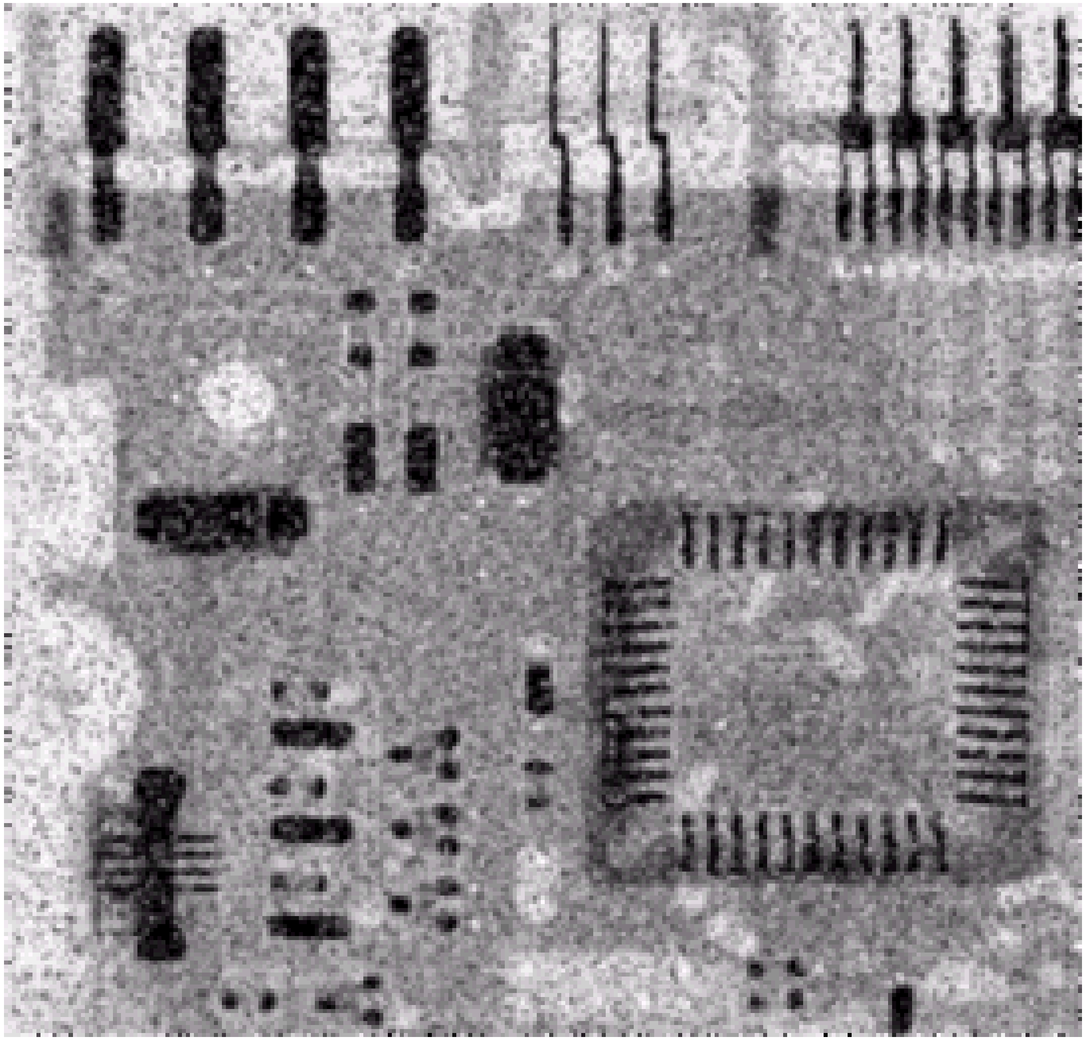
Image After Median Filter

- The median ξ, of a set of values, is the value ξ such that half of the values is less than of equal ξ and the other half is greater or equal ξ
- Filtering is often used to remove noise from images
- Sometimes a median filter works better than an averaging filter

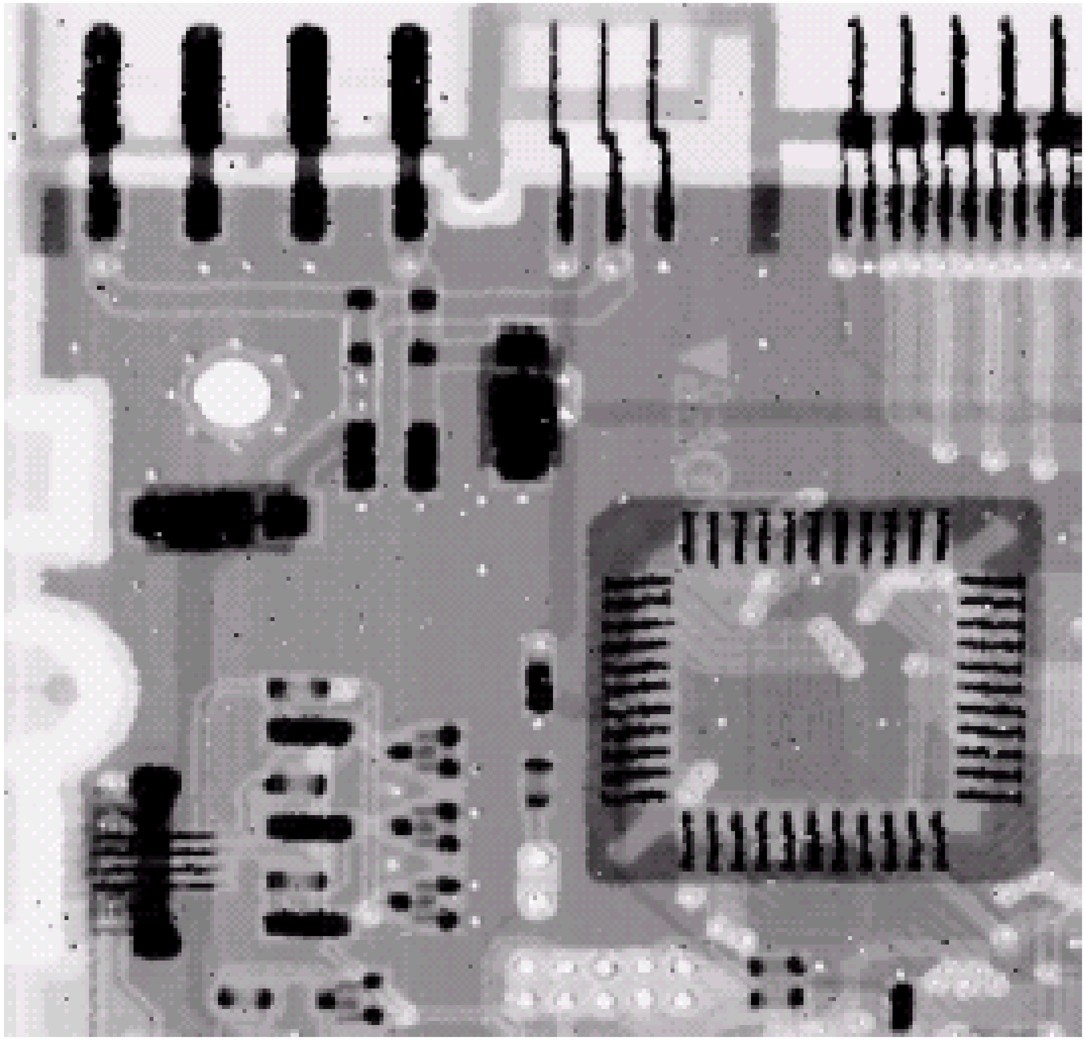
Original Image



After Averaging Filter



After Median Filter

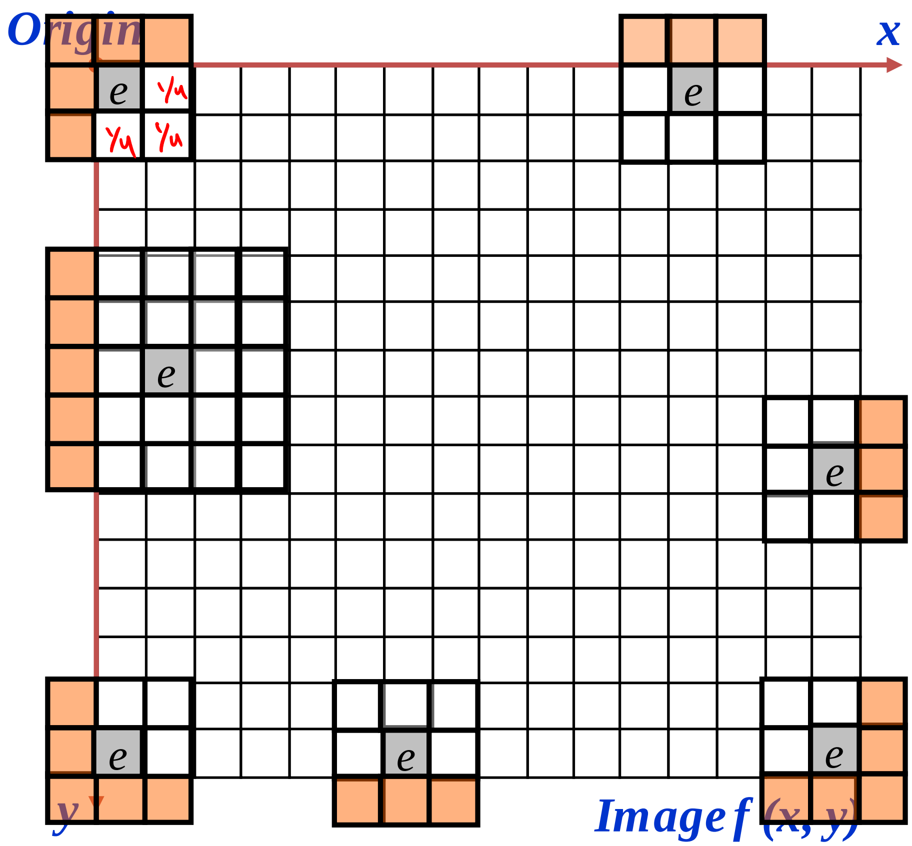


Simple Neighbourhood Operations Example

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

Strange Things Happen at the Edges!

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

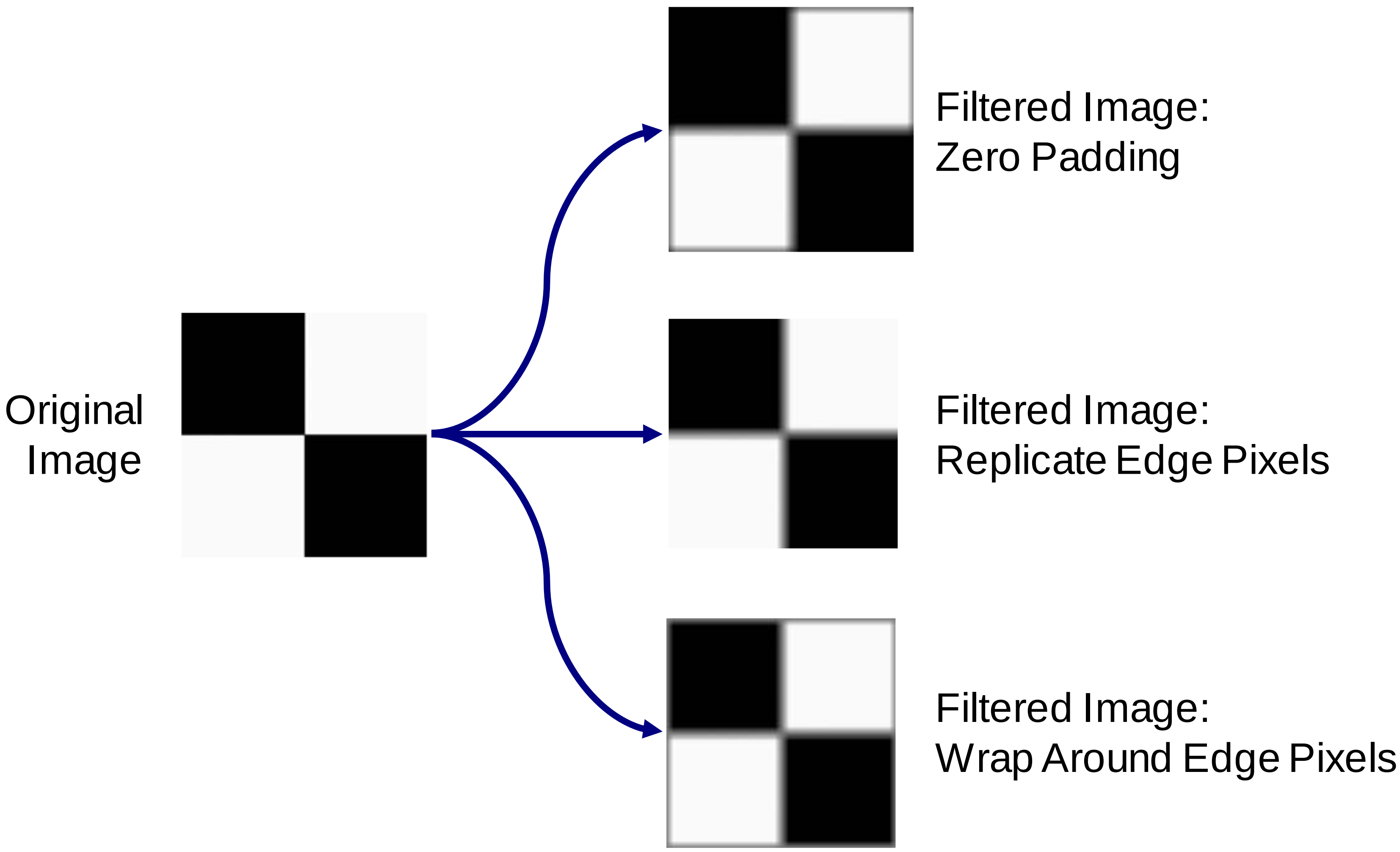


Strange Things Happen At The Edges! (cont…)

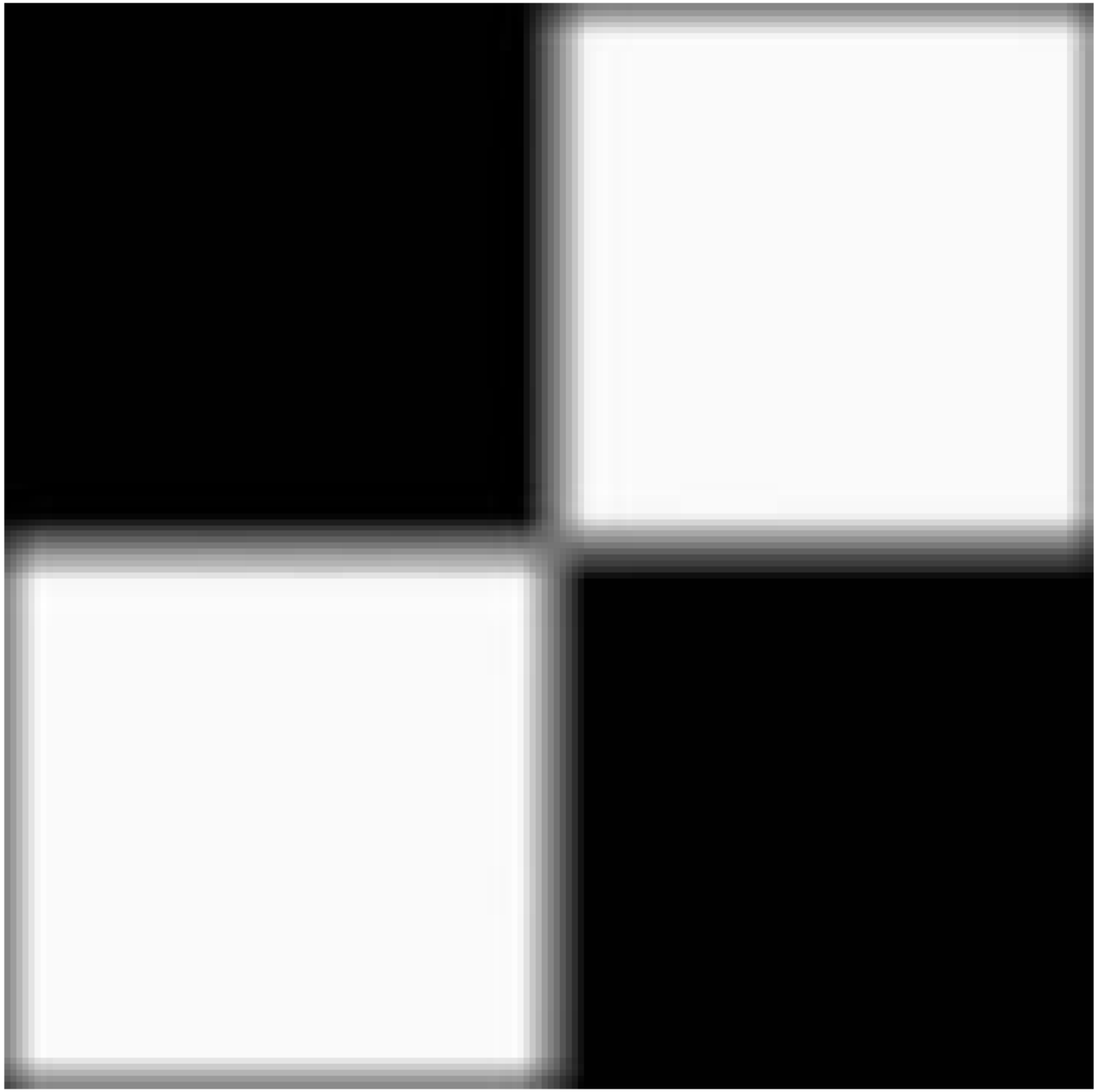
- There are few approaches to dealing with missing edge pixels:
  - Omit missing pixels
    - Only works with some filters
    - Can add extra code and slow down processing
  - Pad the image
    - Typically with either all white or all black pixels
  - Replicate border pixels
  - Truncate the image
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

can be used with patterns

Strange Things Happen At The Edges! (cont…)



Zero Padding



filter: averaging

edges از بین [illegible]

+ zero padding

Replicate Edge Pixels

Wrap Around Edge Pixels

Sharpening Spatial Filters

cont. integration

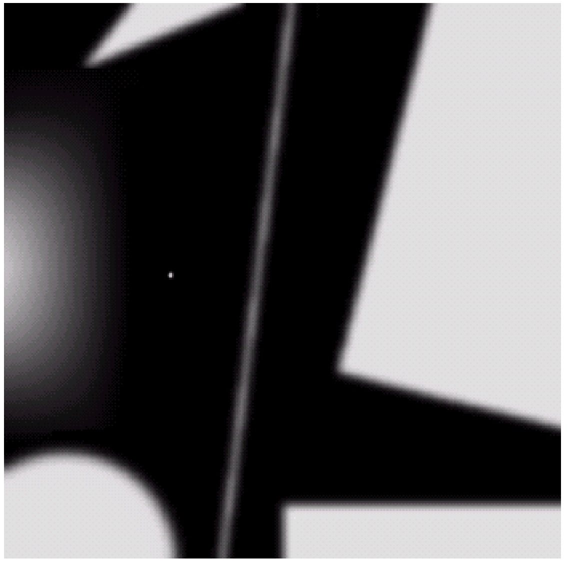
- Previously we have looked at smoothing filters which remove fine detail

discrete summation

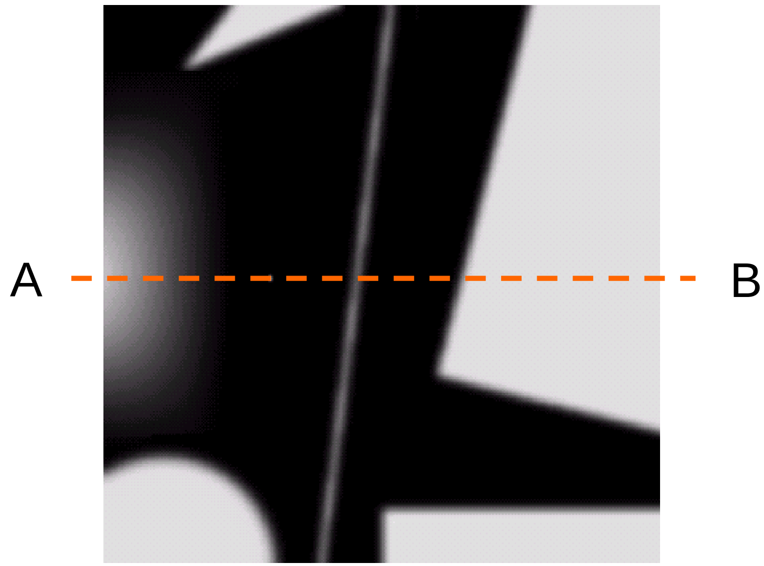
- ***Sharpening*** *spatial filters* seek to **highlight fine detail**
  - Enhance or reduce blurring from images
  - Highlight edges
- Sharpening filters are based on *spatial differentiation*

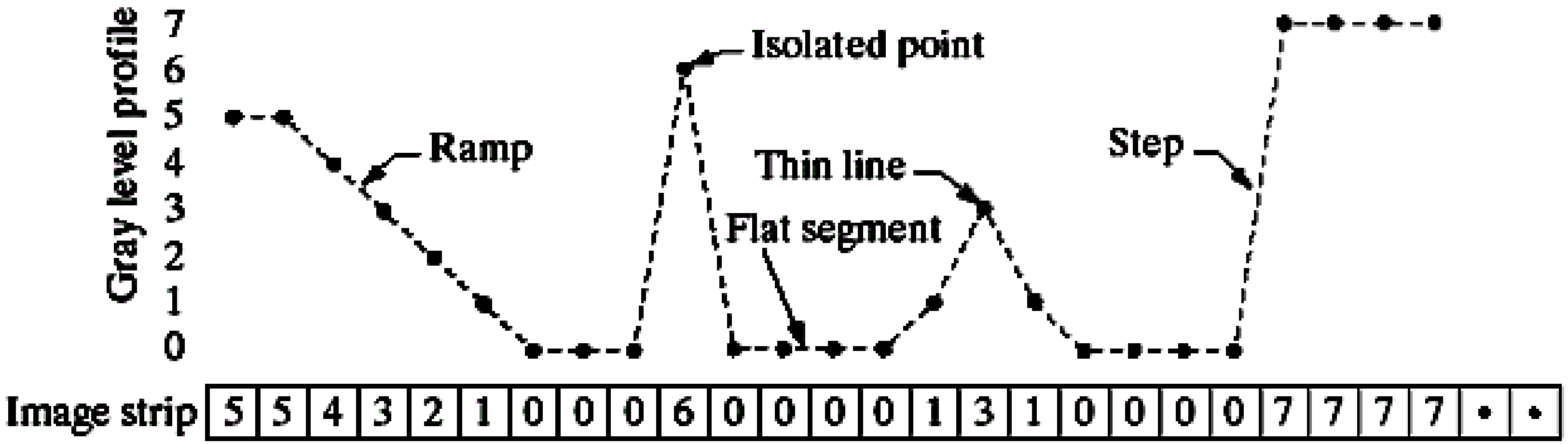
Spatial Differentiation

- Differentiation measures the *rate of change* of a function
- Let’s consider a simple 1 dimensional example



Spatial Differentiation





1^{st} Derivative

The formula for the 1^{st} 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

2^{nd} Derivative

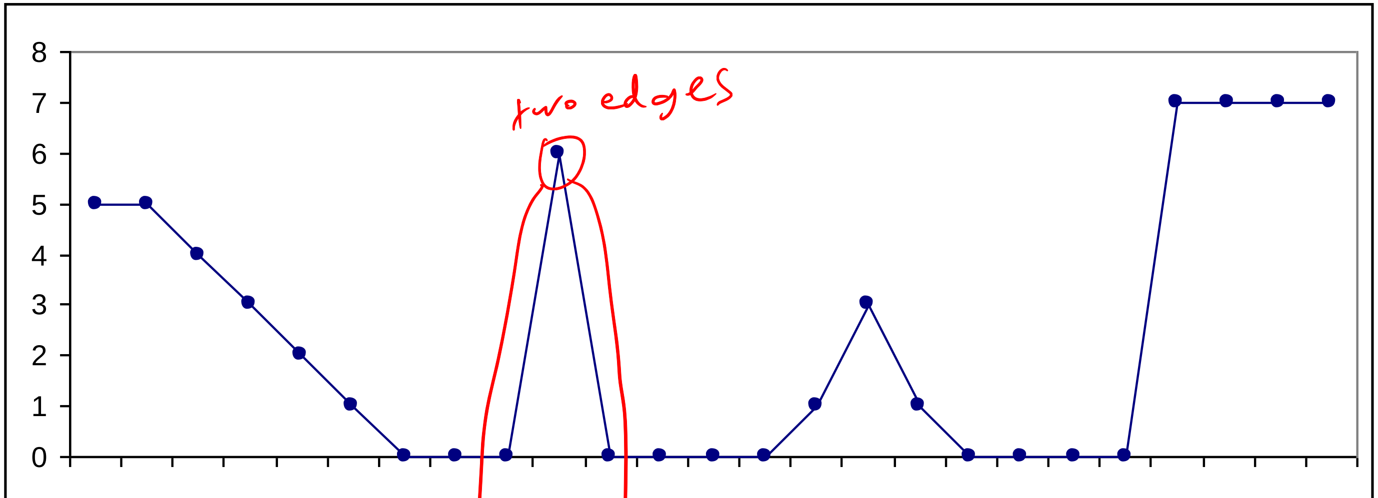
The formula for the 2^{nd} derivative of a function is as follows:

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

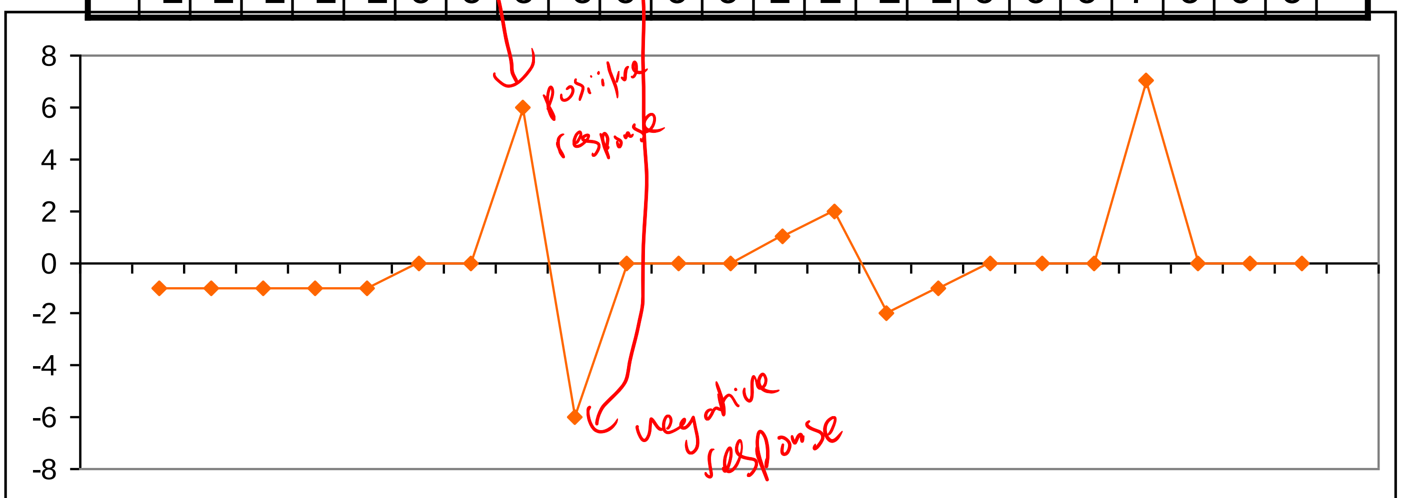
Simply takes into account the values both before and after the current value

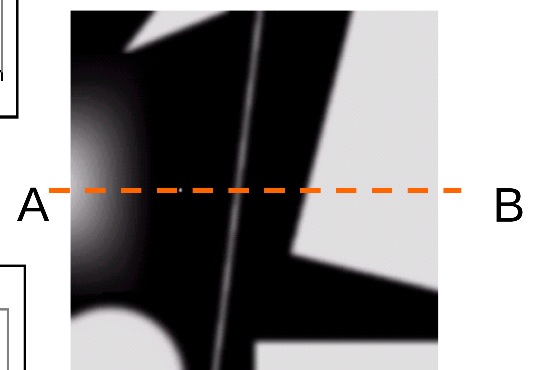
1^{st} Derivative (cont...)

In constant regions output is zero (black) there's no variation

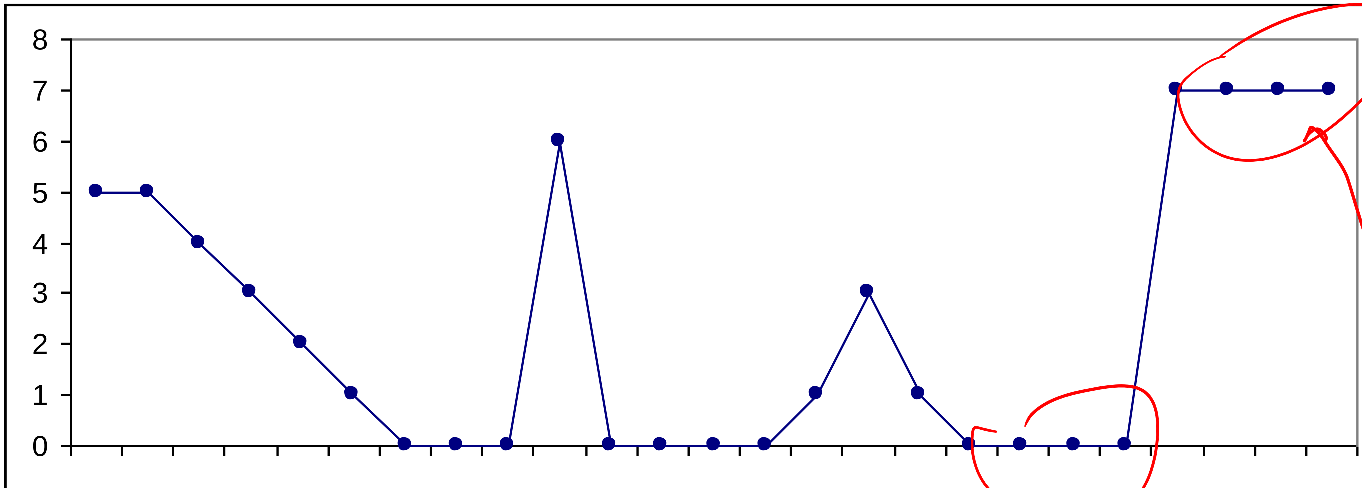


5	5	4	3	2	1	0	0	0	6	0	0	0	0	1	3	1	0	0	0	0	7	7	7	7
	-1	-1	-1	-1	-1	0	0	6	-6	0	0	0	1	2	-2	-1	0	0	0	7	0	0	0	



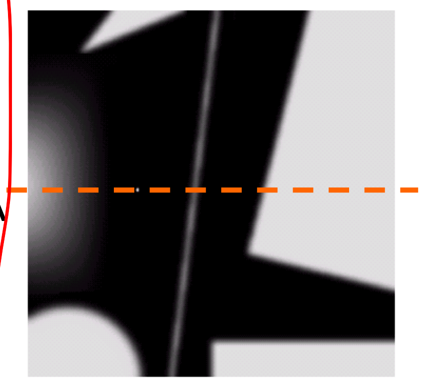


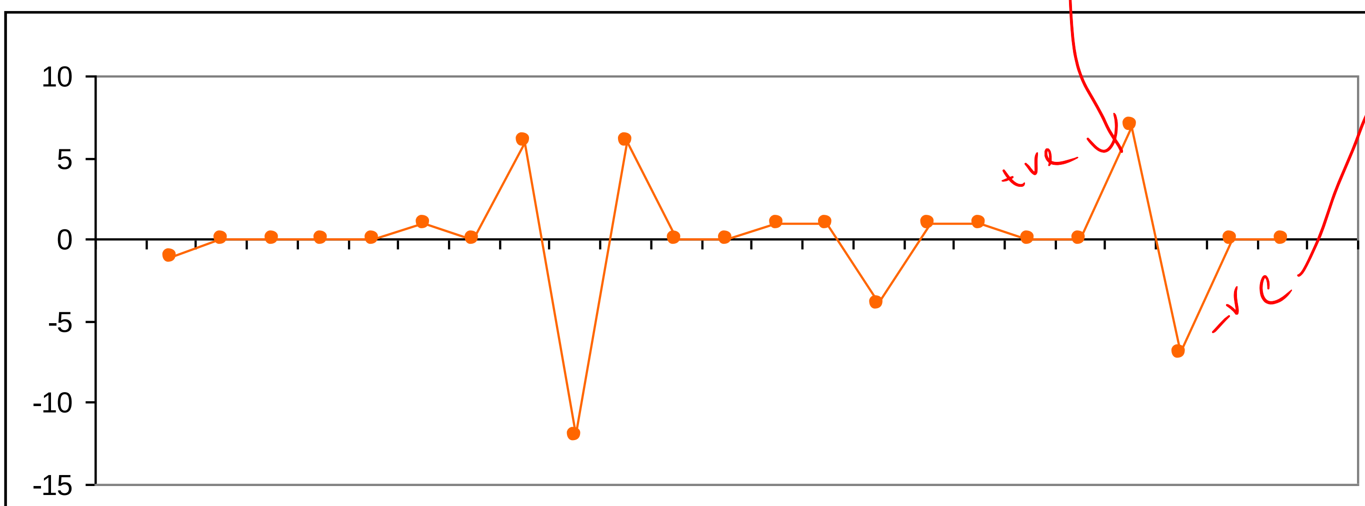
2nd Derivative (cont...)



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

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

A  B



Derivatives definition

- The derivatives of a digital function are defined in terms of **differences**.
- There are various ways to define these differences. However, we require that any definition we use for a *first derivative*

  (1) must be **zero** in areas of **constant intensity**;

  (2) must be **nonzero** at the onset of an **intensity step or ramp**; and

  (3) must be **nonzero along ramps**. بداية

- Similarly, any definition of a *second derivative*

  (1) must be **zero** in **constant areas**; ثابت intensity

  (2) must be **nonzero** at the **onset *and* end of an intensity step or ramp**;

  (3) must be **zero** along **ramps of constant slope**.

- Because we are dealing with digital quantities whose values are finite, the maximum possible intensity change also is finite, and the shortest distance over which that change can occur is between adjacent pixels.

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*

- 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…)

So, the Laplacian can be given as follows:

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

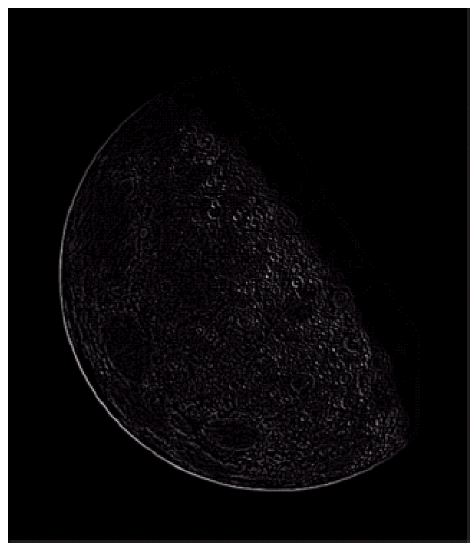
We can easily build a filter based on this

0	1	0
1	-4	1
0	1	0

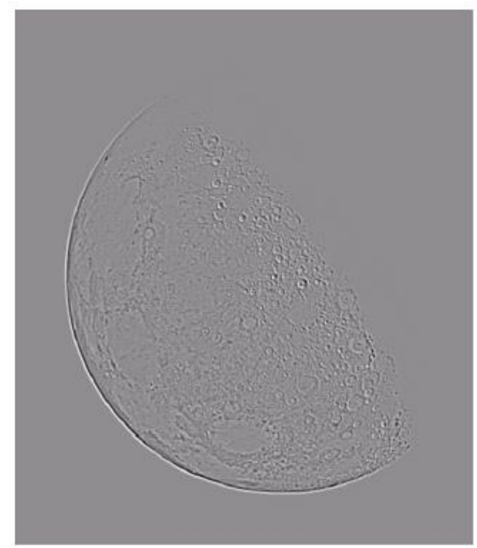
The Laplacian (cont…)

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

Original Image



Laplacian Filtered Image



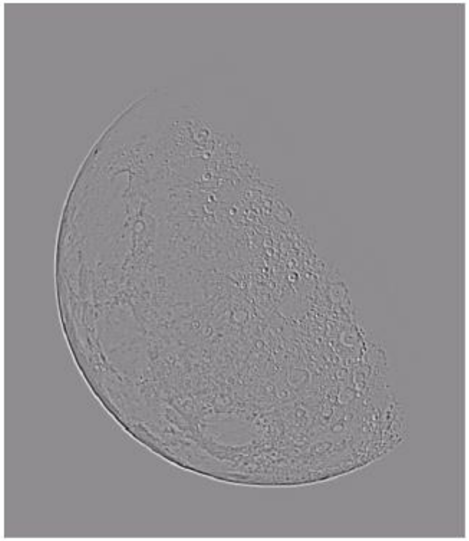
Laplacian Filtered Image Scaled for Display

The Laplacian (cont…)

- Highlights gray-level discontinuities
- Deemphasizes regions with slowly varying graylevels
- Isotropic filter: response is independent of
- Direction
  - rotation-invariant

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

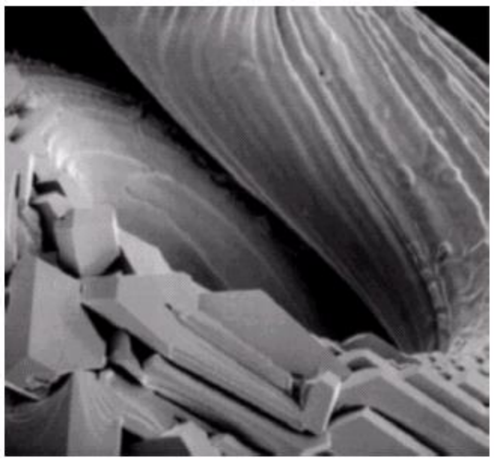
Simplified Image Enhancement

The entire enhancement can be combined into a single filtering operation

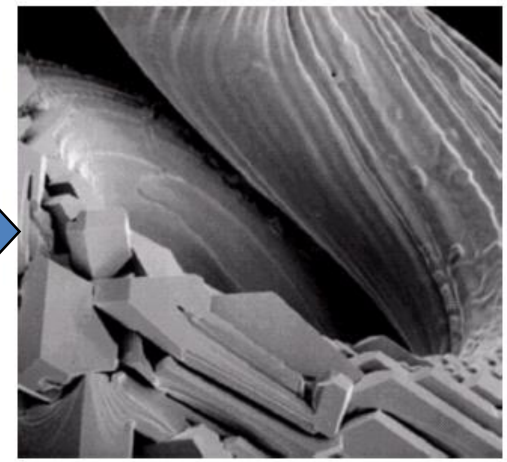
$$
\begin{aligned}
g(x,y) &= f(x,y) - \nabla^2 f \\
&= f(x,y) - [f(x+1,y) + f(x-1,y) \\
&\qquad + f(x,y+1) + f(x,y-1) \\
&\qquad - 4f(x,y)] \\
&= 5f(x,y) - f(x+1,y) - f(x-1,y) \\
&\qquad - f(x,y+1) - f(x,y-1)
\end{aligned}
$$

Simplified Image Enhancement (cont…)

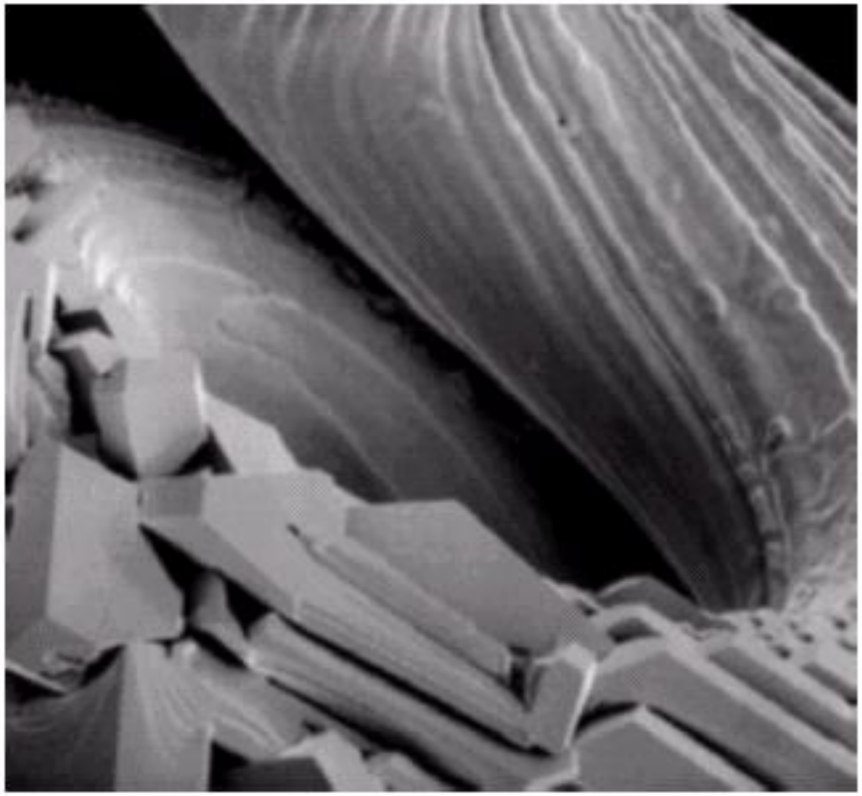
This gives us a new filter which does the whole job for us in one step

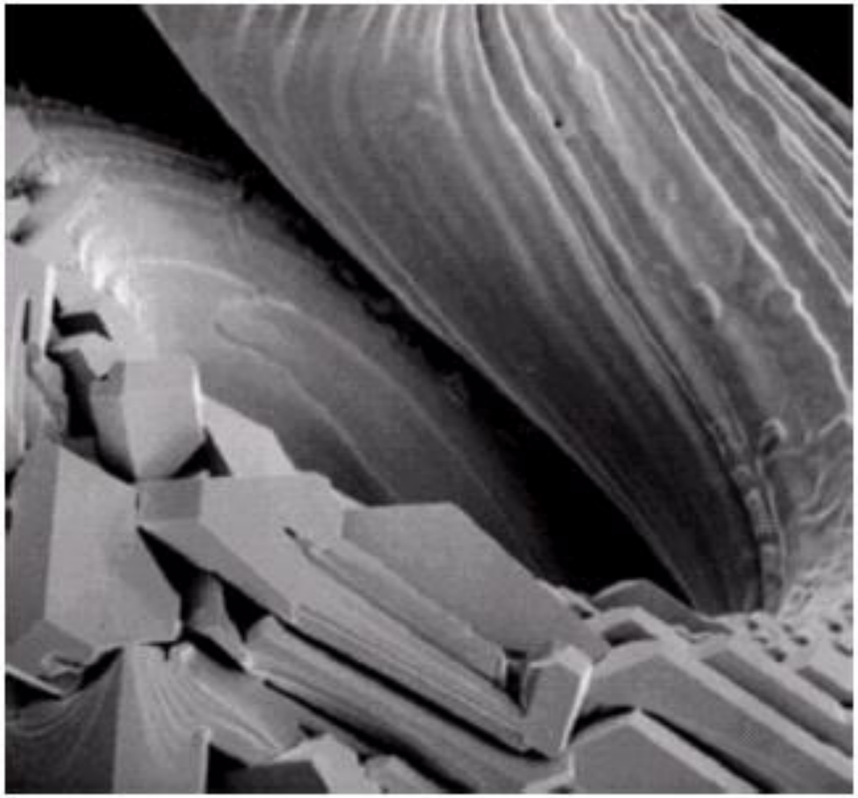


0	-1	0
-1	5	-1
0	-1	0



Simplified Image Enhancement (cont…)





Variants On The Simple Laplacian

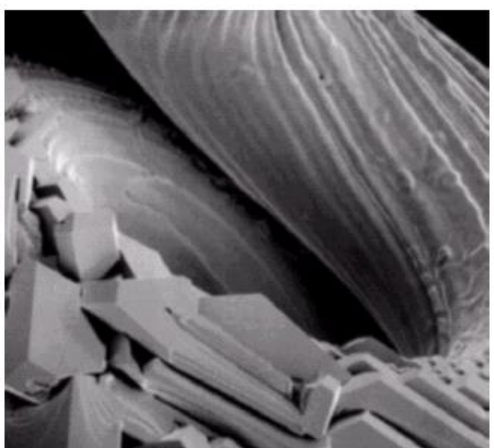
There are lots of slightly different versions of the Laplacian that can be used:

0	1	0
1	-4	1
0	1	0

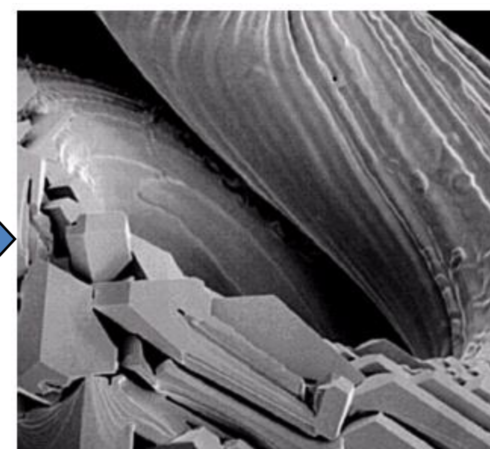
Simple Laplacian

1	1	1
1	-8	1
1	1	1

Variant of Laplacian



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



1^{st} Derivative Filtering

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

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

This vector has the important geometrical property that it points in the direction of the greatest rate of change of *f* at location (x,y)

1^{st} Derivative Filtering (cont…)

The magnitude of this vector is given by:

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

For practical reasons this can be simplified as:

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

1^{st} Derivative Filtering (cont…)

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

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

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

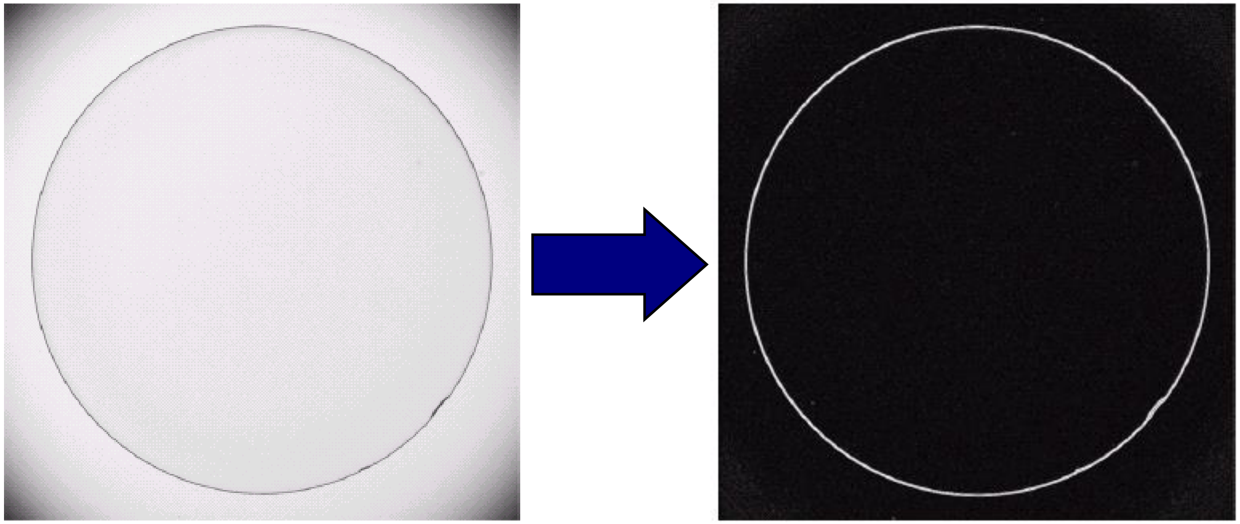
Based on the previous equations we can derive the *Sobel Operators*

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

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

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

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

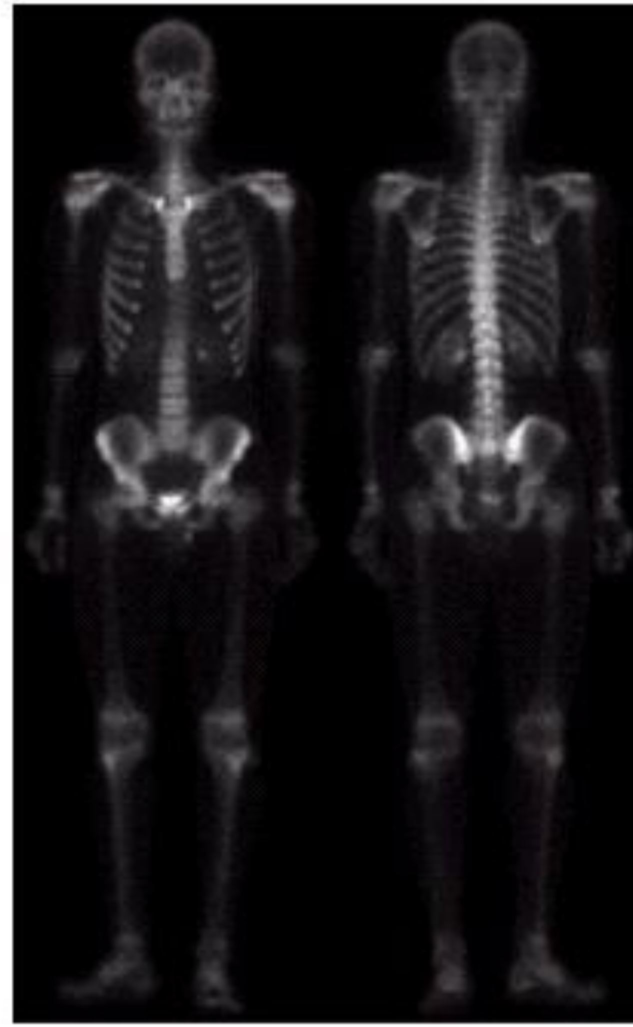
1^{st} & 2^{nd} Derivatives

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

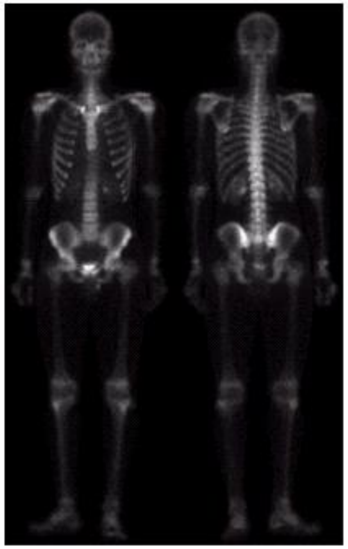
- 1^{st} order derivatives generally produce **thicker edges**
- 2^{nd} order derivatives have **a stronger response to fine detail e.g. thin lines**
- 1^{st} order derivatives have **a stronger response to grey level step**
- 2^{nd} order derivatives produce **a double response at step changes in grey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

Combining Spatial Enhancement Methods

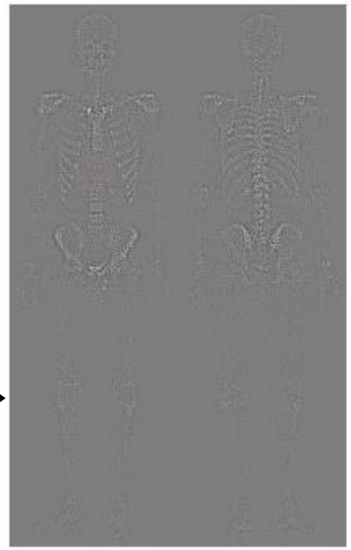
- Successful image enhancement is typically not achieved using a single operation
- Rather we combine a **range of techniques** in order to achieve a final result
- This example will focus on enhancing the bone scan to the right



Combining Spatial Enhancement Methods (cont…)

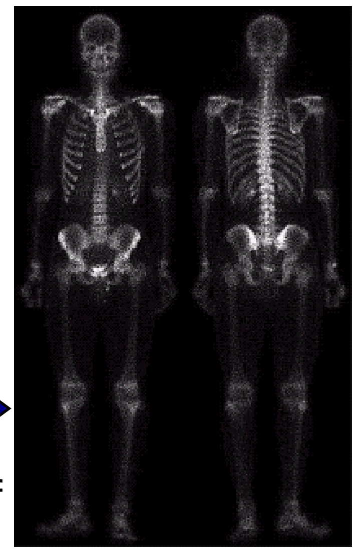


(a)



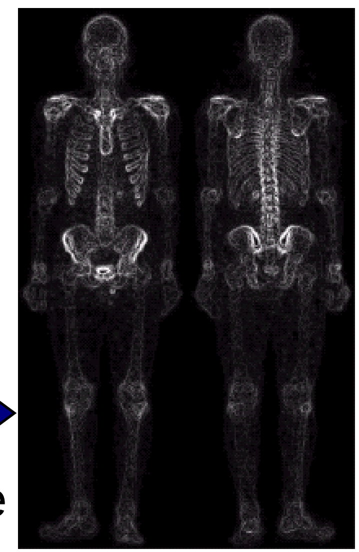
Laplacian filter of bone scan (a)

(b)



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

(c)



Sobel filter of bone scan (a)

(d)

Combining Spatial Enhancement Methods (cont…)

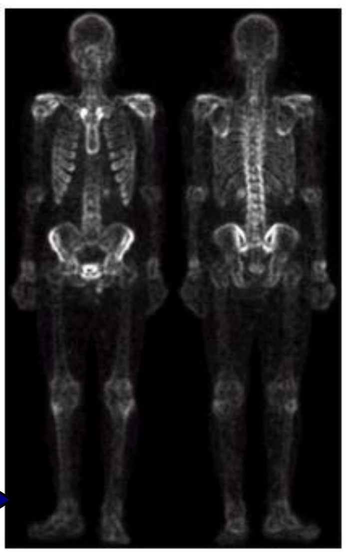
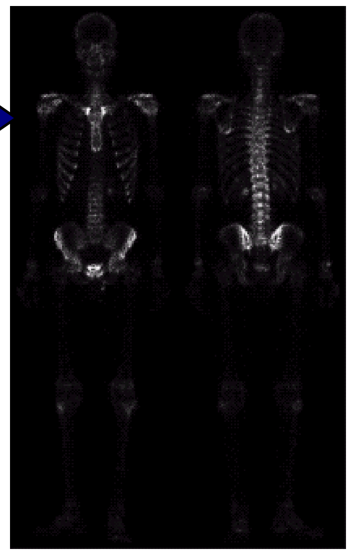


Image (d) smoothed with a 5*5 averaging filter

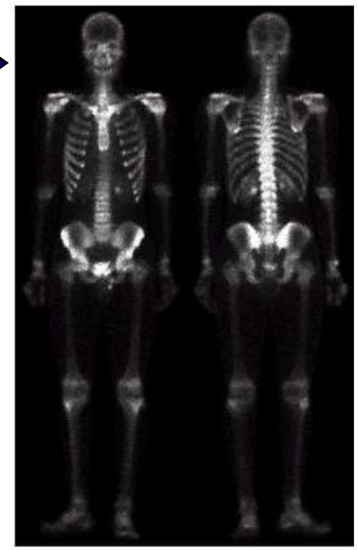
(e)

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



(f)

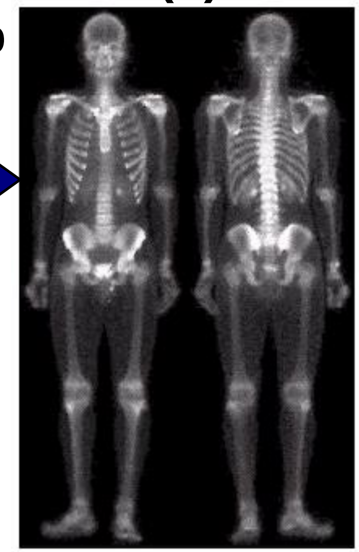
Sharpened image which is sum of (a) and (f)



(g)

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

(h)



Combining Spatial Enhancement Methods (cont…)

Compare the original and final images