DIGITAL IMAGE PROCESSING

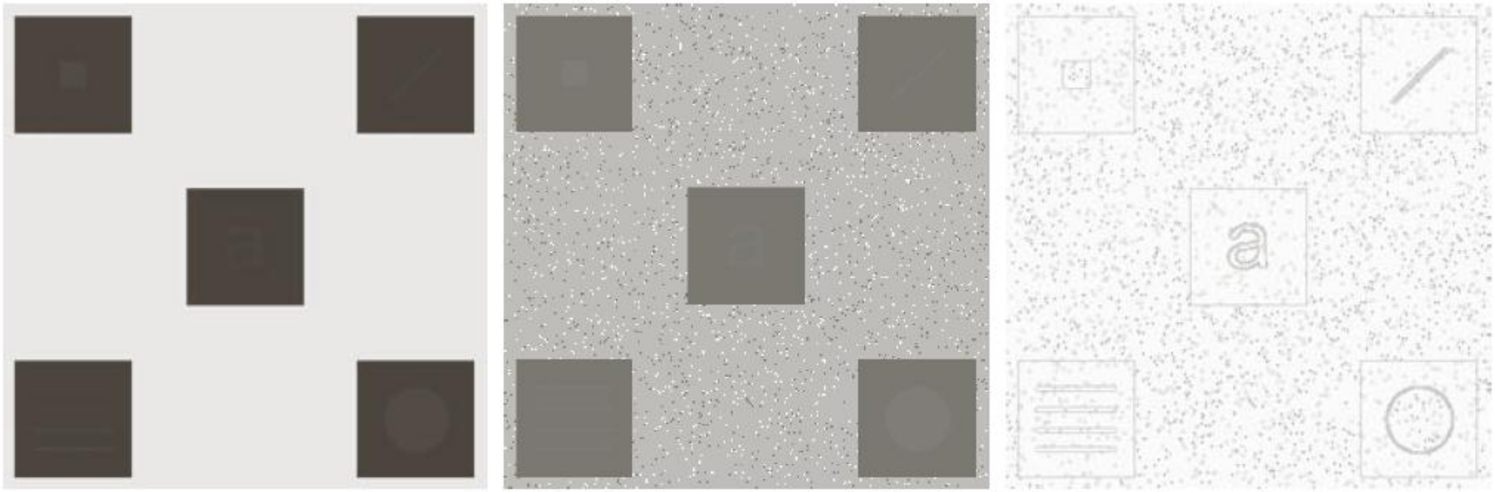
Image Enhancement
(Spatial Filtering 1)

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

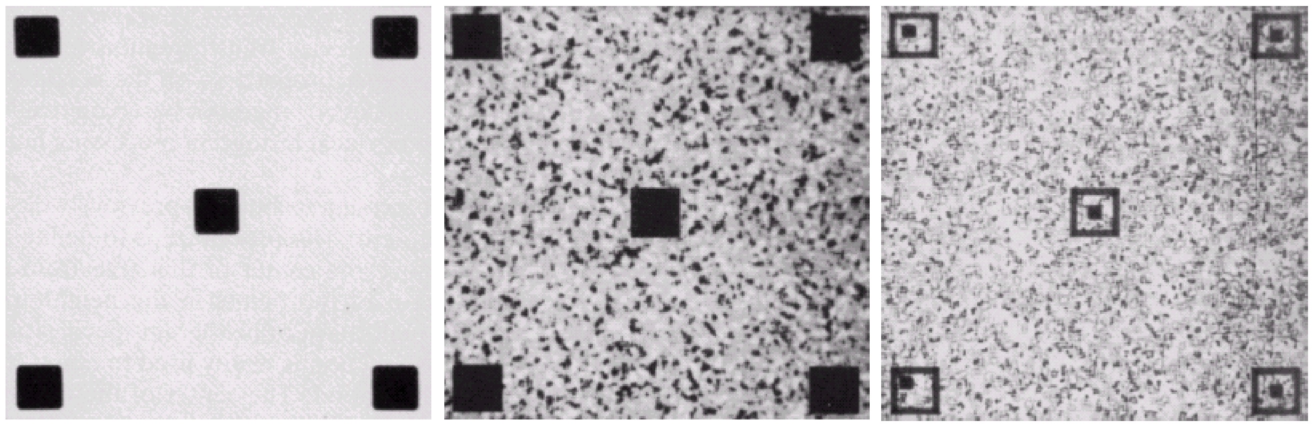
Local Enhancement

- Specify a neighbourhood of pixels
- Perform Histogram equalization on the neighbourhood (e.g. 3x3 or 7x7)
- Compute the gray level for the pixel (centered) in the output image
- Repeat the process for every pixel



a b c

FIGURE 3.26 (a) Original image. (b) Result of global histogram equalization. (c) Result of local histogram equalization applied to (a), using a neighborhood of size 3 × 3.



a b c

FIGURE 3.23 (a) Original image. (b) Result of global histogram equalization. (c) Result of local histogram equalization using a 7×7 neighborhood about each pixel.

Histogram Statistics for Image enhancement

- Histograms of images provide useful statistical parameters
- **Mean** value, $m = \sum_{i=0}^{L-1} r_i \, p(r_i)$ is the average value of gray levels in an image
- Mean is used as a measure of judging whether an area in an image is darker or lighter
- The gray level **variance** of pixels in an image is given by

$$\sigma^2 = \sum_{i=0}^{L-1} [r_i - m]^2 \, p(r_i)$$

- The variance provides a measure of the **contrast** in the image

$$m = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \qquad \sigma^2 = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [f(x, y) - m]^2$$

Histogram Statistics for Image enhancement

- Consider the following 2-bit image of size 5x5, L=4

0	0	1	1	2
1	2	3	0	1
3	3	2	2	0
2	3	1	0	0
1	1	3	2	2

MN=25

$$p(r_0) = \frac{6}{25} = 0.24; \; p(r_1) = \frac{7}{25} = 0.28$$

$$p(r_2) = \frac{7}{25} = 0.28; \; p(r_3) = \frac{5}{25} = 0.20$$

$$m = \sum_{i=0}^{3} r_i \, p(r_i)$$

$$m = (0)(0.24) + (0)(0.28) + (2)(0.28) + (3)(0.20)$$

$$m = 1.44$$

Similarly

$$m = \frac{1}{25} \sum_{x=0}^{4} \sum_{y=0}^{4} f(x, y) = 1.44$$

Histogram Statistics for Image enhancement… Example

- To judge whether a particular pixel (x,y) is *relatively* dark/light at that point is to compare the local mean with global mean

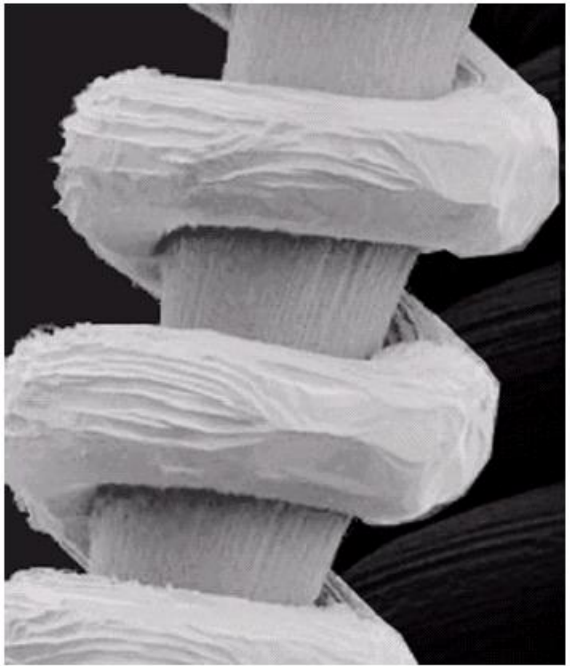
$$m_{S_{x,y}} \leq k_0 m_G$$

- $$g(x,y) = \begin{cases} E.f(x,y) & \text{if } \; m_{S_{x,y}} \leq k_0 m_G \; \textbf{AND} \; k_1\sigma_G \leq \sigma_{S_{x,y}} \leq k_2\sigma_G \\ f(x,y) & \text{otherwise} \end{cases}$$

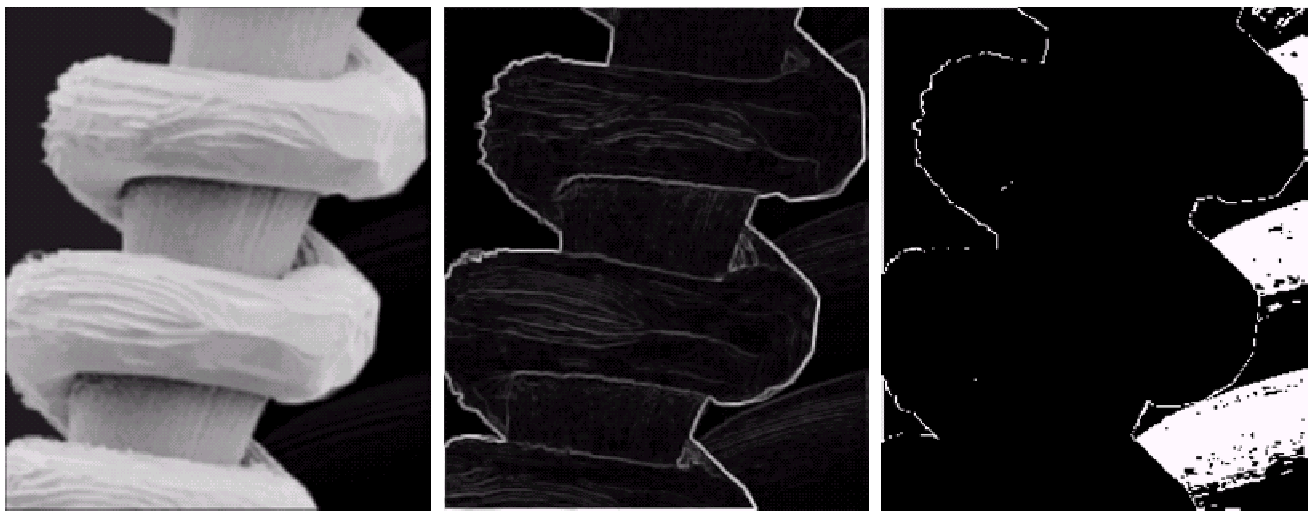
The values of the constants used in this example are image specific and can be tuned after a bit of experimentation

E=4, **k0=0.4** (areas where we need enhancement are darker than the Global average, therefore less than half), **k1=0.02** and **k2=0.4**

The values of k1 and k2 are < 1, if we need enhancements for dark areas
And >1, if we need enhancements for lighter areas



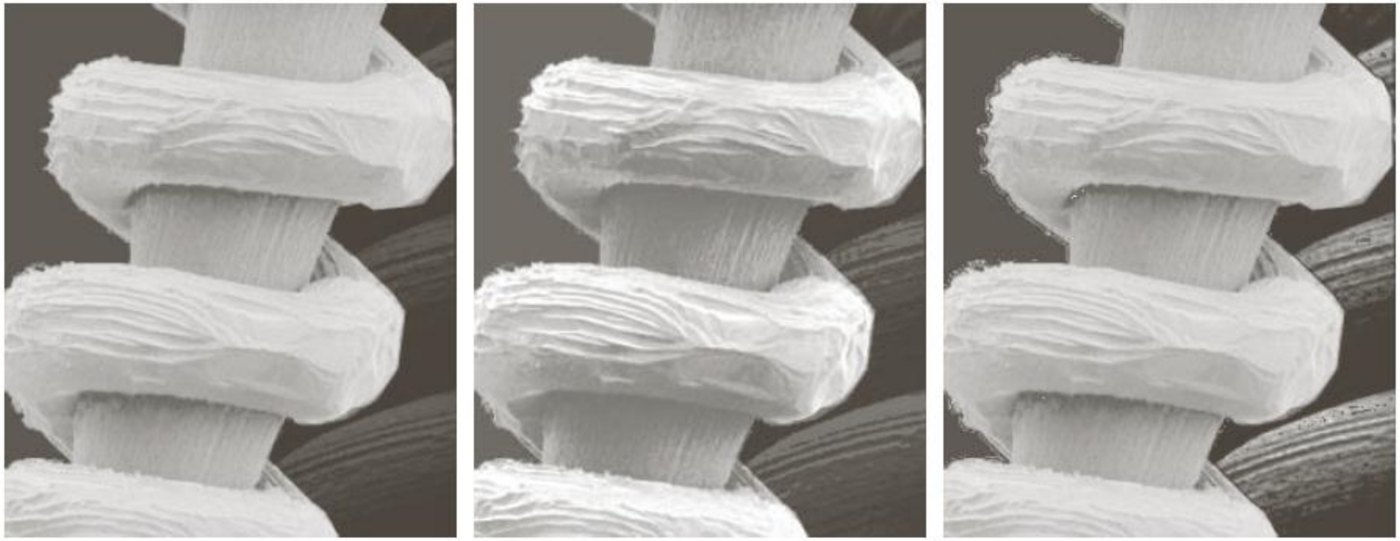
Chapter 3
Image Enhancement in the Spatial Domain



a b c

FIGURE 3.25 (a) Image formed from all local means obtained from Fig. 3.24 using Eq. (3.3-21). (b) Image formed from all local standard deviations obtained from Fig. 3.24 using Eq. (3.3-22). (c) Image formed from all multiplication constants used to produce the enhanced image shown in Fig. 3.26.

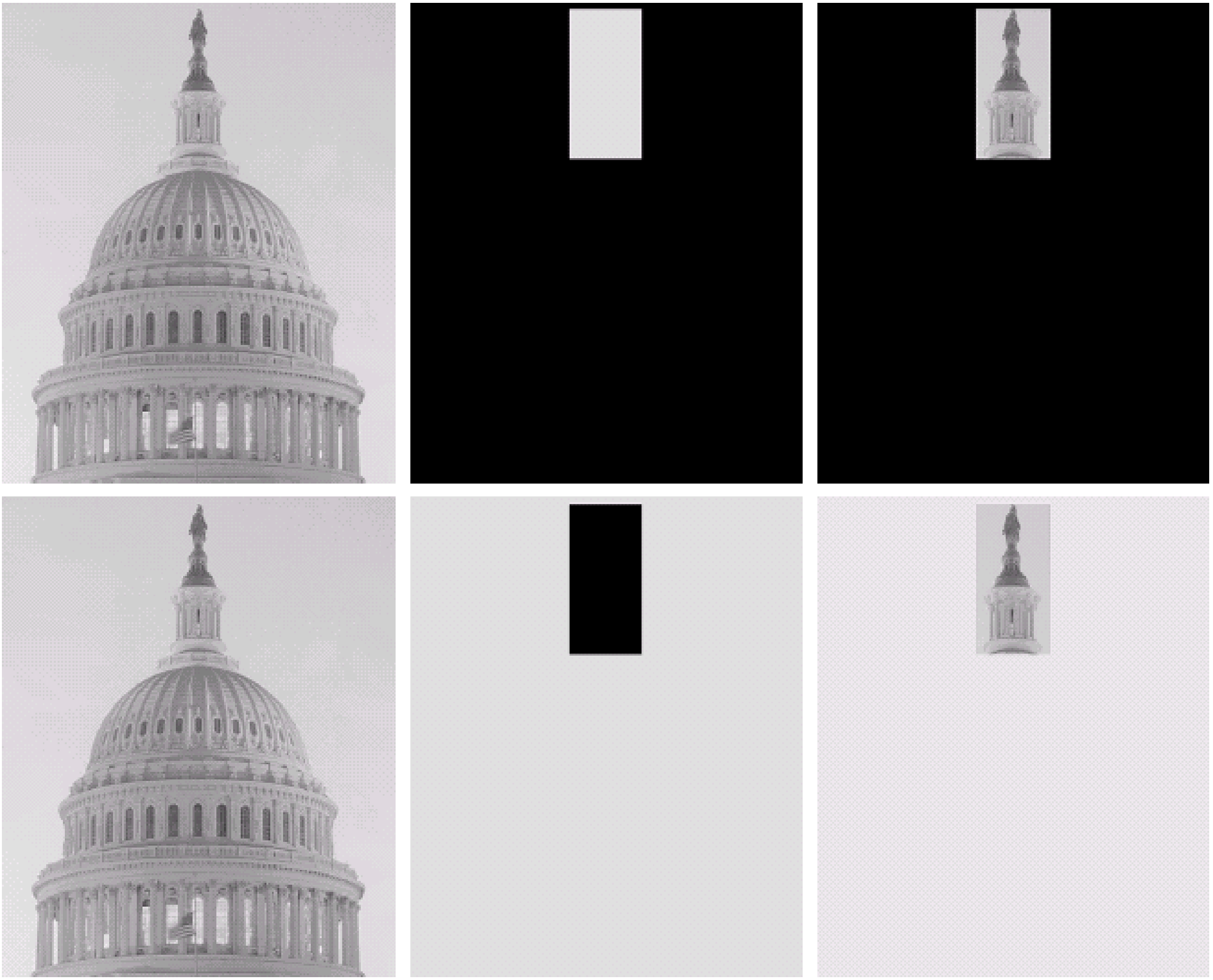
Histogram Statistics for Image enhancement… Example



a b c

FIGURE 3.27 (a) SEM image of a tungsten filament magnified approximately 130×. (b) Result of global histogram equalization. (c) Image enhanced using local histogram statistics. (Original image courtesy of Mr. Michael Shaffer, Department of Geological Sciences, University of Oregon, Eugene.)

DIP using Arithmetic/Logic Operations



a b c
d e f

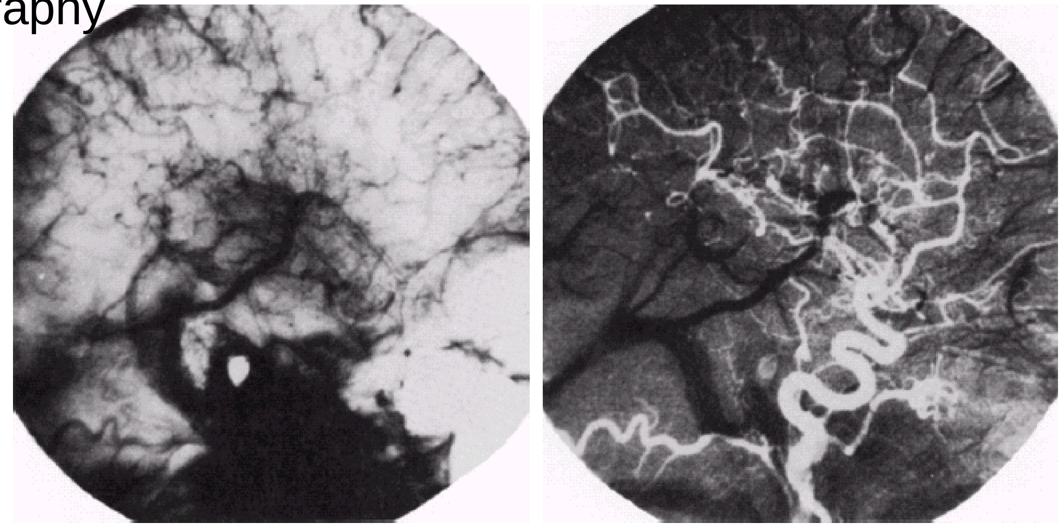
FIGURE 3.27 (a) Original image. (b) AND image mask. (c) Result of the AND operation on images (a) and (b). (d) Original image. (e) OR image mask. (f) Result of operation OR on images (d) and (e).

Image subtraction

$$g(x, y) = f(x, y) - h(x, y)$$

Detecting a change in series of images (video)

Mask mode radiography

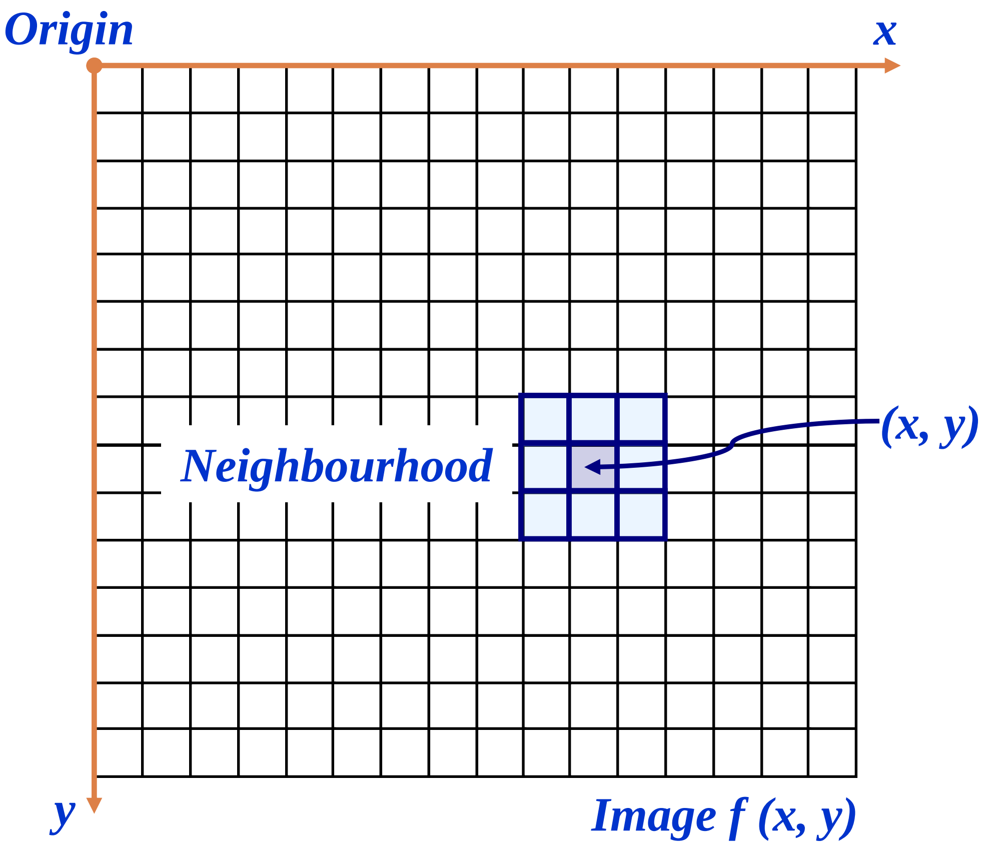


a b

FIGURE 3.29 Enhancement by image subtraction. (a) Mask image. (b) An image (taken after injection of a contrast medium into the bloodstream) with mask subtracted out.

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

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

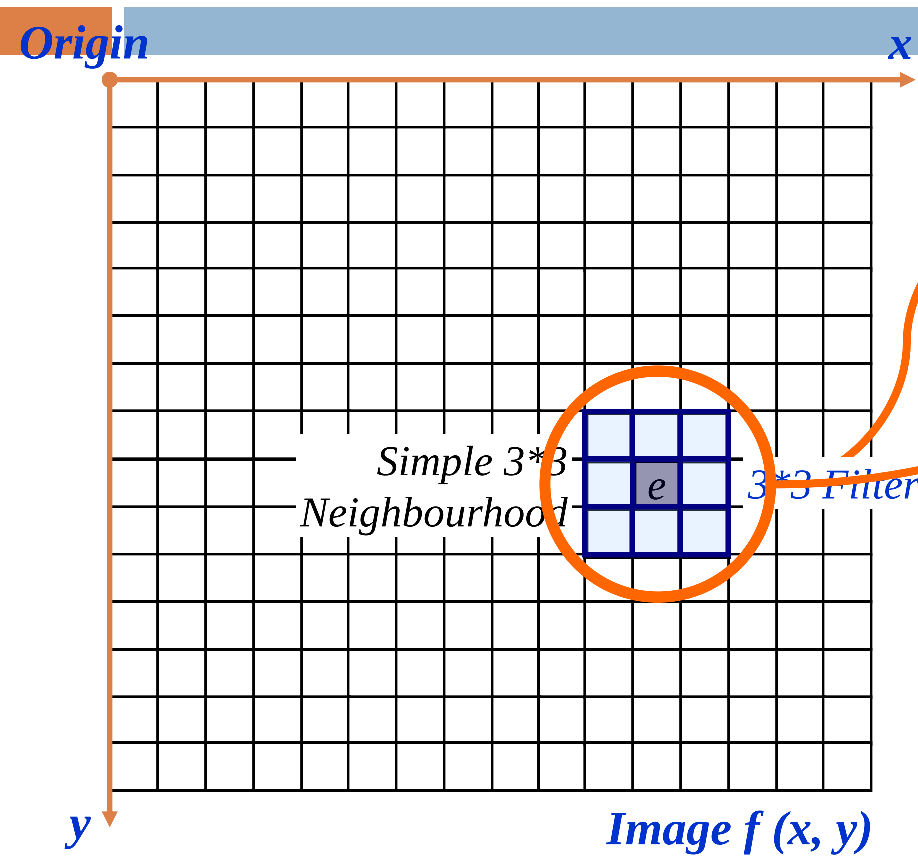
y

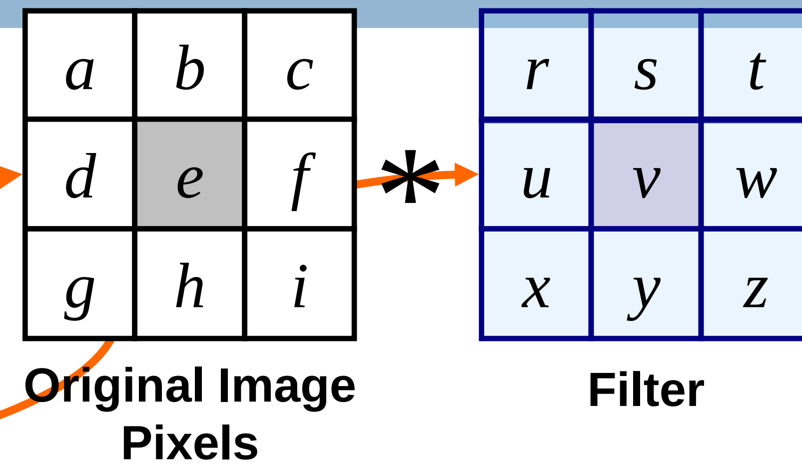
Enhanced Image

x

y

The Spatial Filtering Process

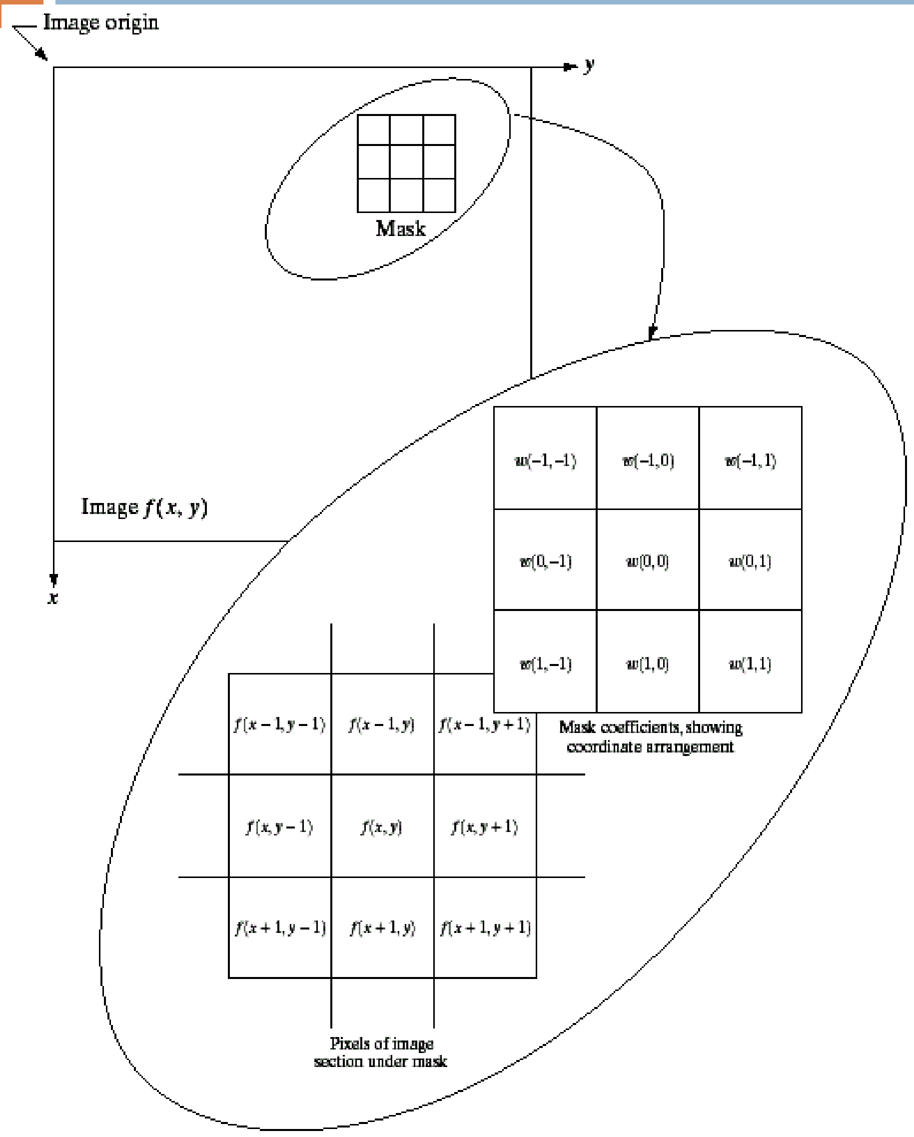




$$e_{processed} = v*e + r*a + s*b + t*c + u*d + w*f + x*g + y*h + z*i$$

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

Spatial Filtering: Equation Form



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

Filtering can be given in equation form as shown above

Notations are based on the image shown to the left

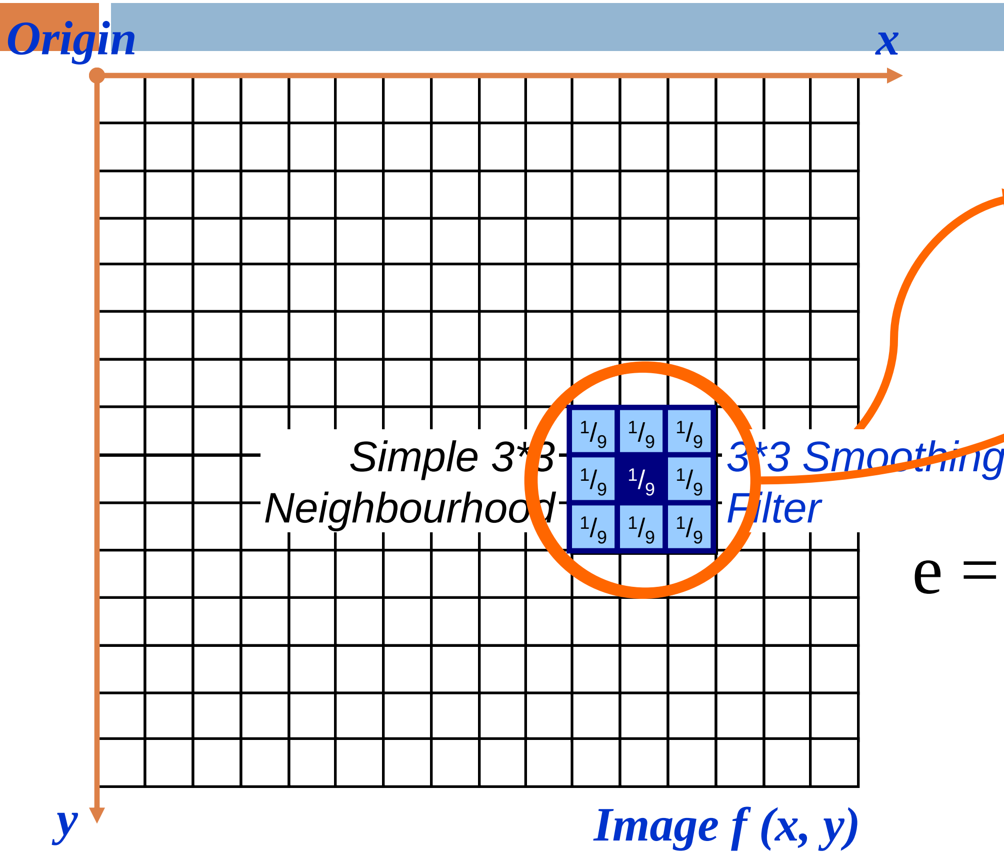
Smoothing Spatial Filters

- 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

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

Simple averaging filter

Smoothing Spatial Filtering



104	100	108
99	106	98
95	90	85

Original Image Pixels

*

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

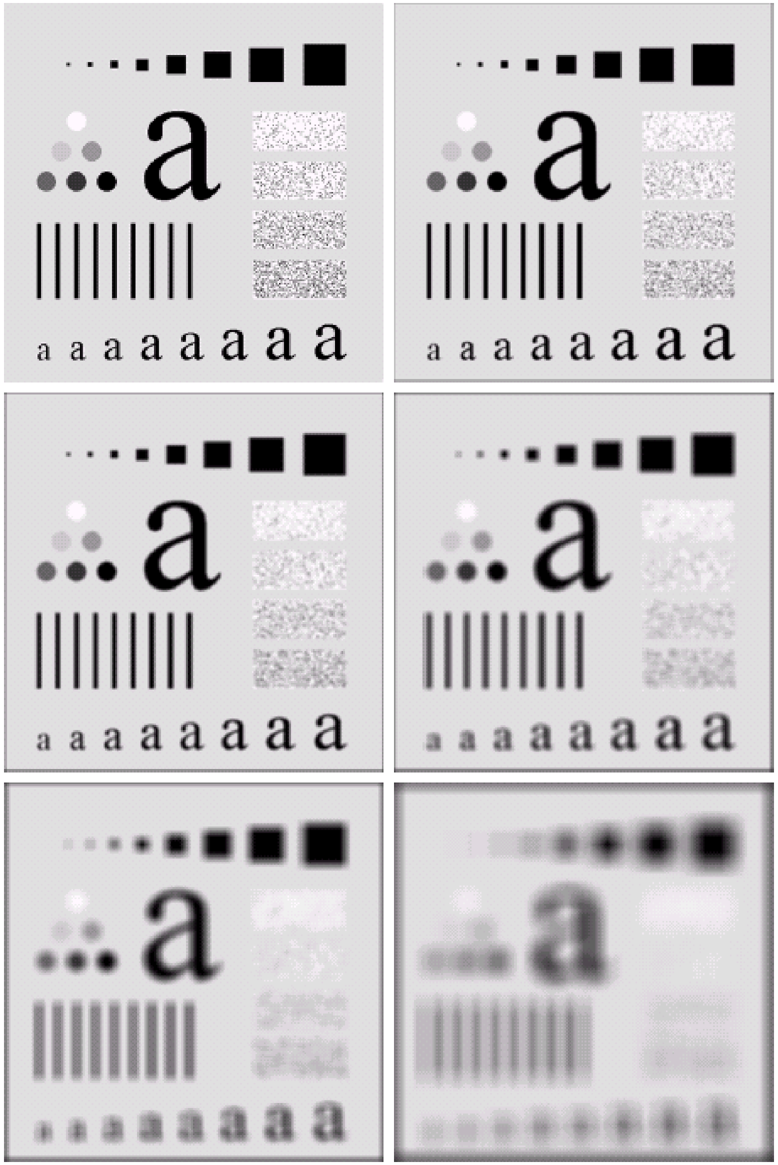
Filter

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

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

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



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

Image Smoothing Example

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

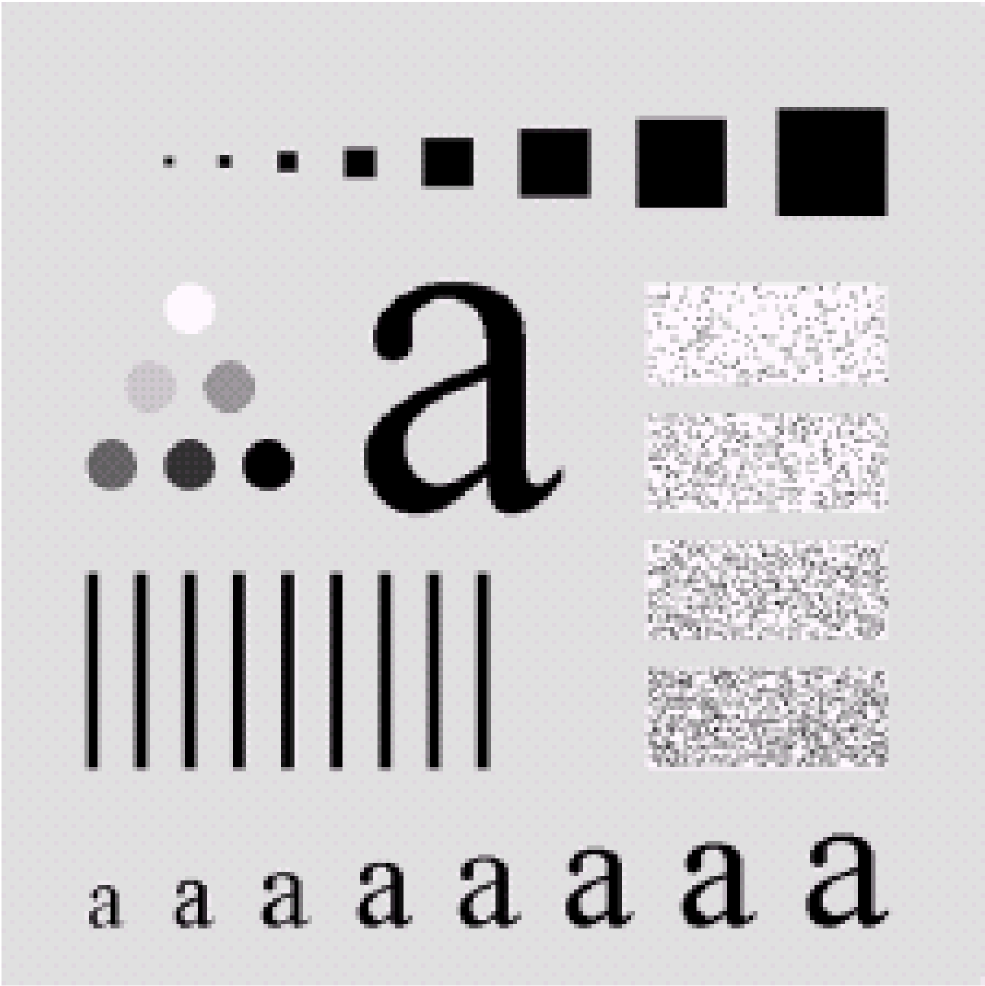
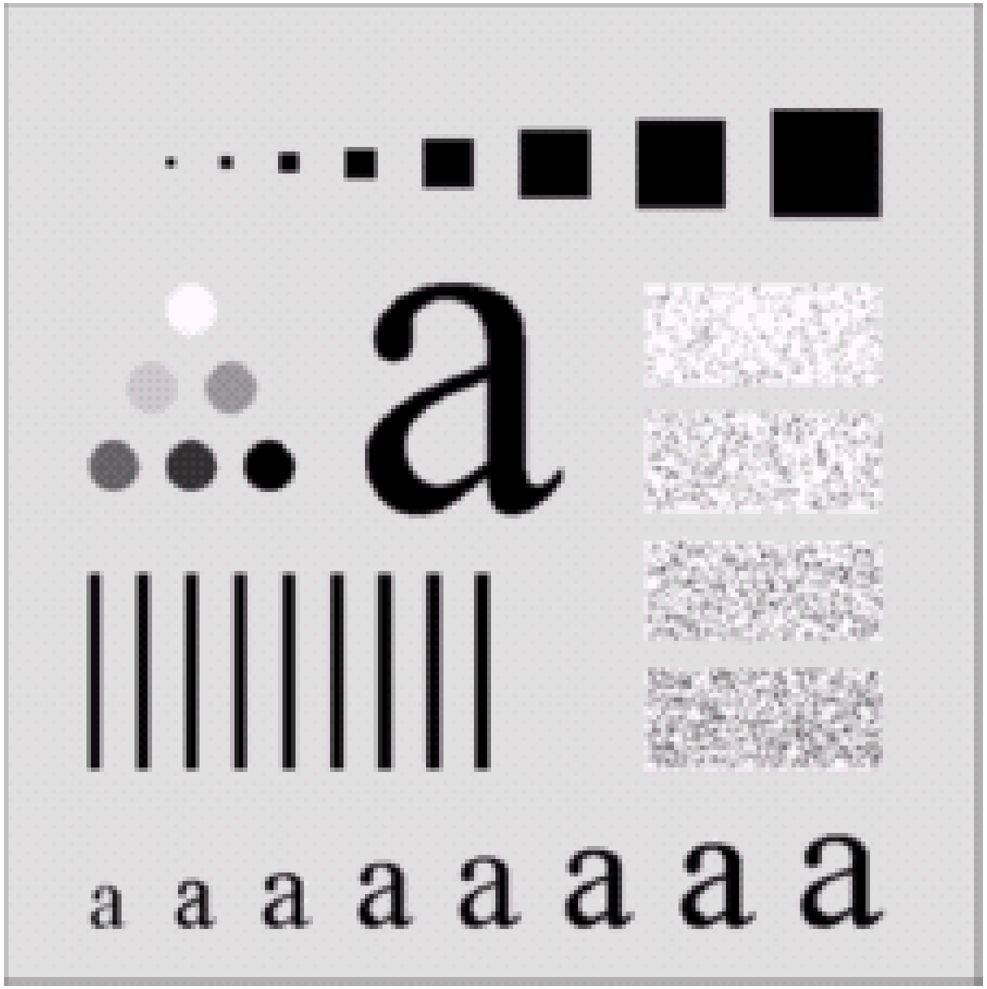
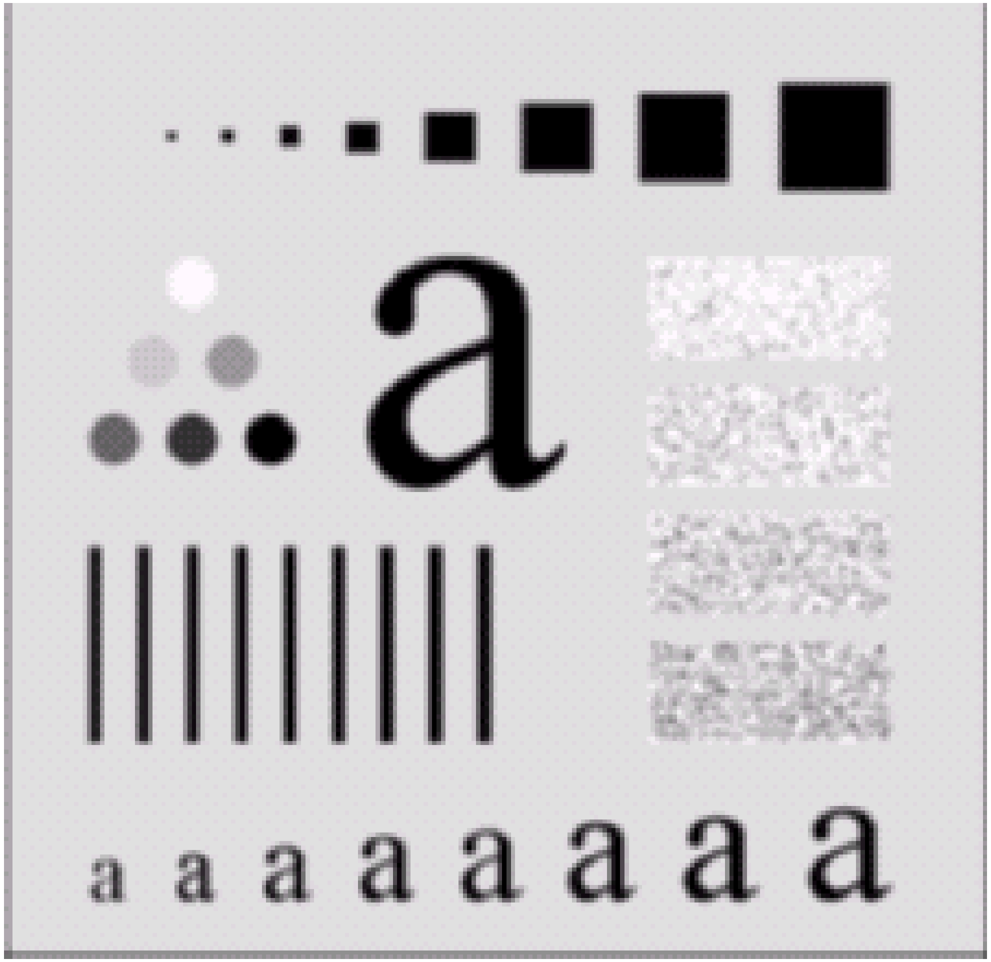


Image Smoothing Example



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

Image Smoothing Example



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

Image Smoothing Example

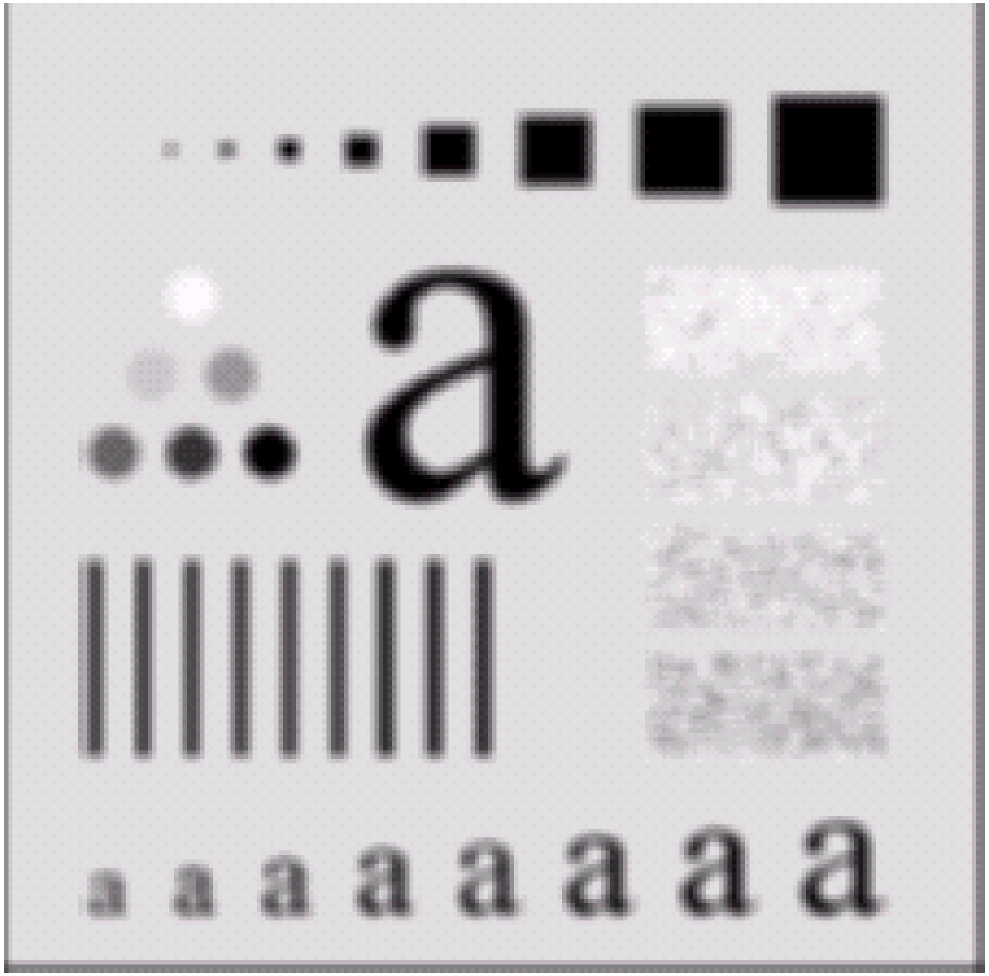
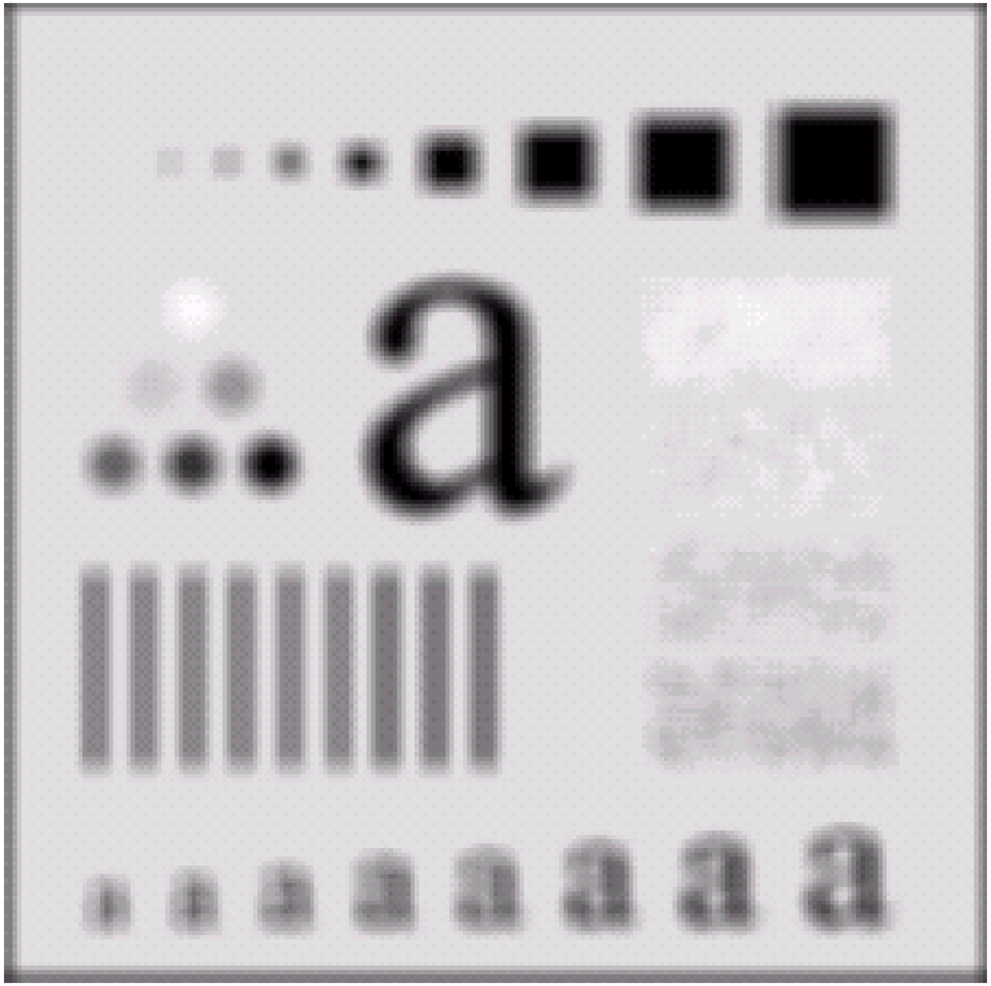
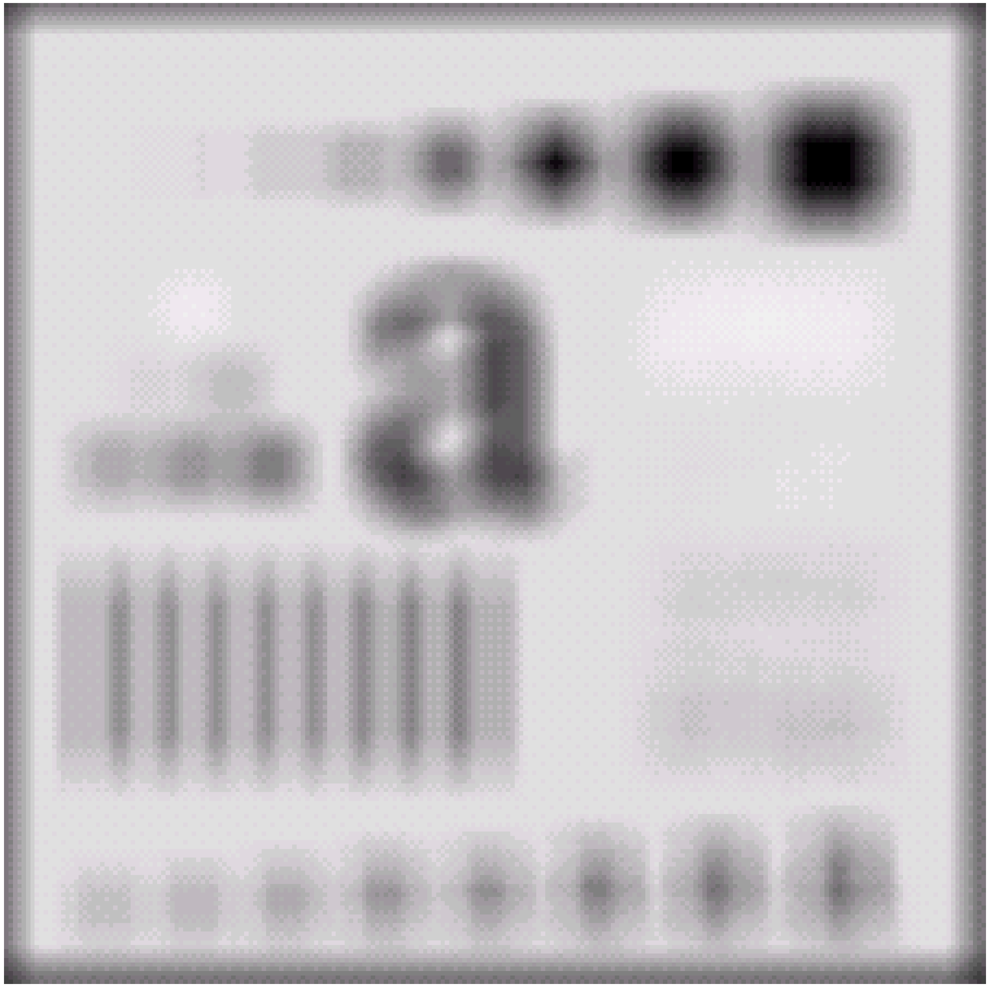


Image Smoothing Example



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

Image Smoothing Example



Weighted Smoothing Filters

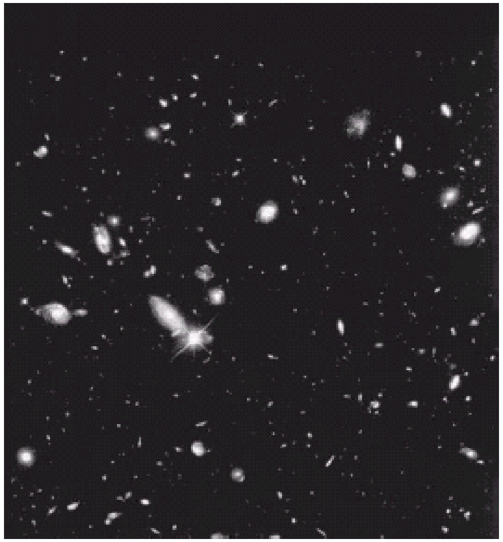
- More effective smoothing filters can be generated by allowing different pixels in the neighbourhood different weights in the averaging function
  - Pixels closer to the central pixel are more important
  - Often referred to as a *weighted averaging*

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

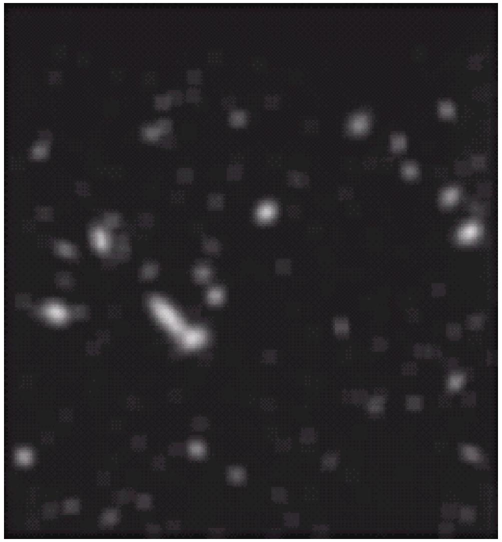
Weighted averaging filter

Another Smoothing Example

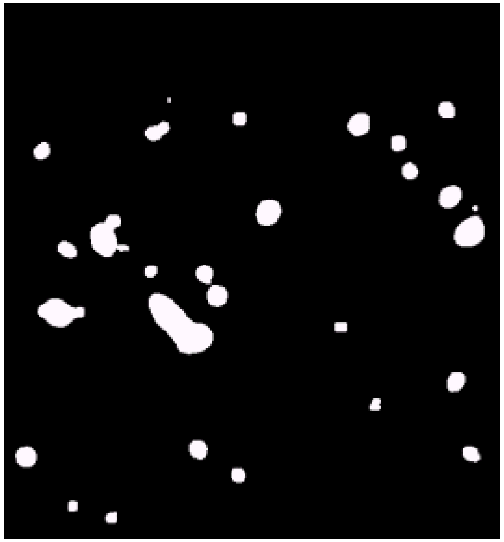
- By smoothing the original image we get rid of lots of the finer detail which leaves only the gross features for thresholding



Original Image



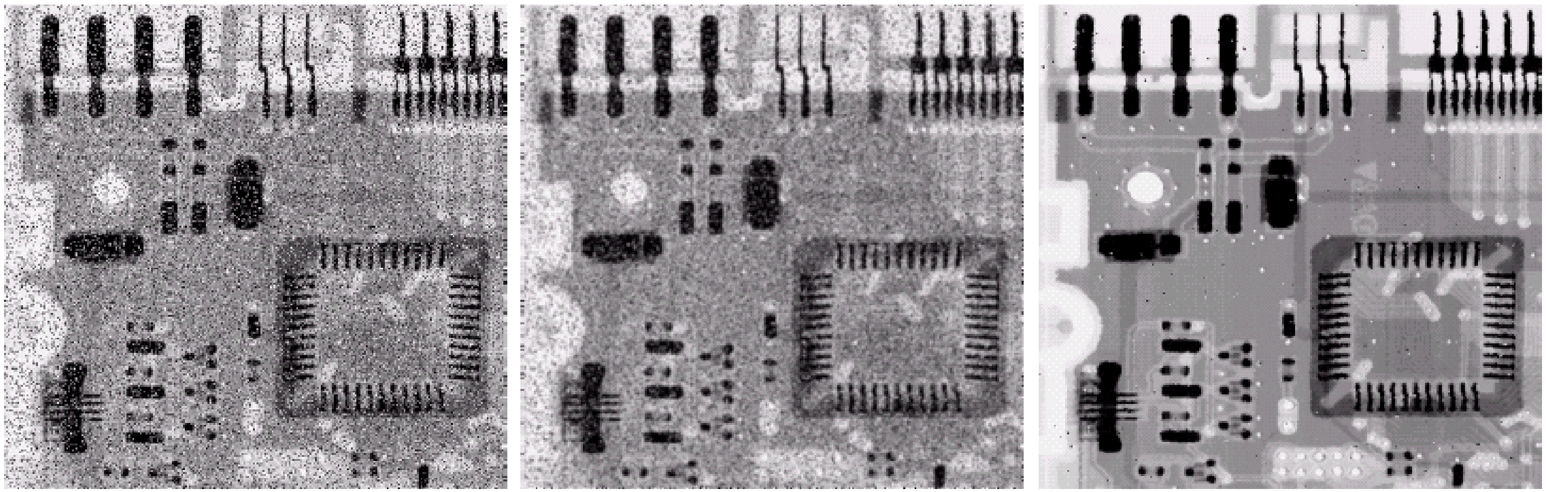
Smoothed Image



Thresholded Image

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

Averaging Filter Vs. Median Filter Example

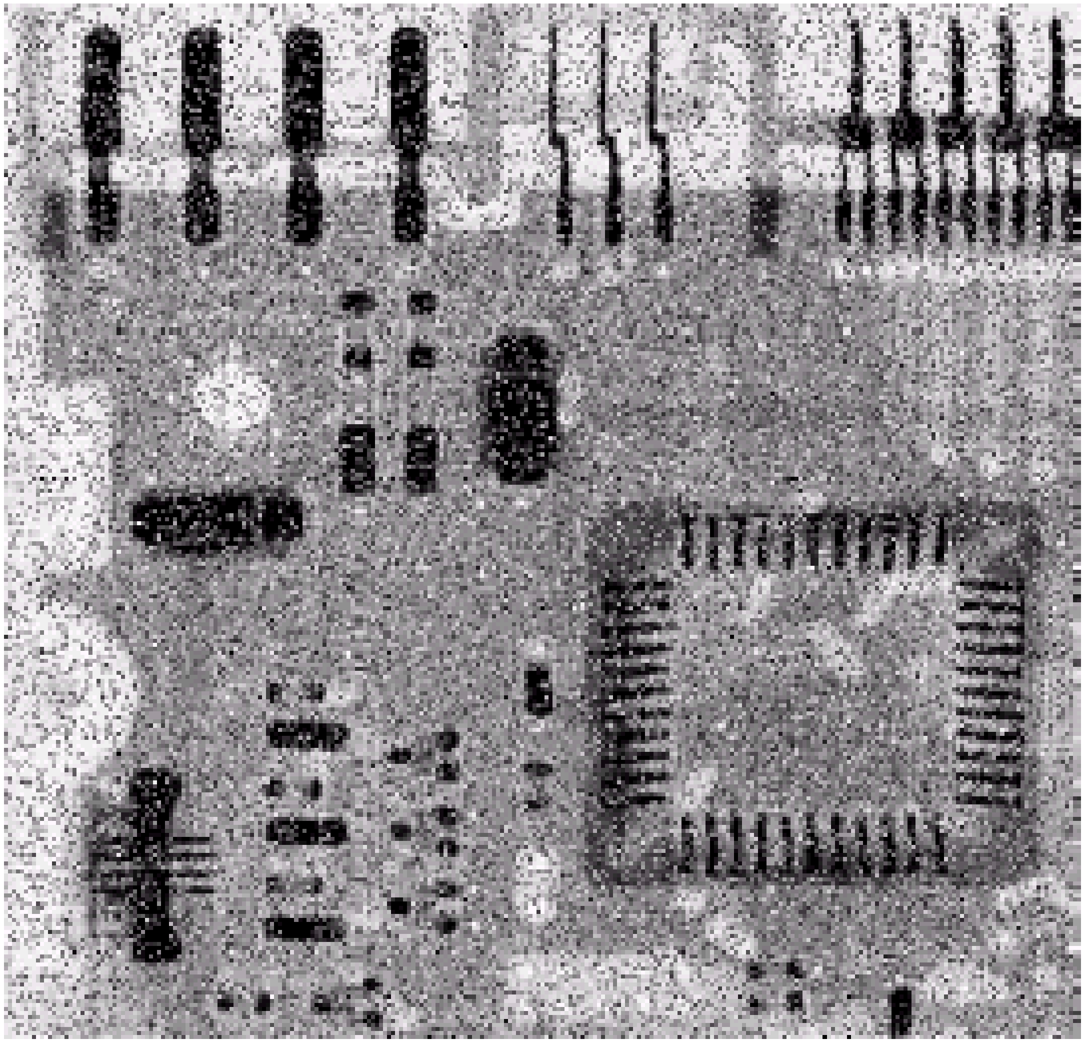


Original Image With Noise | **Image After Averaging Filter** | **Image After Median Filter**

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

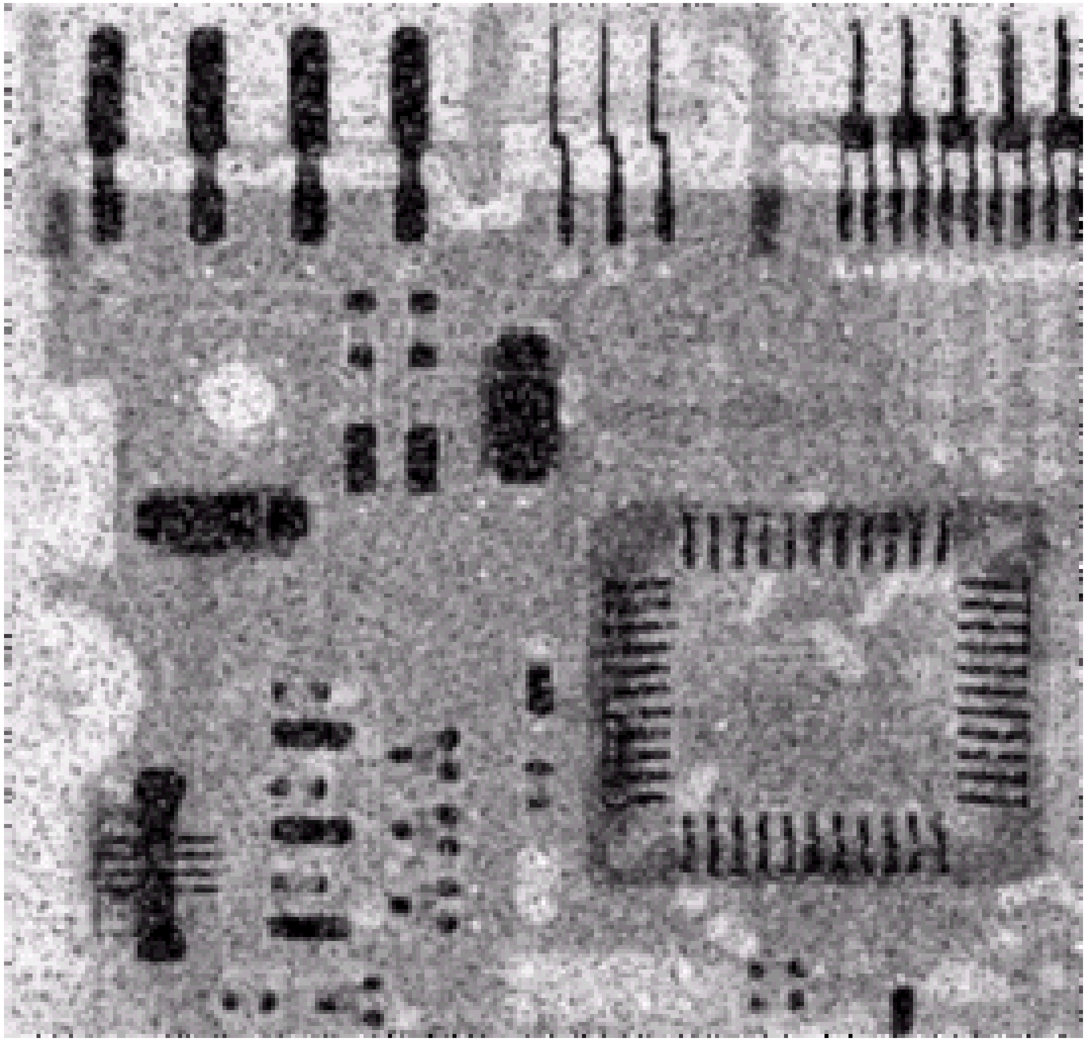
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Averaging Filter Vs. Median Filter Example



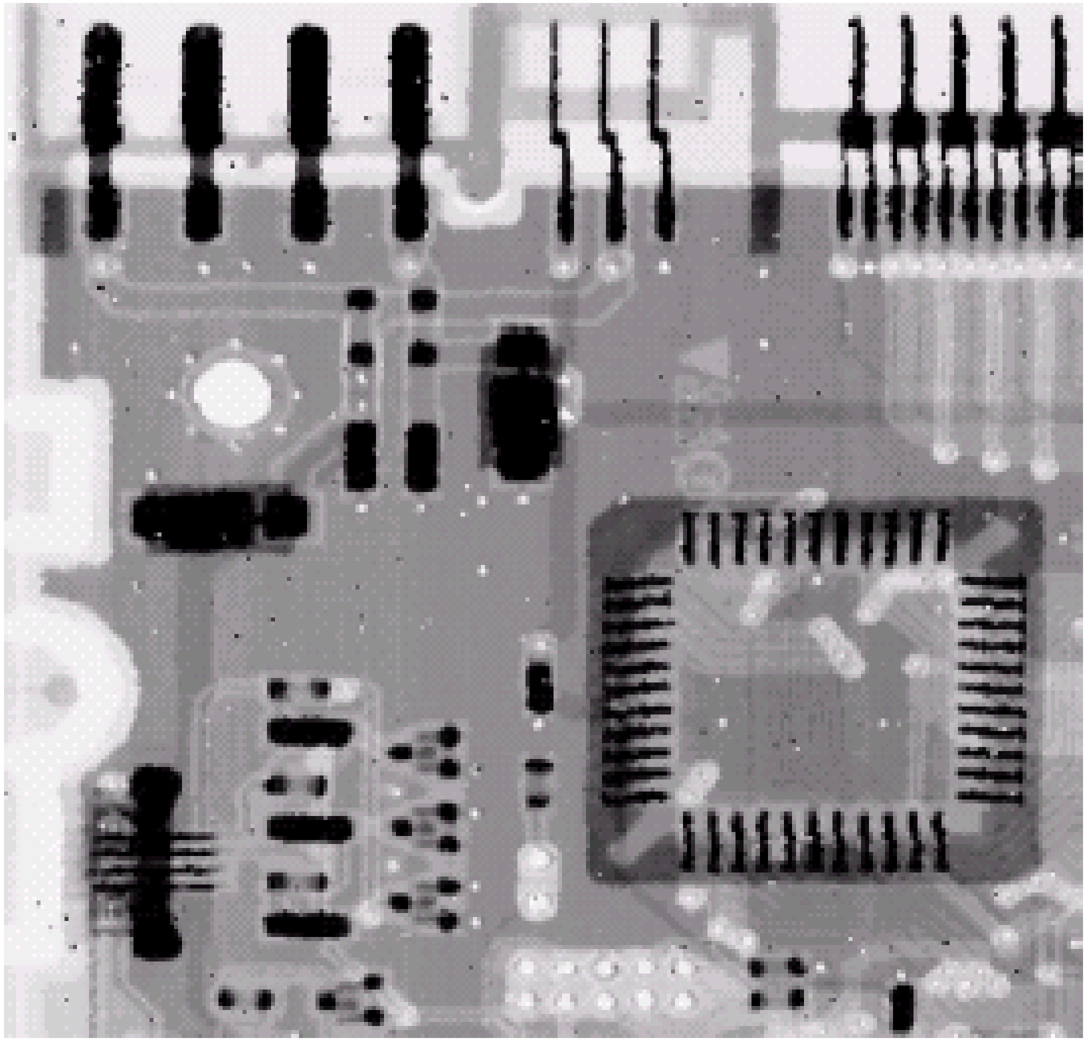
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Averaging Filter Vs. Median Filter Example



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

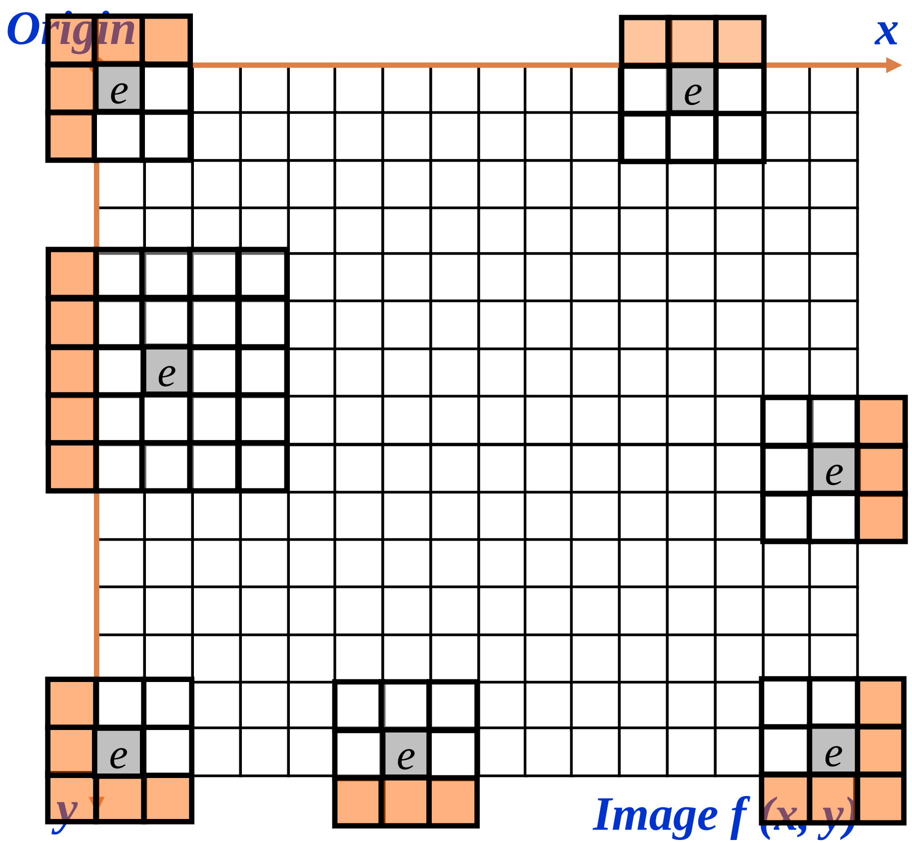
Averaging Filter Vs. Median Filter Example



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

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…)

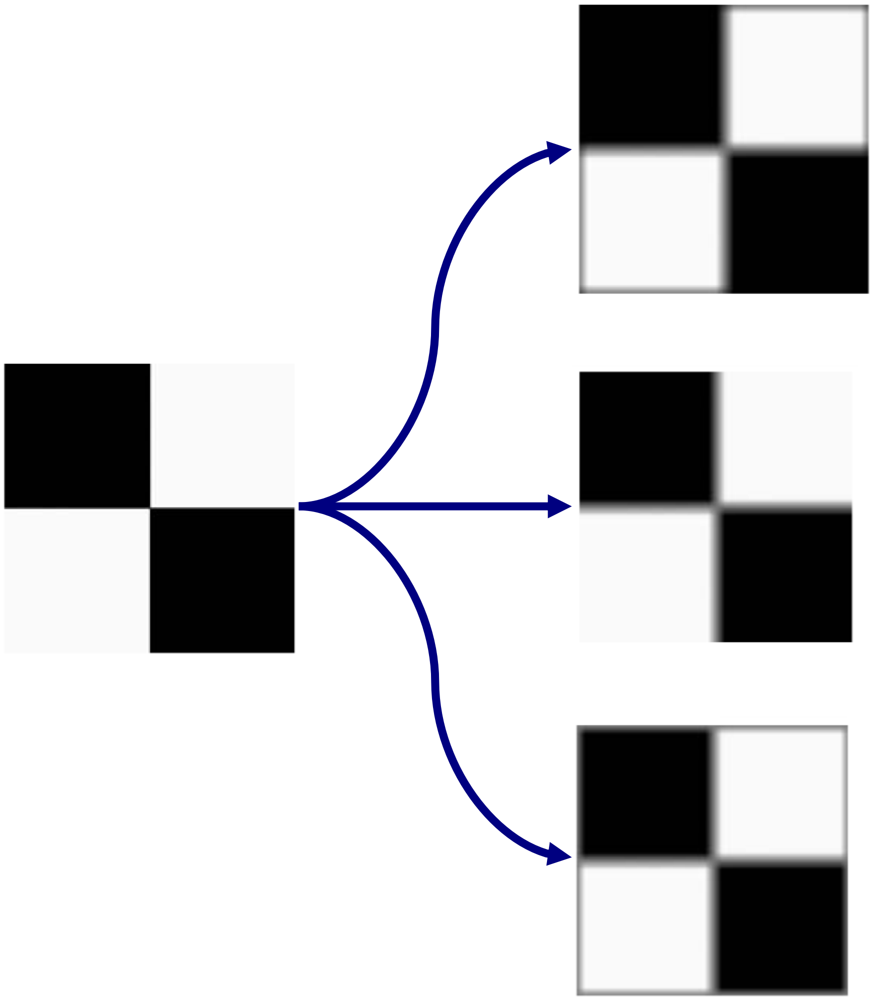
- There are a few approaches to dealing with missing edge pixels:
  - Omit missing pixels
    - Only works with some filters
    - Can add extra code and slow down processing
  - Pad the image
    - Typically with either all white or all black pixels
  - Replicate border pixels
  - Allow pixels *wrap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

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! (cont…)

Original Image



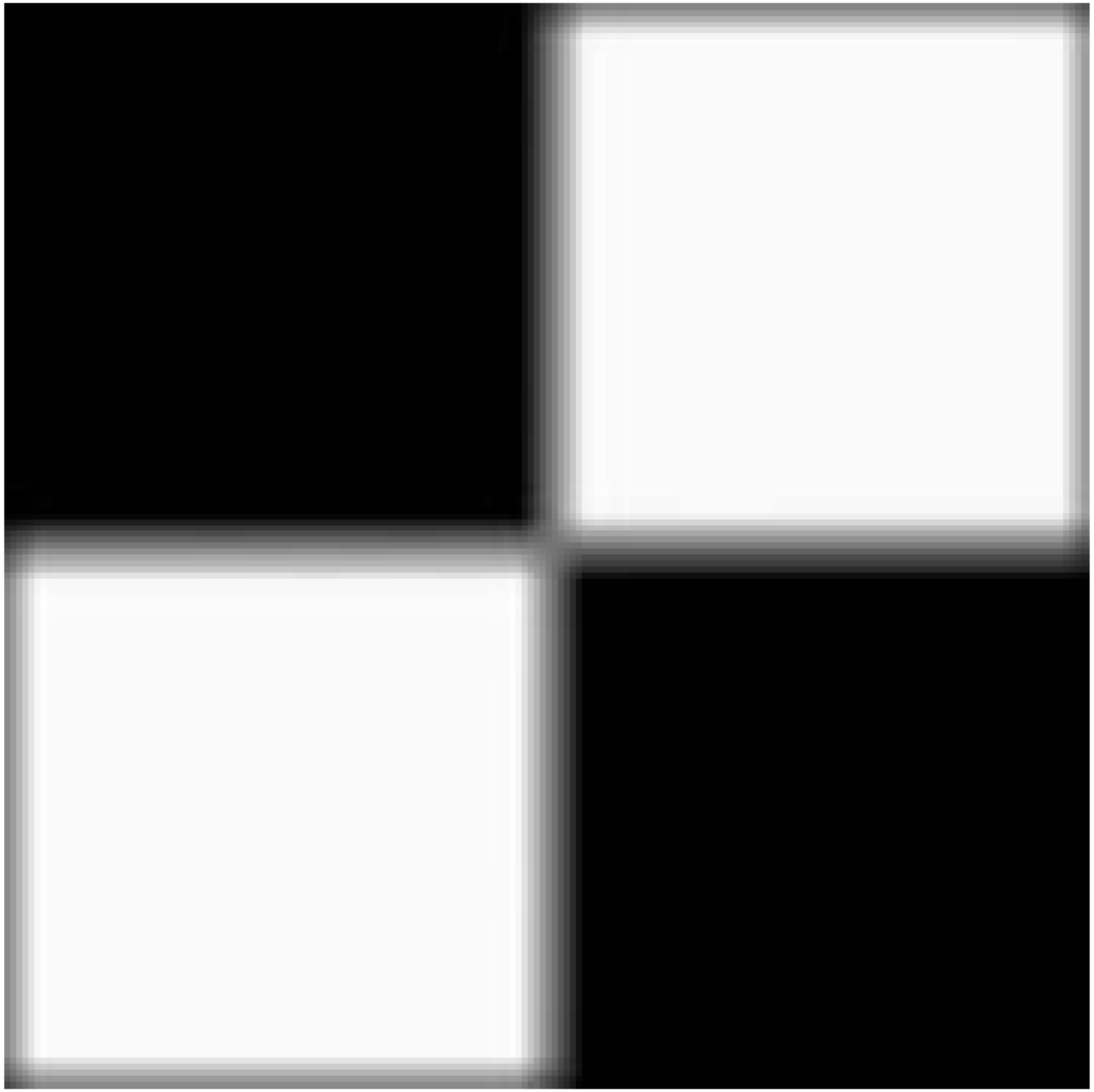
Filtered Image:
Zero Padding

Filtered Image:
Replicate Edge Pixels

Filtered Image:
Wrap Around Edge Pixels

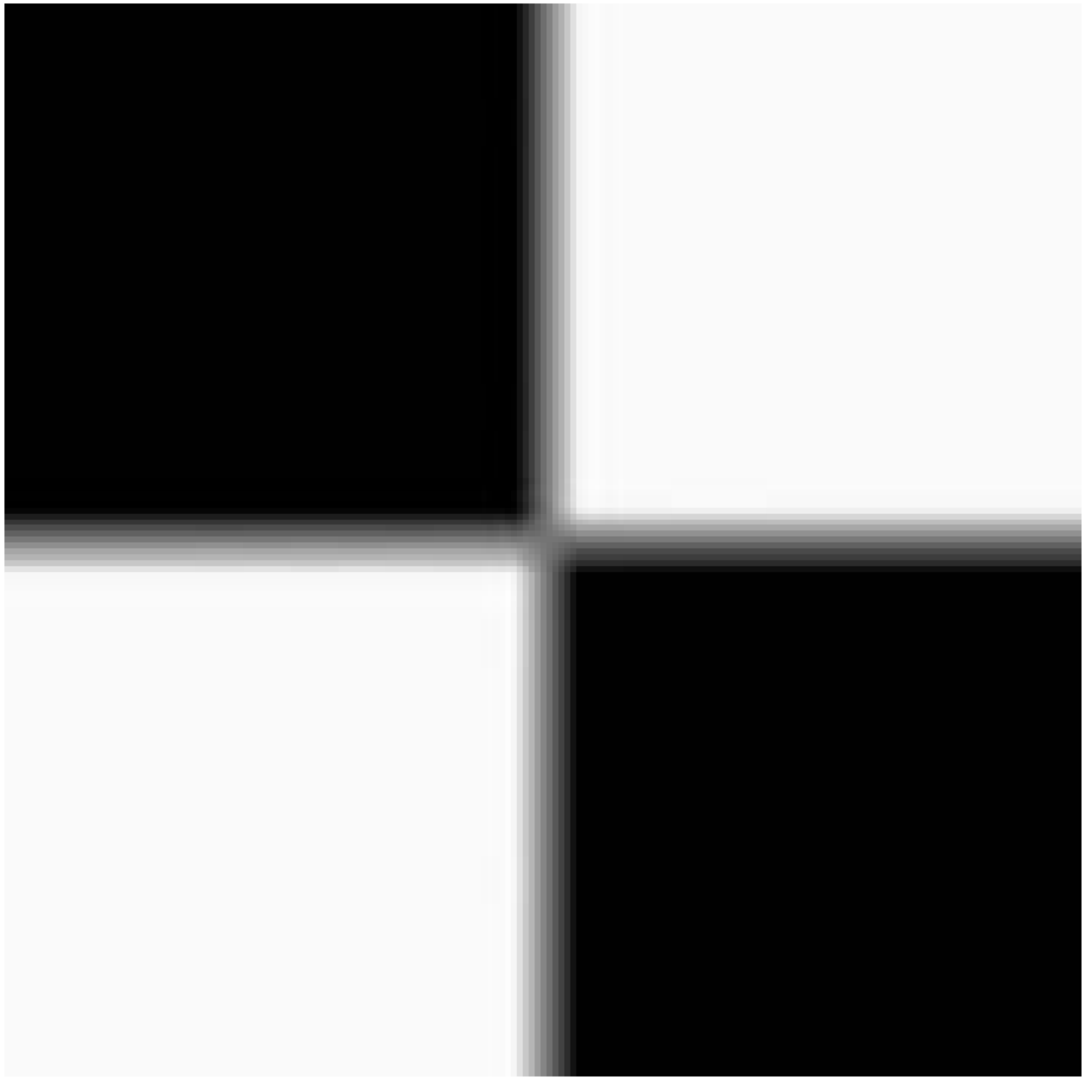
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

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…)

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

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

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

- For symmetric filters it makes no difference

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
- Next time we will looking at sharpening filters and more on filtering and image enhancement