Computer Graphics and Image Processing
Lecture 3

Muhammad Usman Ghani Khan

Department of Computer Science & Engineering
UET, Lahore

Autumn Semester

Outline

Agenda

Today Discussion

- What is image enhancement?
- Different kinds of image enhancement
- Histogram processing
- Point processing
- Neighbourhood operations

What is image enhancement?

- Image enhancement is the process of making images more useful. The reasons for doing this include:
  - Highlighting interesting detail in images
  - Removing noise from images
  - Making images more visually appealing

Image Enhancement: Example 1

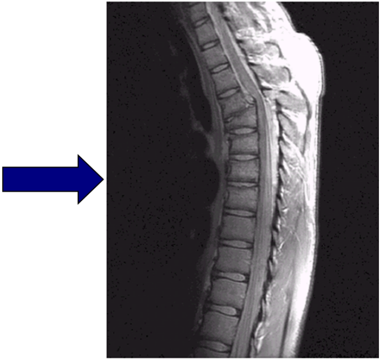


Image Enhancement: Example 2

Image Enhancement: Example 3

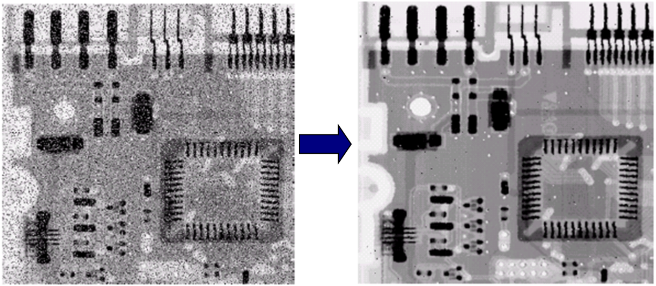


Image Enhancement: Example 4

Spatial & Frequency Domains

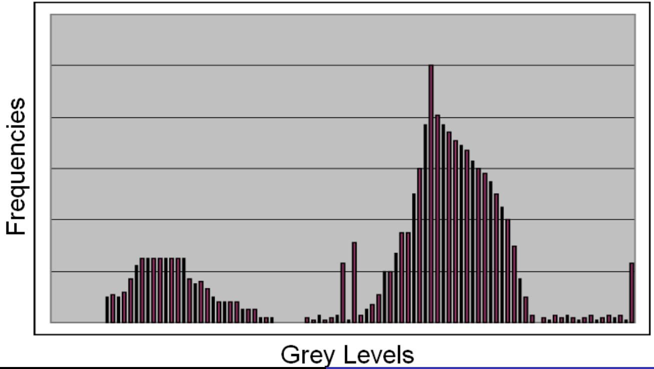
There are two broad categories of image enhancement techniques

1. Spatial domain techniques
   - Direct manipulation of image pixels
2. Frequency domain techniques
   - Manipulation of Fourier transform or wavelet transform of an image

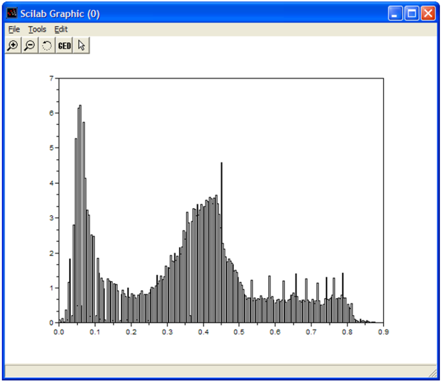
For the moment we will concentrate on techniques that operate in the spatial domain

Image Histograms

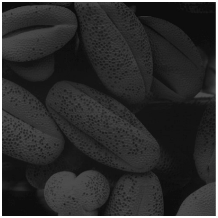
- The histogram of an image shows us the distribution of grey levels in the image
- Massively useful in image processing, especially in segmentation

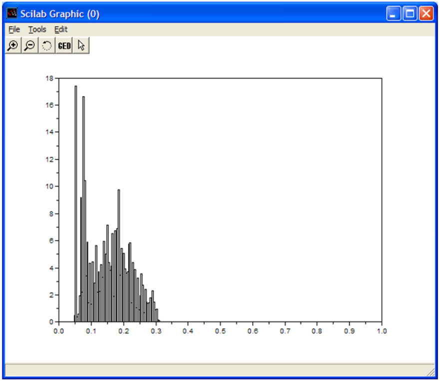


Histogram: Example 1

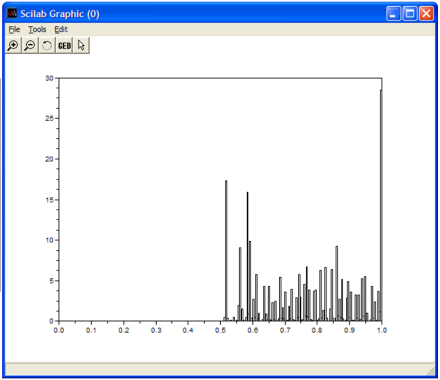


Histogram: Example 2

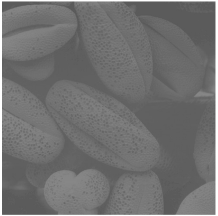


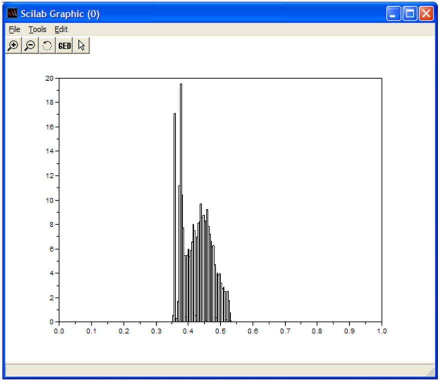


Histogram: Example 3

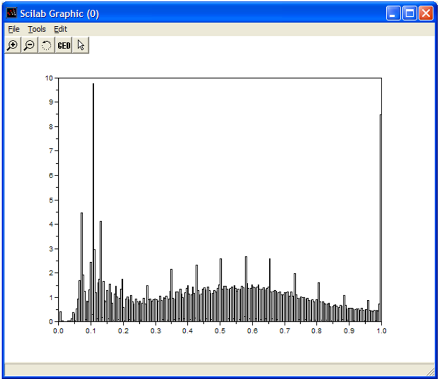


Histogram: Example 4

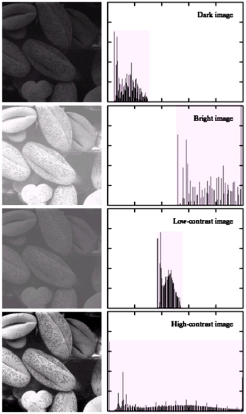




Histogram: Example 5



Histograms Examples: Final Note



- A selection of images and their histograms
- Notice the relationships between the images and their histograms
- Note that the high contrast image has the most evenly spaced histogram

Outline

1. Image Enhancement
2. Image Histograms
   - Histogram Processing
3. Image Enhancement (Spatial Filtering 1)
   - Spatial filer examples
   - Filtering at Edges

Contrast Stretching

- We can fix images that have poor contrast by applying a pretty simple contrast specification
- The interesting part is how do we decide on this transformation function?

Histogram Equalisation

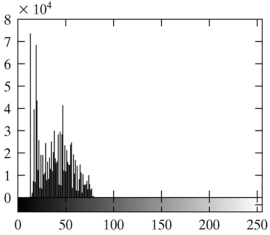
- Spreading out the frequencies in an image (or equalising the image) is a simple way to improve dark or washed out images

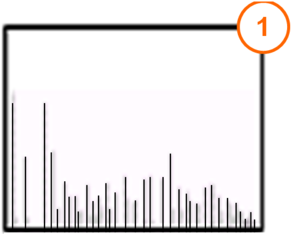
$$S_k = T(r_k) = \sum_{j=1}^{k} P_r(r_j) = \sum_{j=1}^{k} \frac{n_j}{n} \quad (1)$$

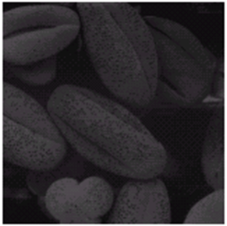
where

- r_k : input intensity
- s_k : processed intensity
- k : the intensity range (*e.g.*, 0.0 to 1.0)
- n_j : the frequency of intensity j
- n : the sum of all frequencies

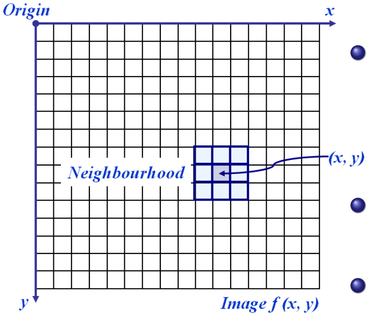
Equalisation Examples







Neighbourhood Operations



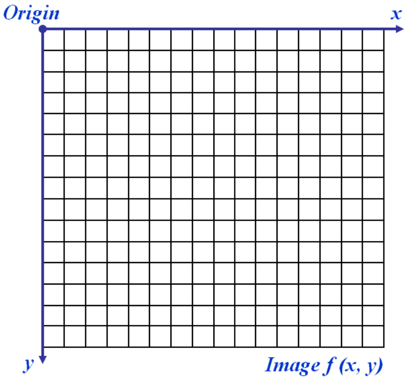
- Neighbourhood operations simply operate on a larger neighbourhood of pixels than point operations
- Neighbourhoods are mostly a rectangle around a central pixel
- Any size rectangle and any shape filter are possible

Simple Neighbourhood Operations

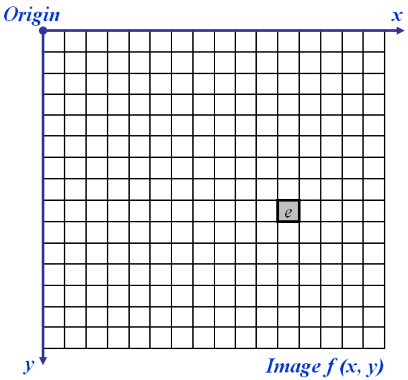
Some simple neighbourhood operations include:

1. Min: Set the pixel value to the minimum in the neighbourhood
2. Max: Set the pixel value to the maximum in the neighbourhood
3. Median: The median value of a set of numbers is the midpoint value in that set

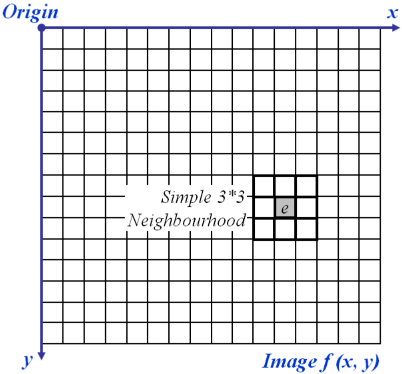
The Spatial Filtering Process



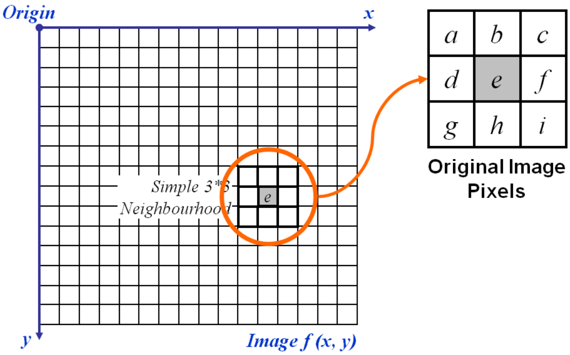
The Spatial Filtering Process



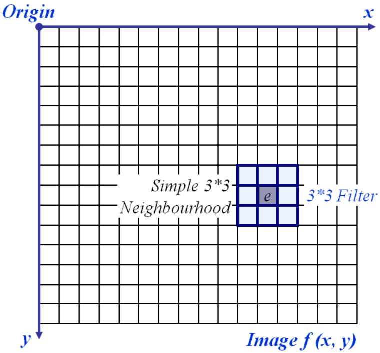
The Spatial Filtering Process

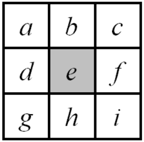


The Spatial Filtering Process



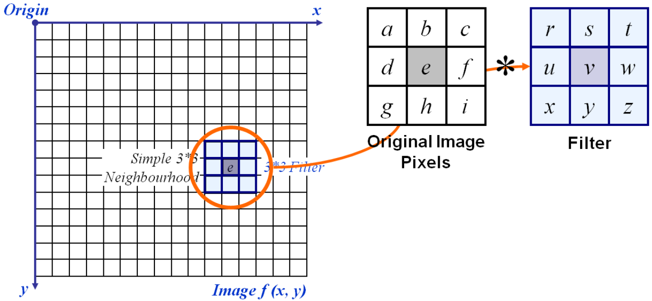
The Spatial Filtering Process



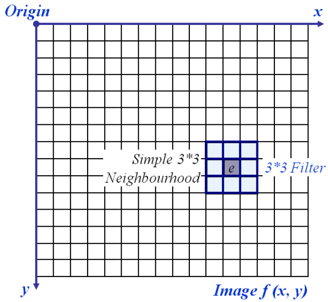


Original Image Pixels

The Spatial Filtering Process



The Spatial Filtering Process



a	b	c
d	e	f
g	h	i

Original Image Pixels

$*$

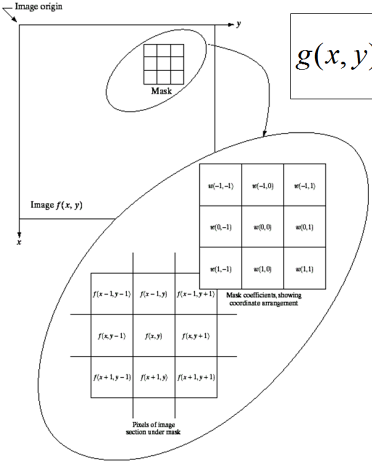
r	s	t
u	v	w
x	y	z

Filter

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

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

Outline

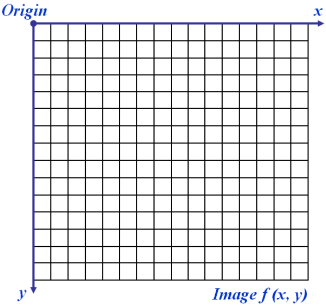
Smoothing Spatial Filters

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

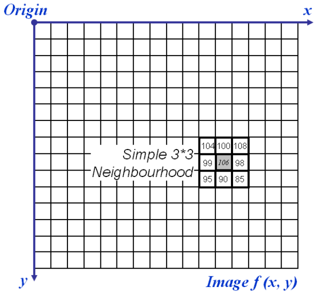
- Simply average all of the pixels in a neighbourhood around a central value
- Especially useful in removing noise from images
- Also useful for highlighting gross detail

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

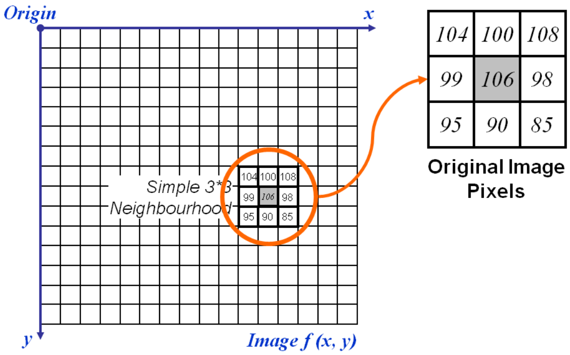
Smoothing Spatial Filtering



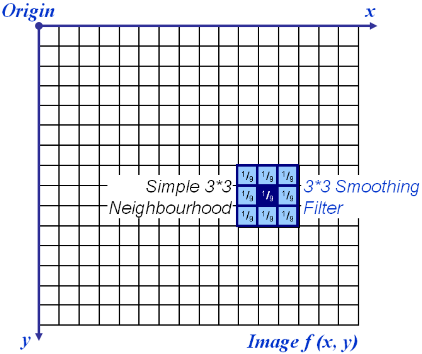
Smoothing Spatial Filtering



Smoothing Spatial Filtering



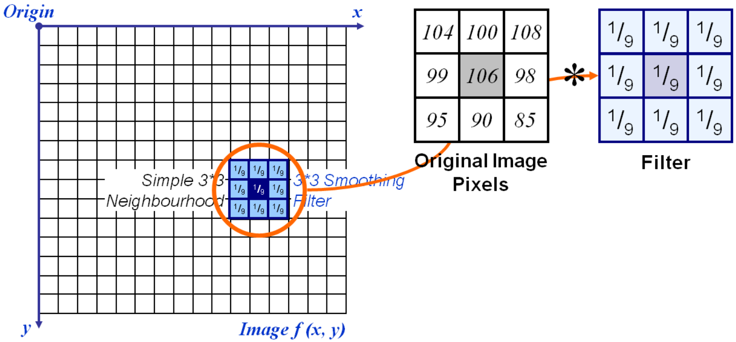
Smoothing Spatial Filtering



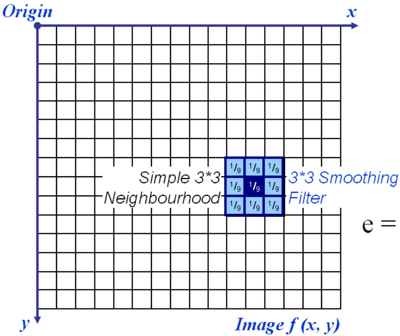
104	100	108
99	106	98
95	90	85

Original Image Pixels

Smoothing Spatial Filtering



Smoothing Spatial Filtering



104	100	108
99	106	98
95	90	85

Original Image Pixels

$*$

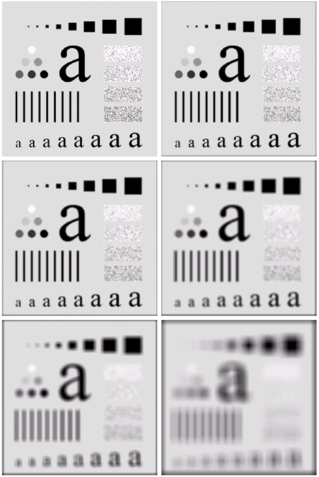
$^1/_9$	$^1/_9$	$^1/_9$
$^1/_9$	$^1/_9$	$^1/_9$
$^1/_9$	$^1/_9$	$^1/_9$

Filter

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

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

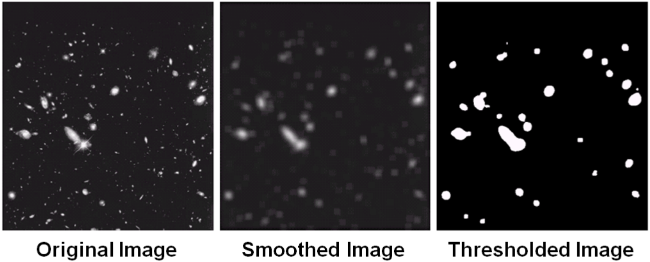
- Pixels closer to the central pixel are more important
- Often referred to as a weighted averaging

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

Weighted averaging filter

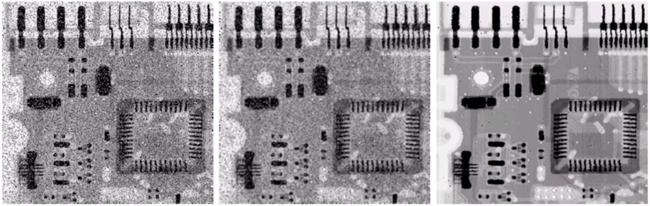
Another Smoothing Example

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



Averaging Filter vs Median Filter Example

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

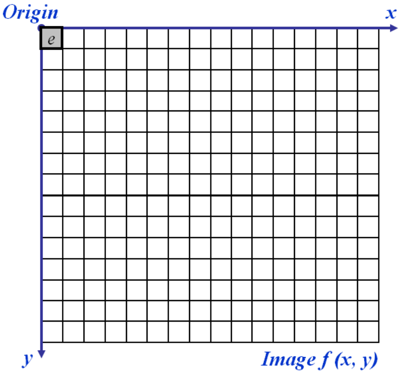


Original Image With Noise — Image After Averaging Filter — Image After Median Filter

Outline

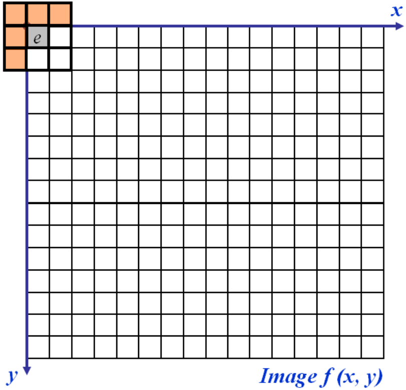
Strange Things Happen At The Edges

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



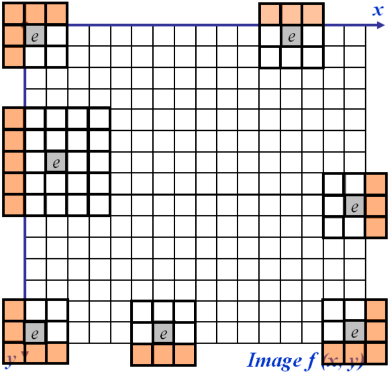
Strange Things Happen At The Edges

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



Strange Things Happen At The Edges

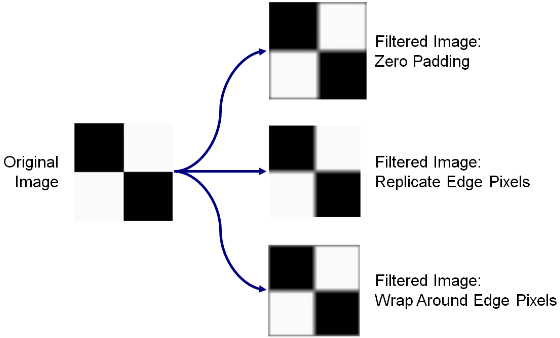
At the edges of an image we are missing pixels to form a neighbourhood



Dealing with Edges

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

Dealing with Edges — Examples



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

$$\begin{array}{|c|c|c|} \hline a & b & c \\ \hline d & e & e \\ \hline f & g & h \\ \hline \end{array} \;*\; \begin{array}{|c|c|c|} \hline r & s & t \\ \hline u & v & w \\ \hline x & y & z \\ \hline \end{array}$$

Original Image Pixels **Filter**

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

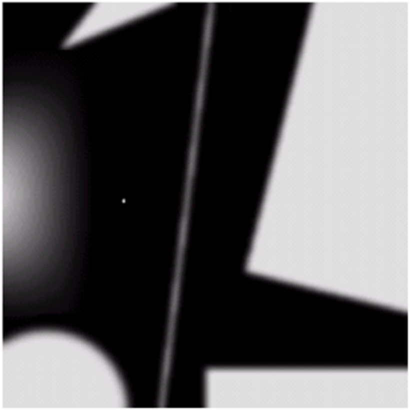
Sharpening Spatial Filters

Sharpening spatial filters seek to highlight fine detail

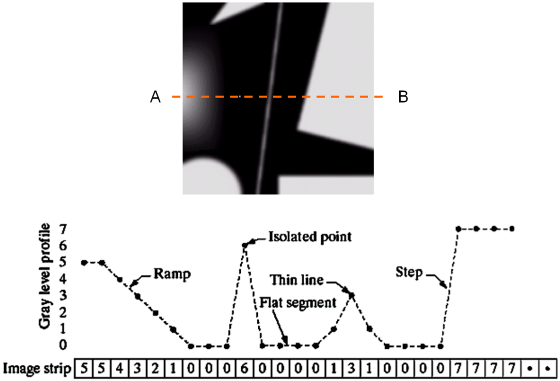
- Remove blurring from images
- Highlight edges
- Sharpening filters are based on spatial differentiation

Spatial Differentiation

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



Spatial Differentiation



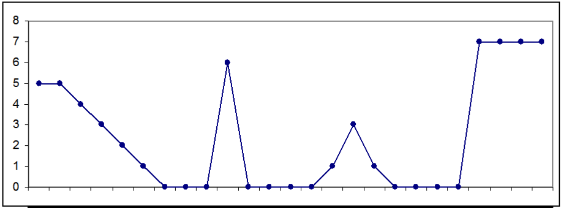
1^{st} Derivative

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

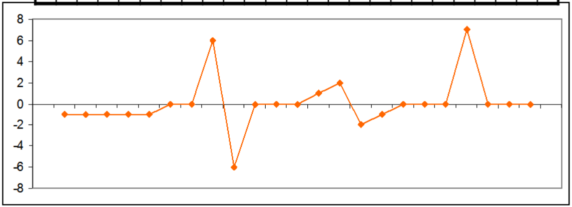
$$\frac{\partial f}{\partial x} = f(x+1) - f(x) \tag{2}$$

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

1^{st} Derivative



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	



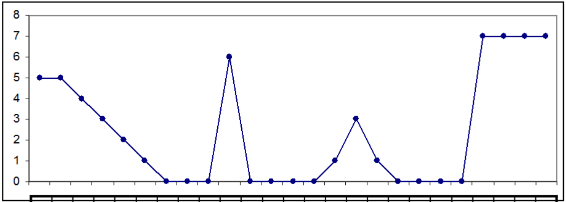
2^{nd} Derivative

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

$$\frac{\partial f^2}{\partial x^2} = f(x+1) + f(x-1) - 2f(x) \tag{3}$$

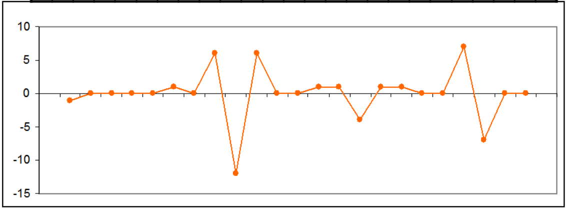
Simply takes into account the values both before and after the current value

2^{nd} Derivative



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	



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

The Laplacian

The Laplacian is defined as follows:

$$\bigtriangledown^2 f = \frac{\partial^2 f}{\partial^2 x} + \frac{\partial^2 f}{\partial^2 y} \tag{4}$$

where partial 1^{st} Derivative in the x direction is defined as follows:

$$\frac{\partial^2 f}{\partial^2 x} = f(x+1, y) + f(x-1, y) - 2f(x, y) \tag{5}$$

and in the y direction as follows:

$$\frac{\partial^2 f}{\partial^2 y} = f(x, y+1) + f(x, y-1) - 2f(x, y) \tag{6}$$

The Laplacian

So, the Laplacian can be built as:

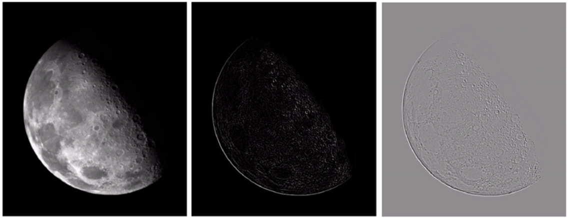
$$\bigtriangledown^2 f = [f(x+1,y)+f(x-1,y)+f(x,y+1)+f(x,y-1)]-4f(x,y) \tag{7}$$

We can easily build a filter based on this

0	1	0
1	-4	1
0	1	0

The Laplacian Example

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



Original Image | Laplacian Filtered Image | Laplacian Filtered Image Scaled for Display

But That Is Not Very Enhanced!

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

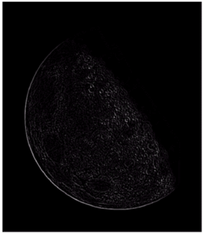
$$g(x, y) = f(x, y) - \bigtriangledown^2 f \qquad (8)$$

Laplacian
Filtered Image
Scaled for Display

The Laplacian Example

Original Image

-



Laplacian Filtered Image

=

Sharpened Image

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

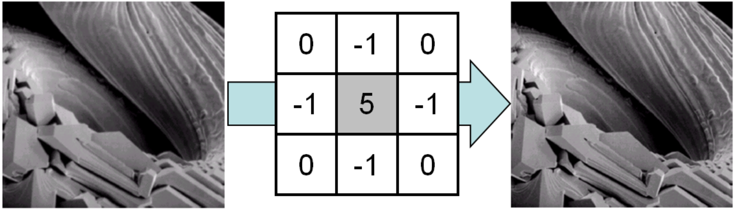
Simplified Image Enhancement

The entire enhancement can be combined into a single filtering operation

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

Simplified Image Enhancement

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



Variants On The Simple Laplacian

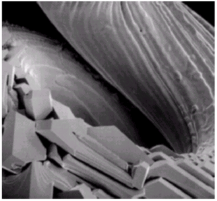
There are lots of slightly different versions of the Laplacian that can be used:

Simple Laplacian

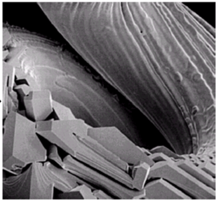
0	1	0
1	-4	1
0	1	0

Variant of Laplacian

1	1	1
1	-8	1
1	1	1



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



1^{st} Derivative Filtering

- Implementing 1^{st} 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:

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

1^{st} Derivative Filtering

The magnitude of this vector is given by:

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

For practical reasons this can be simplified as:

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

1^{st} Derivative Filtering

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

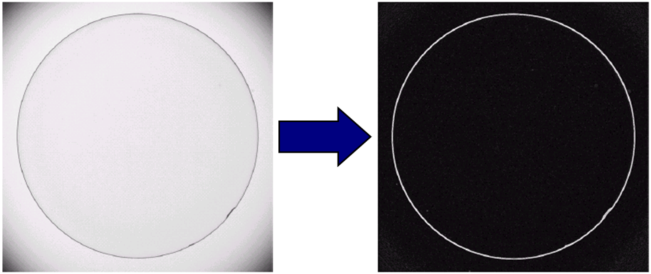
Based on the previous equations we can derive the Sobel Operators

-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 Operators



1^{st} and 2^{nd} Derivative Comparison

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

1. 1^{st} order derivatives generally produce thicker edges
2. 2^{nd} order derivatives have a stronger response to fine detail *e.g.*, thin lines
3. 1^{st} order derivatives have stronger response to grey level step
4. 2^{nd} order derivatives produce a double response at step changes in grey level