

# Image Enhancement (Spatial Filtering 1)

In this lecture we will look at spatial filtering techniques:

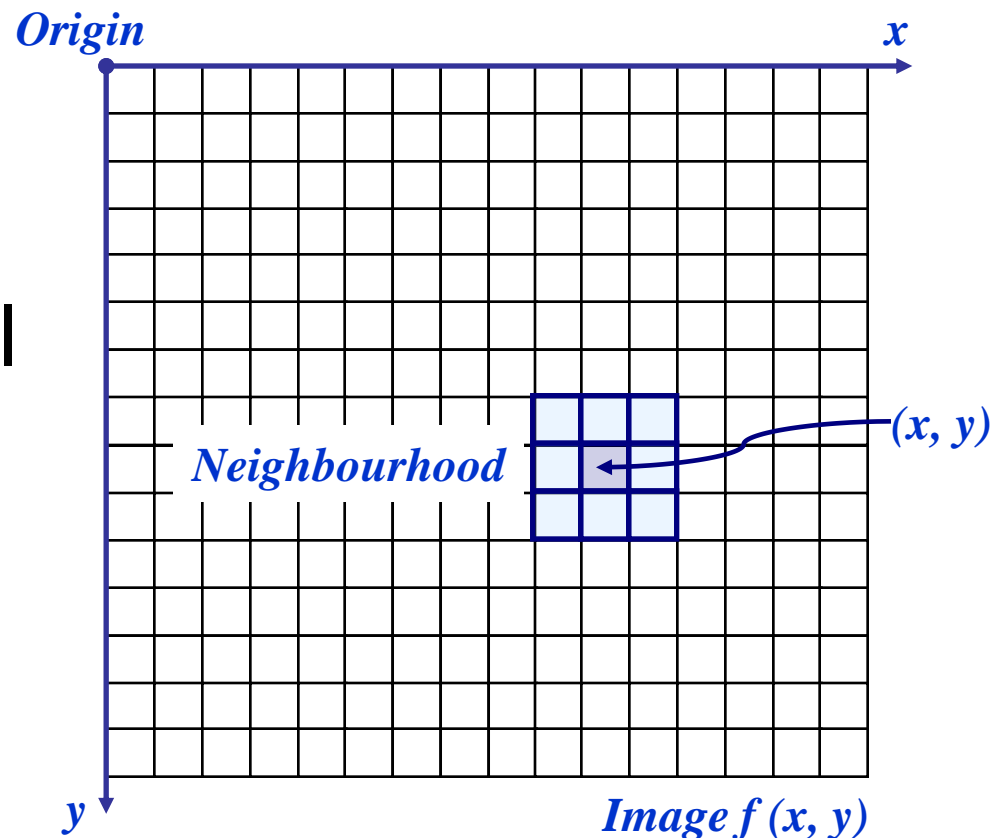
- Neighbourhood operations
- What is spatial filtering?
- Smoothing operations
- What happens at the edges?
- Correlation and convolution

# 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

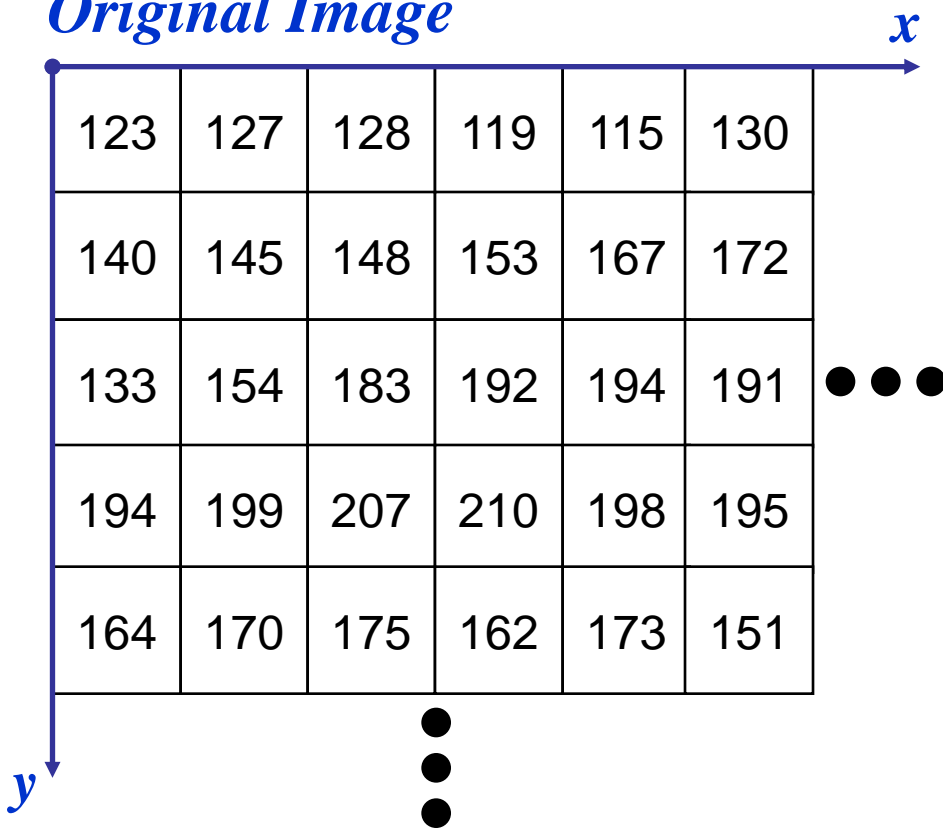
Some simple neighbourhood operations include:

- **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

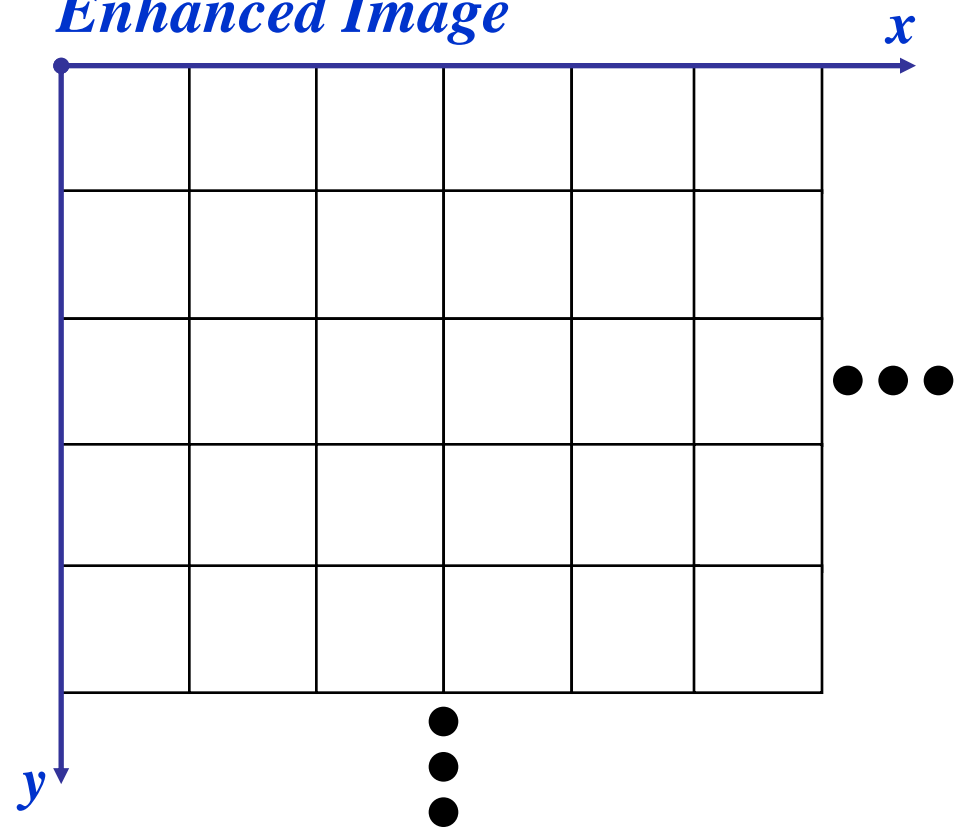
*Original Image*



A 5x6 grid of numerical values representing an original image. The grid is labeled with a blue 'x' axis pointing right and a blue 'y' axis pointing down. To the right of the grid, there are three black dots indicating continuation. Below the grid, there are three black dots indicating continuation.

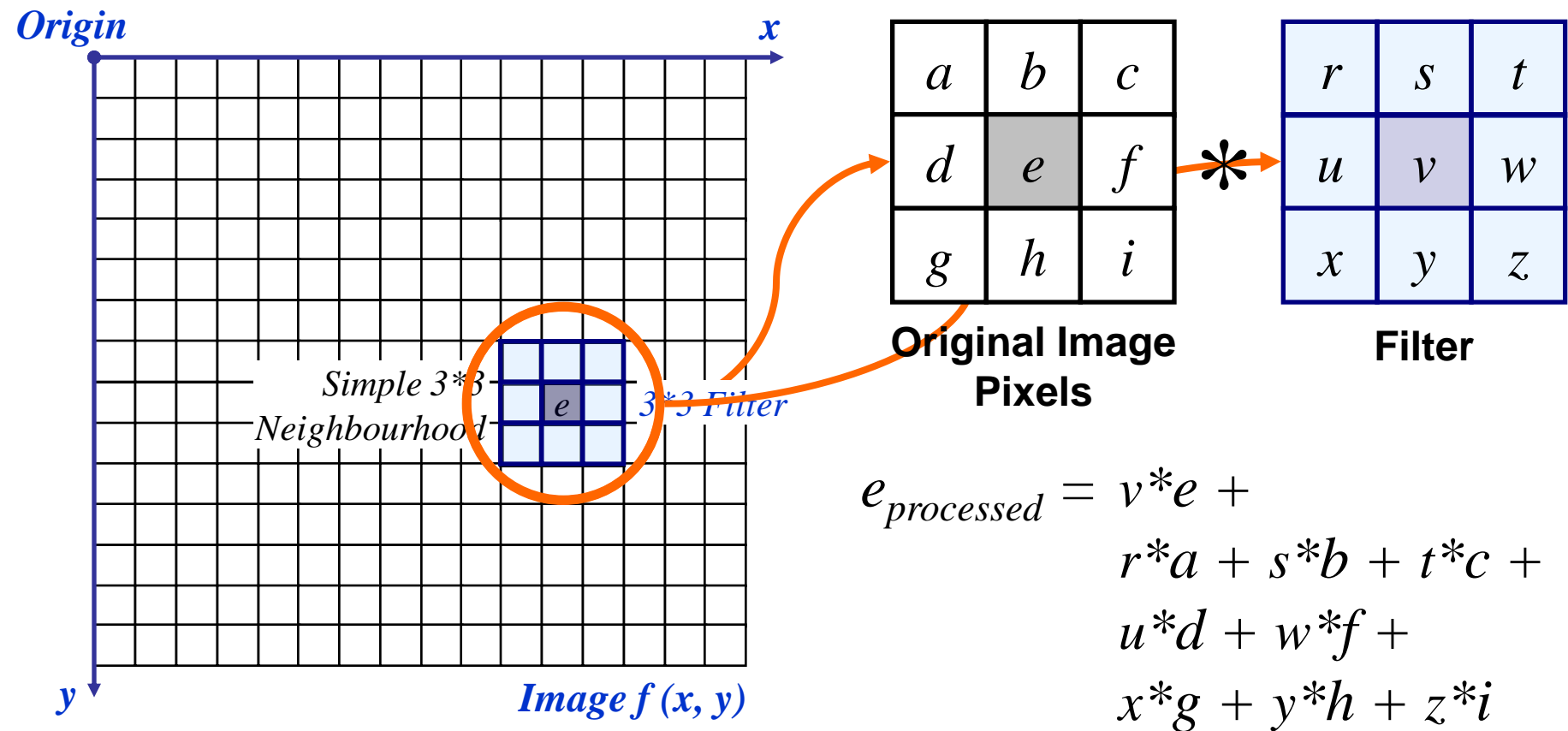
123	127	128	119	115	130
140	145	148	153	167	172
133	154	183	192	194	191
194	199	207	210	198	195
164	170	175	162	173	151

*Enhanced Image*



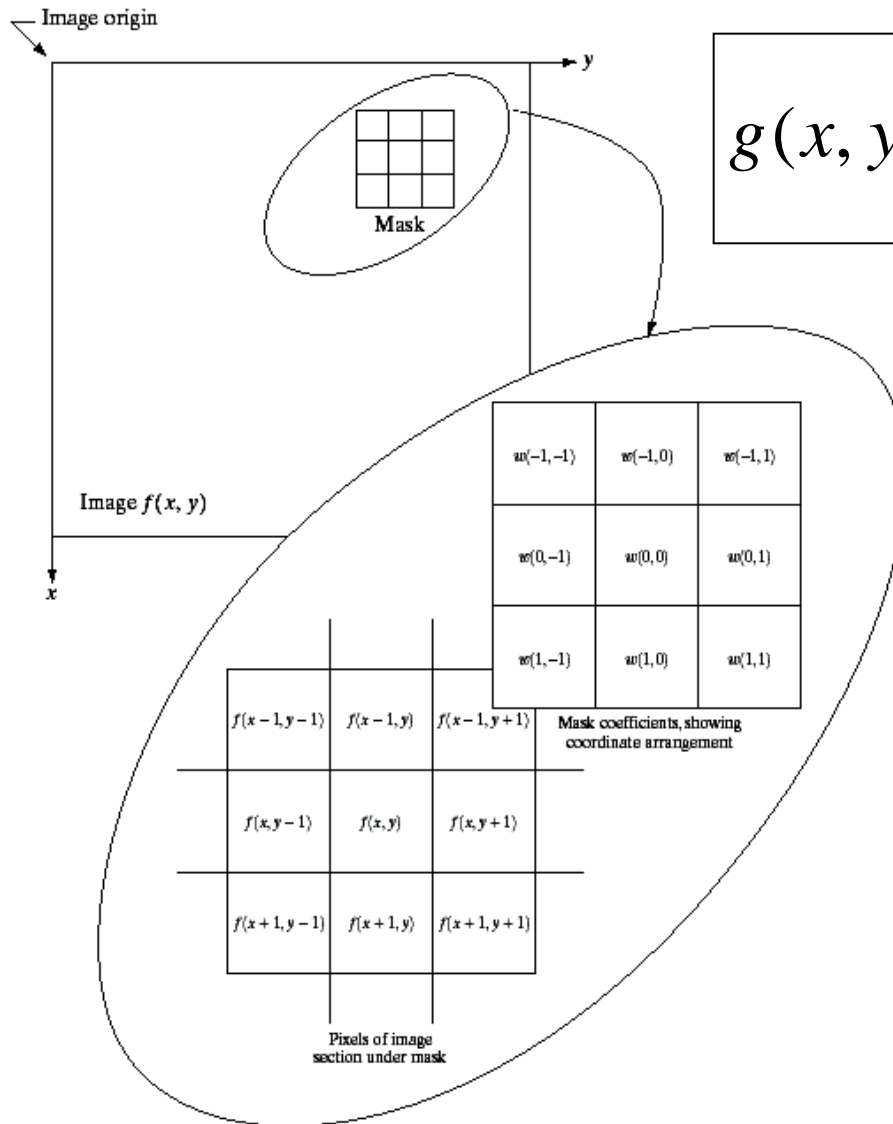
A 5x6 grid of empty cells representing an enhanced image. The grid is labeled with a blue 'x' axis pointing right and a blue 'y' axis pointing down. To the right of the grid, there are three black dots indicating continuation. Below the grid, there are three black dots indicating continuation.


# The Spatial Filtering Process



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

# Smoothing Spatial Filters

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

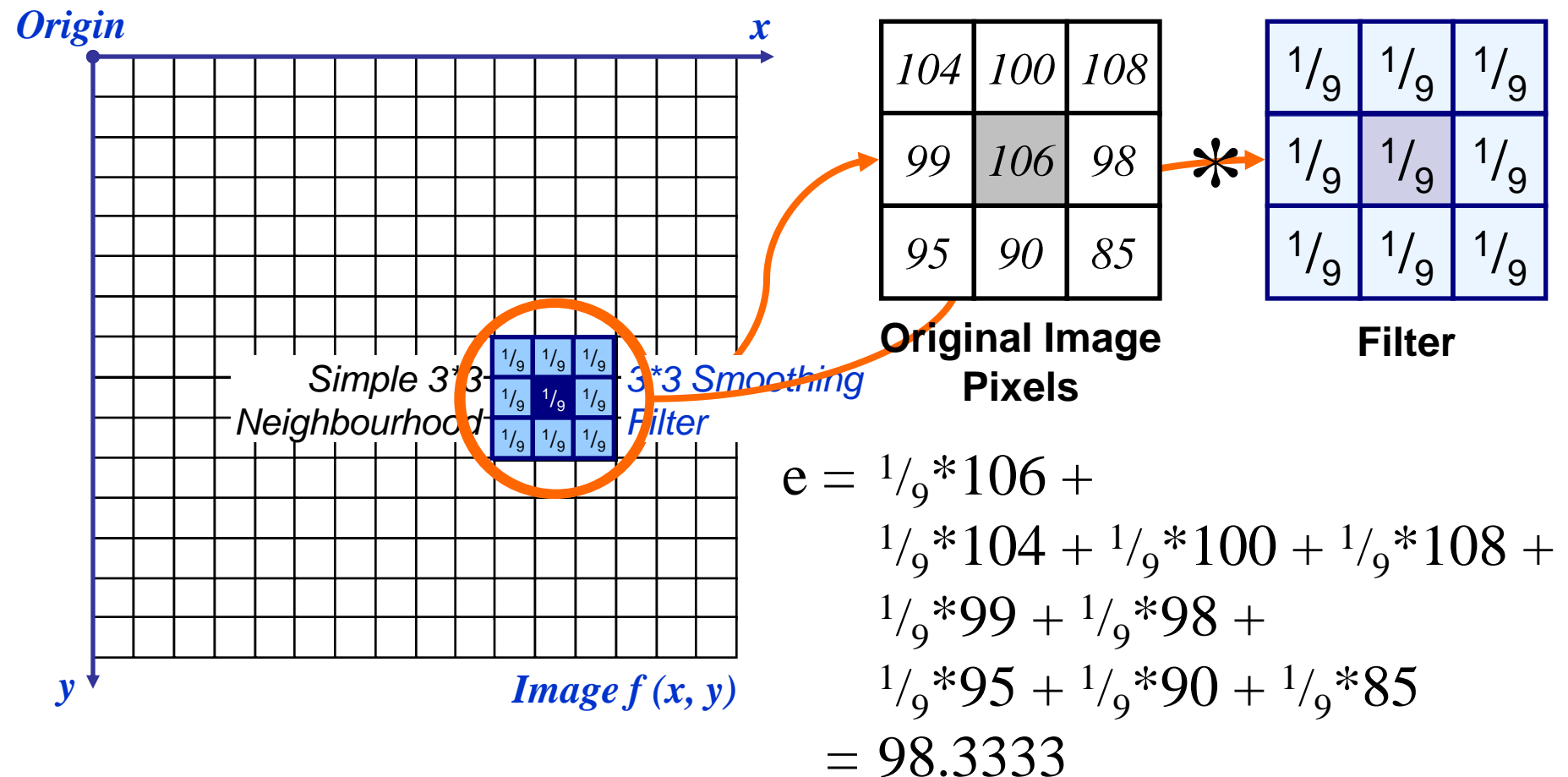
- Simply average all of the pixels in a neighbourhood around a central value
- Especially useful in removing noise from images
- Also useful for highlighting gross detail

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

Simple  
averaging  
filter



# Smoothing Spatial Filtering



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

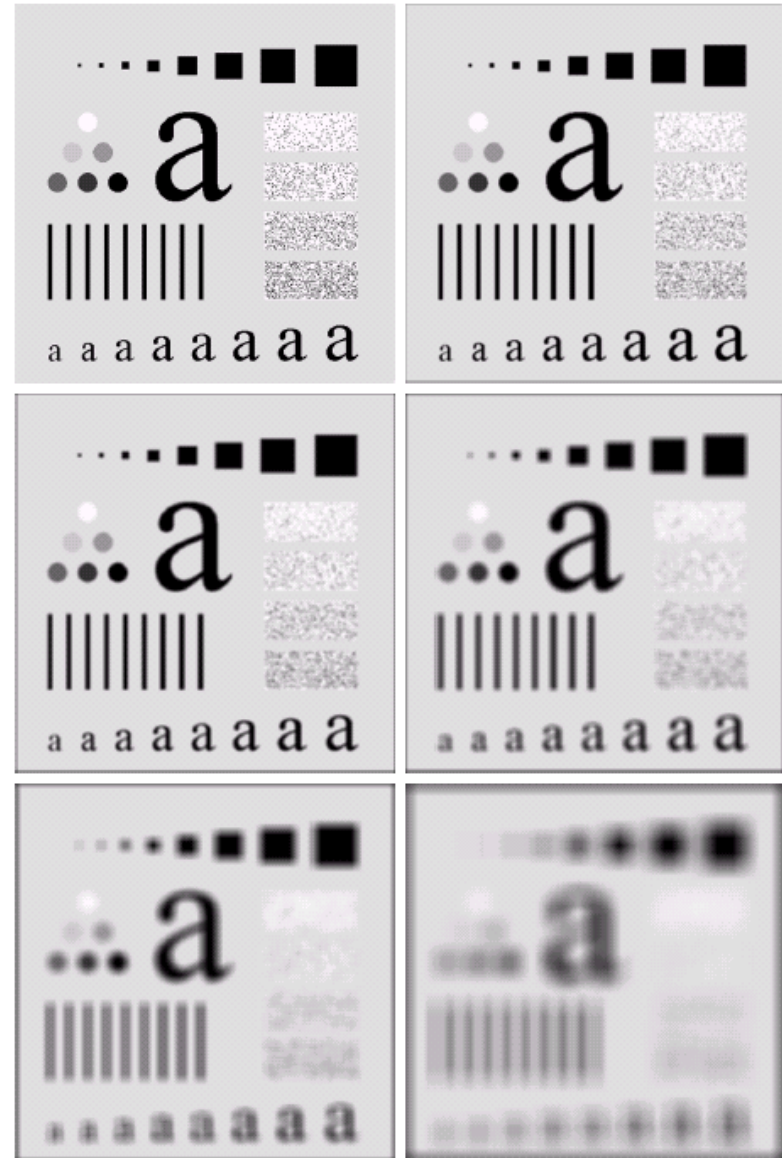
# Image Smoothing Example

The image at the top left is an original image of size 500\*500 pixels

The subsequent images show the image after filtering with an averaging filter of increasing sizes

– 3, 5, 9, 15 and 35

Notice how detail begins to disappear



# Weighted Smoothing Filters

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

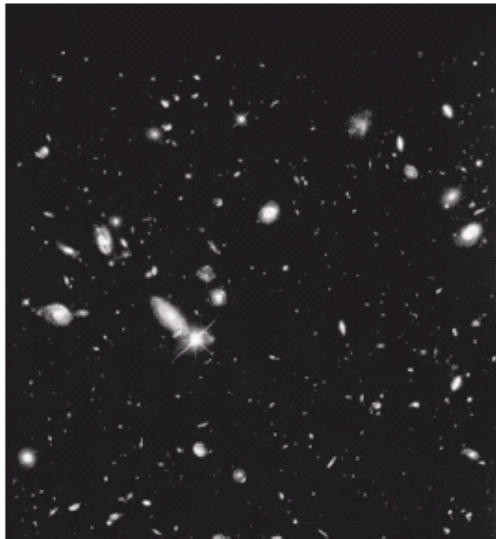
- Pixels closer to the central pixel are more important
- Often referred to as a *weighted averaging*

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

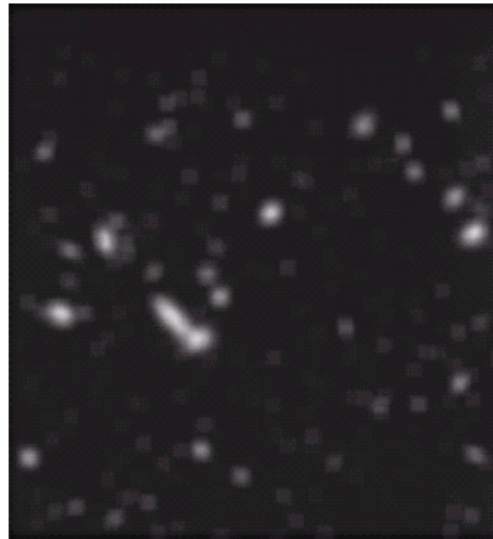
Weighted  
averaging filter

# 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**

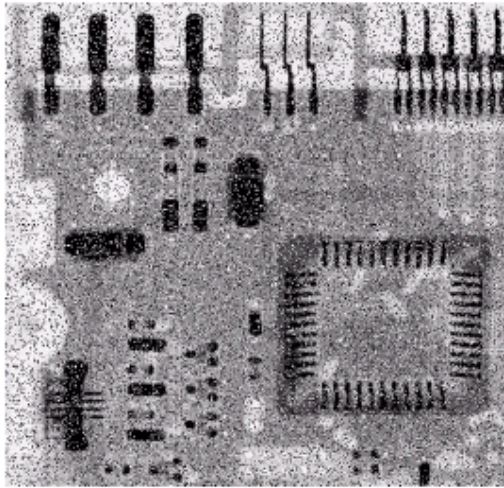


**Smoothed Image**

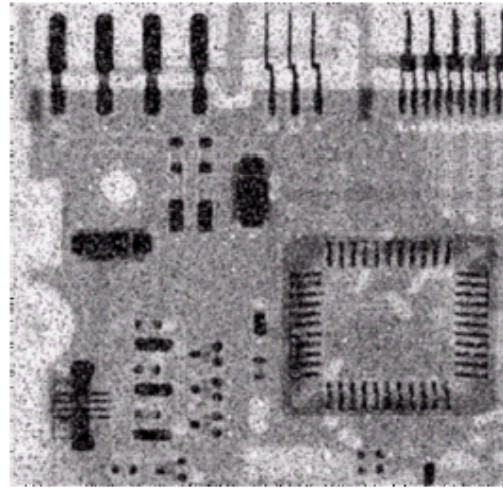


**Thresholded Image**

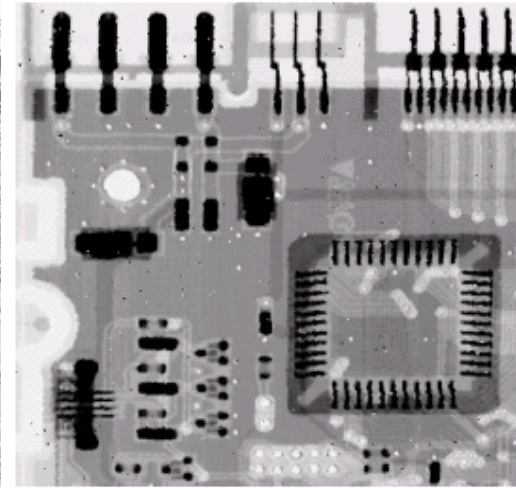
# Averaging Filter Vs. Median Filter Example



**Original Image  
With Noise**



**Image After  
Averaging Filter**



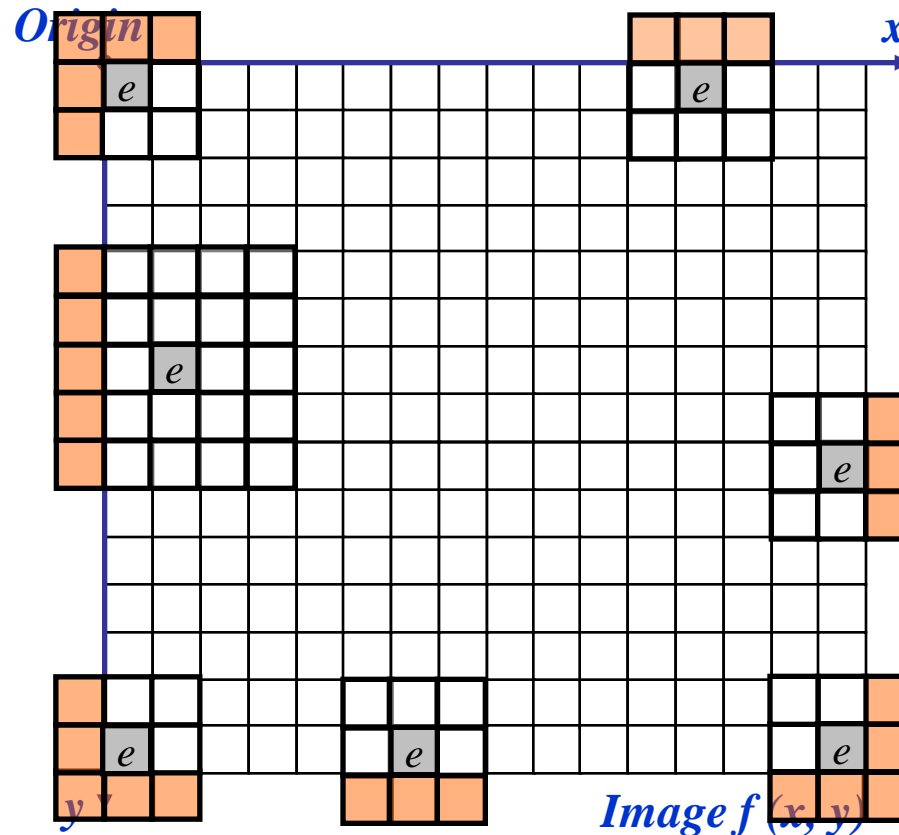
**Image After  
Median Filter**

Filtering is often used to remove noise from images

Sometimes a median filter works better than an averaging filter

# 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:

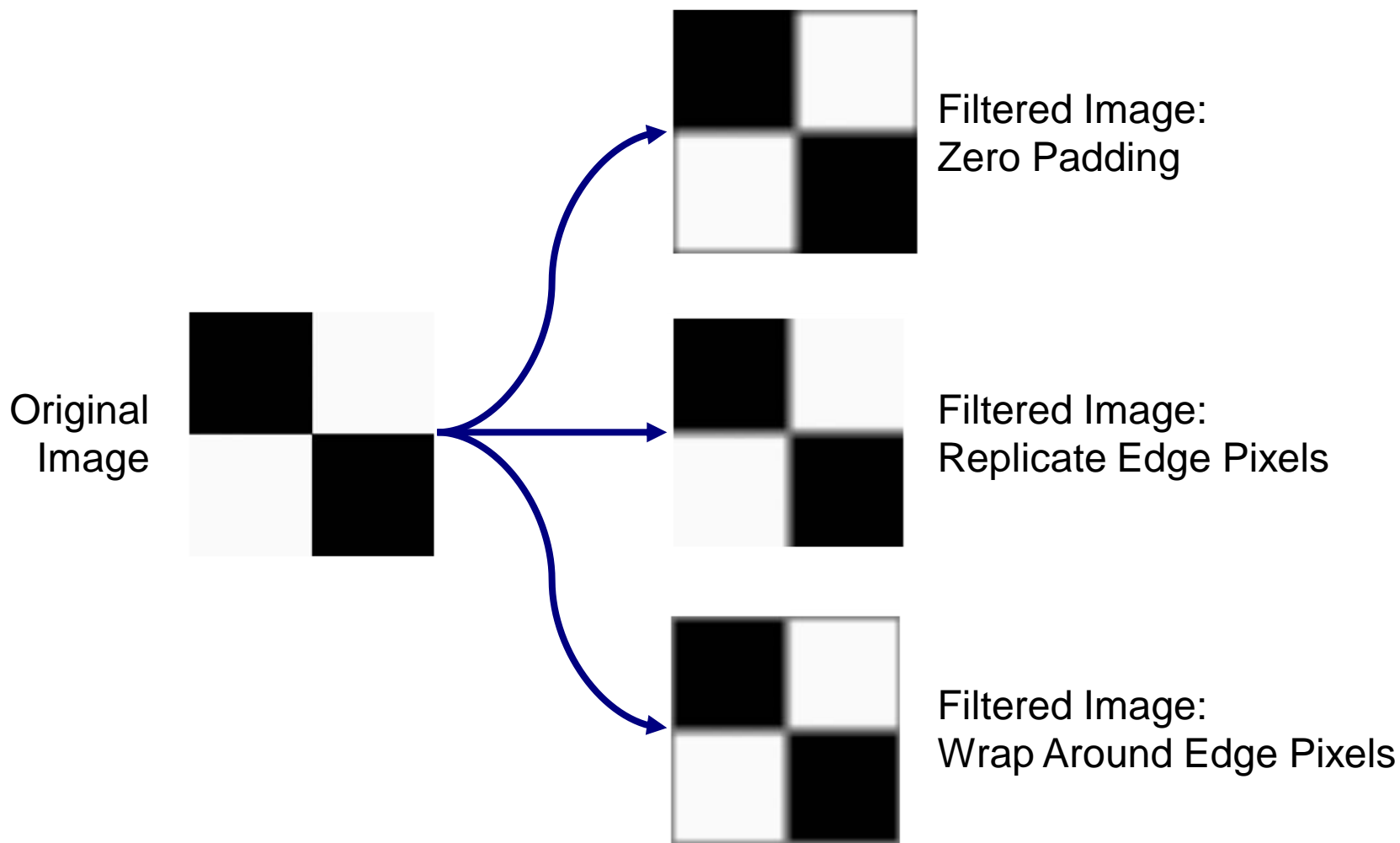
- Omit missing pixels
  - Only works with some filters
  - Can add extra code and slow down processing
- Pad the image
  - Typically with either all white or all black pixels
- Replicate border pixels
- Truncate the image
- Allow pixels *wrap around* the image
  - Can cause some strange image artefacts

# 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...)



# Correlation & Convolution

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

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

$a$	$b$	$c$
$d$	$e$	$e$
$f$	$g$	$h$

Original Image  
Pixels

$*$

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

Filter

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

For symmetric filters it makes no difference

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