# Image Enhancement

Spatial Filtering 1

# Contents

In this session we will look at spatial filtering techniques:

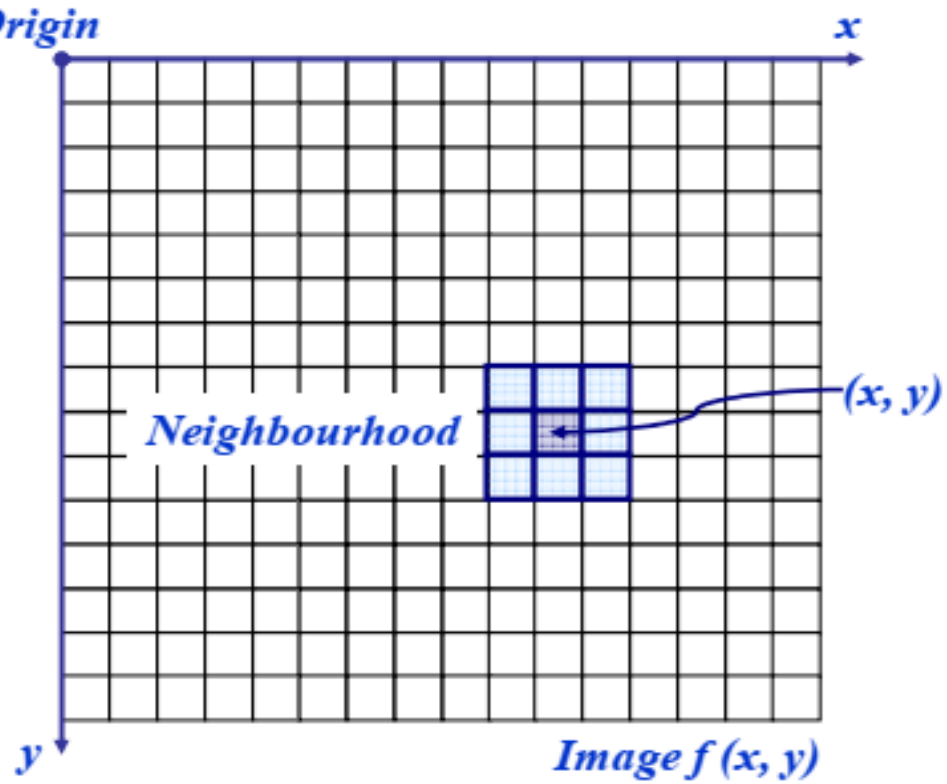
- Neighbourhood operations
- What is spatial filtering?
- Smoothing operations
- What happens at the edges?
- Unsharp Masking

# Neighborhood Operations

Neighbourhood operations simply operate on a larger neighbourhood of pixels than point operations

Neighbourhoods are mostly a rectangle around a central pixel

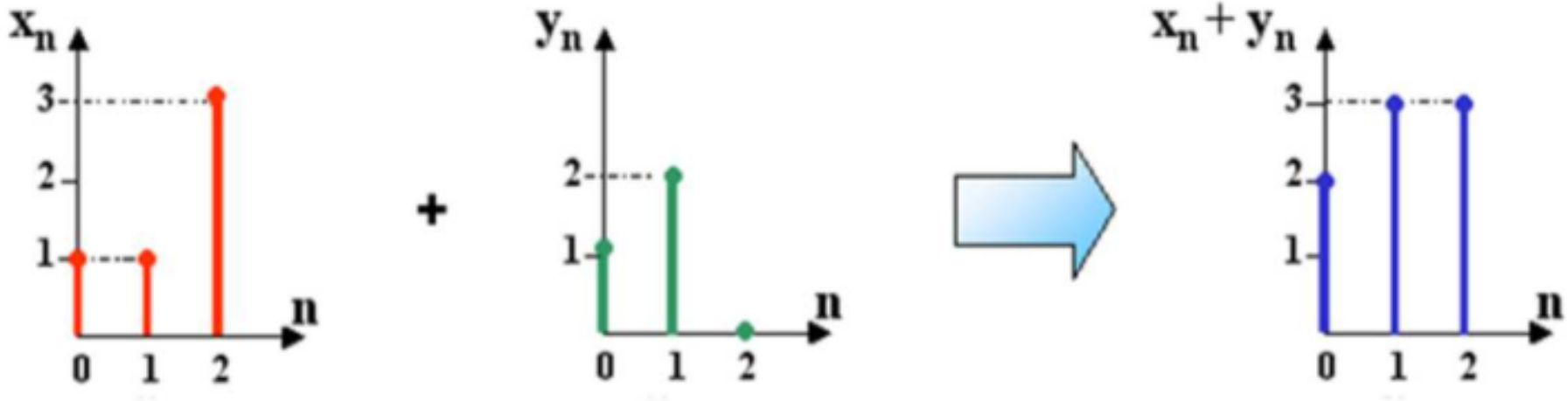
Any size rectangle and any shape filter are possible



# Linear vs Non Linear Filters

In mathematics, a linear map or linear function $f(x)$ is a function that satisfies the following two properties:

- Additivity: $f(x + y) = f(x) + f(y)$.
- Homogeneity of degree 1: $f(\alpha x) = \alpha f(x)$ for all $\alpha$.



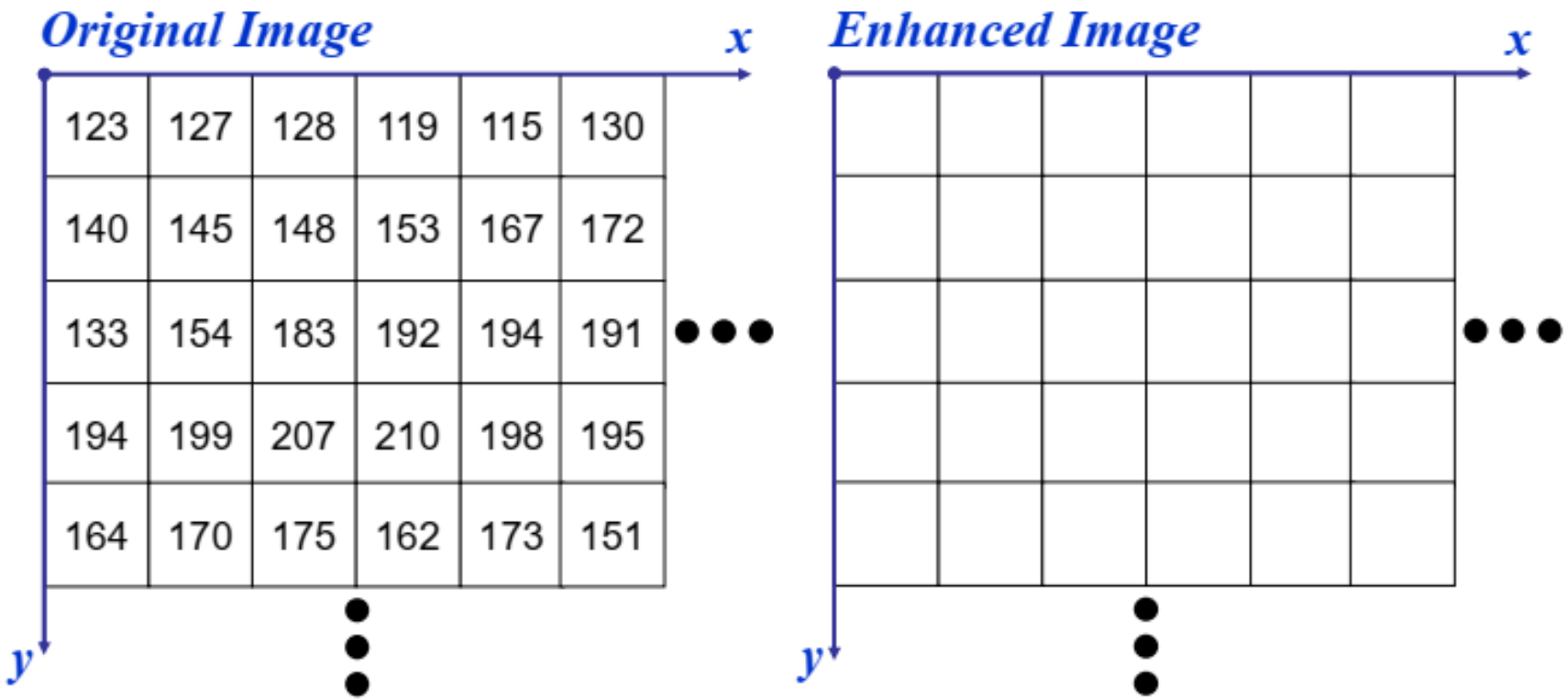

1. **mean(Xn)+mean(Yn)=mean(Xn+Yn)**
   8/3=1+5/3 **which is OK, so mean is linear function.**
2. **median(Xn)+median(Yn)=median(Xn+Yn),**
   1+1=3 **which is not true, median is non linear.**

# Simple Neighborhood Operations

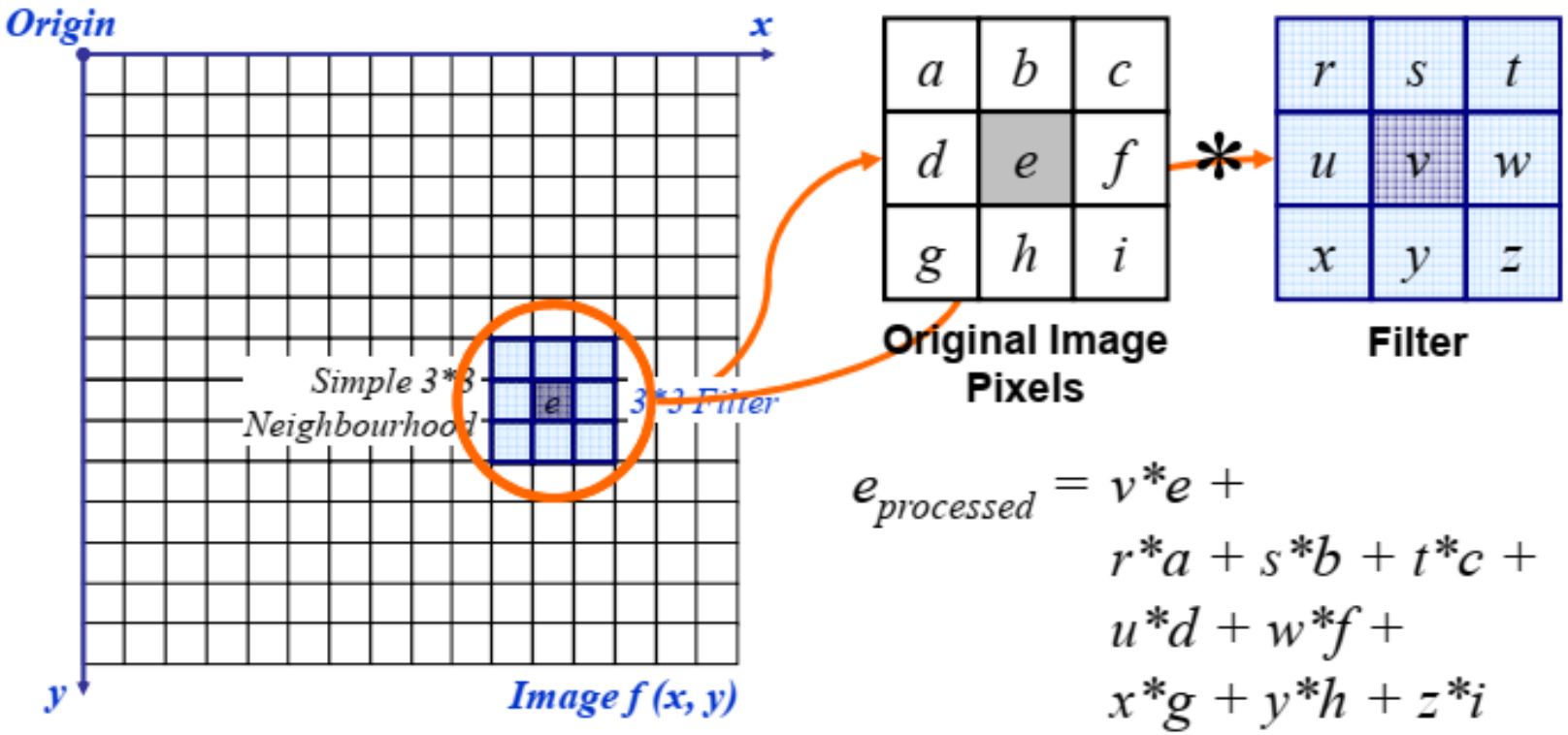
Some simple **non linear** neighbourhood operations include:

- **Min (dilation):** Set the pixel value to the minimum in the neighbourhood
- **Max (erosion):** 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 Neighborhood Operations



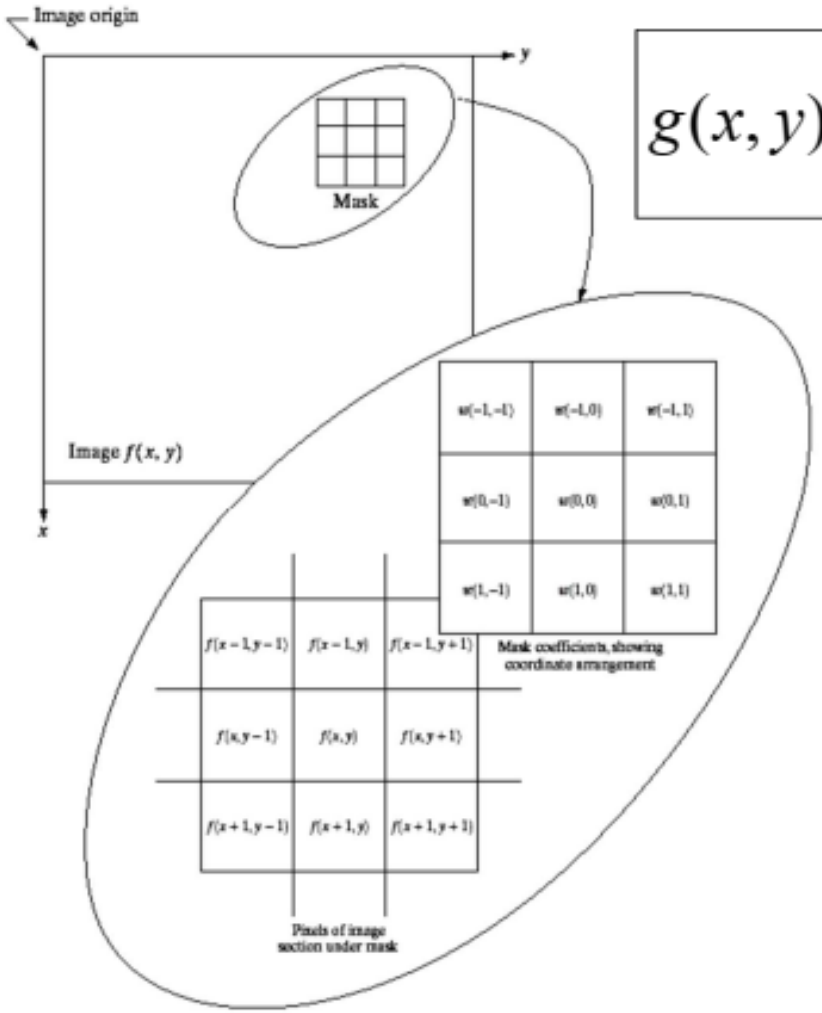
# The Spatial Filtering Process




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

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

# Spatial Filtering: Equation Form




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

Filtering can be given in equation form as shown above

Notations are based on the image shown to the left

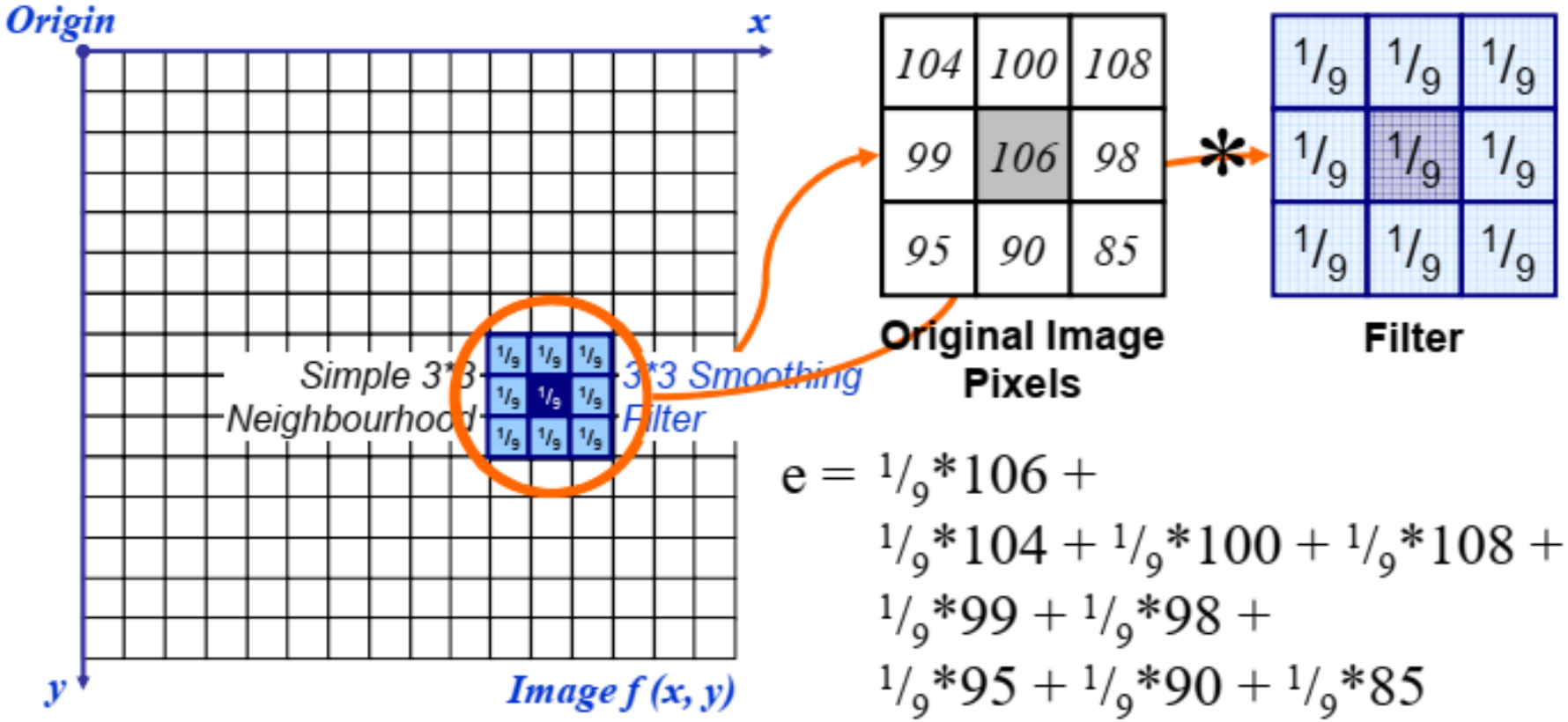
# Smoothing Spatial Filters

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




$$e = 1/9*106 + 1/9*104 + 1/9*100 + 1/9*108 + 1/9*99 + 1/9*98 + 1/9*95 + 1/9*90 + 1/9*85 = 98.3333$$

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

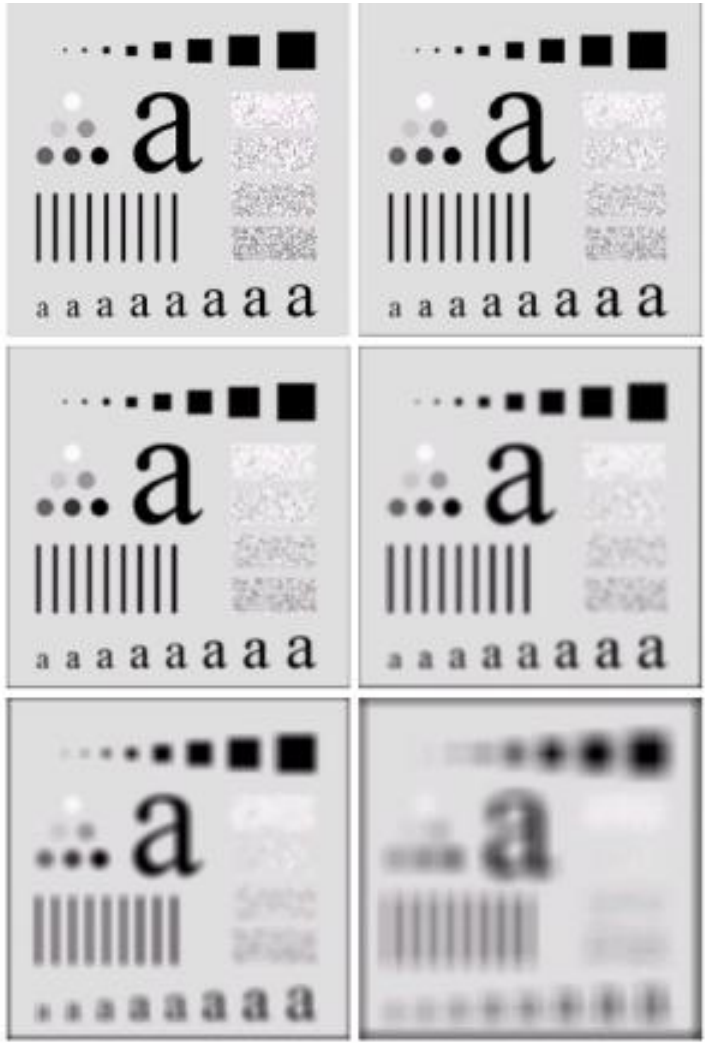
# Image Smoothing Example

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

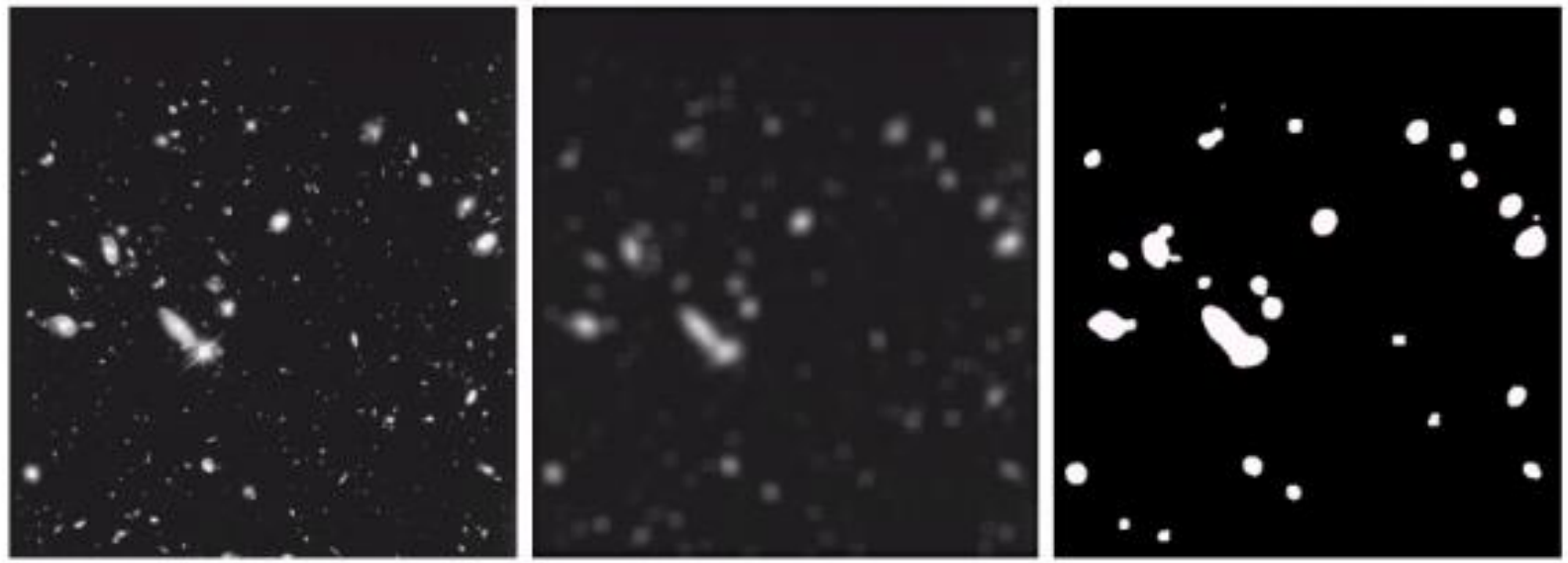
More effective smoothing filters can be generated by allowing different pixels in the neighbourhood different weights in the averaging function

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

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

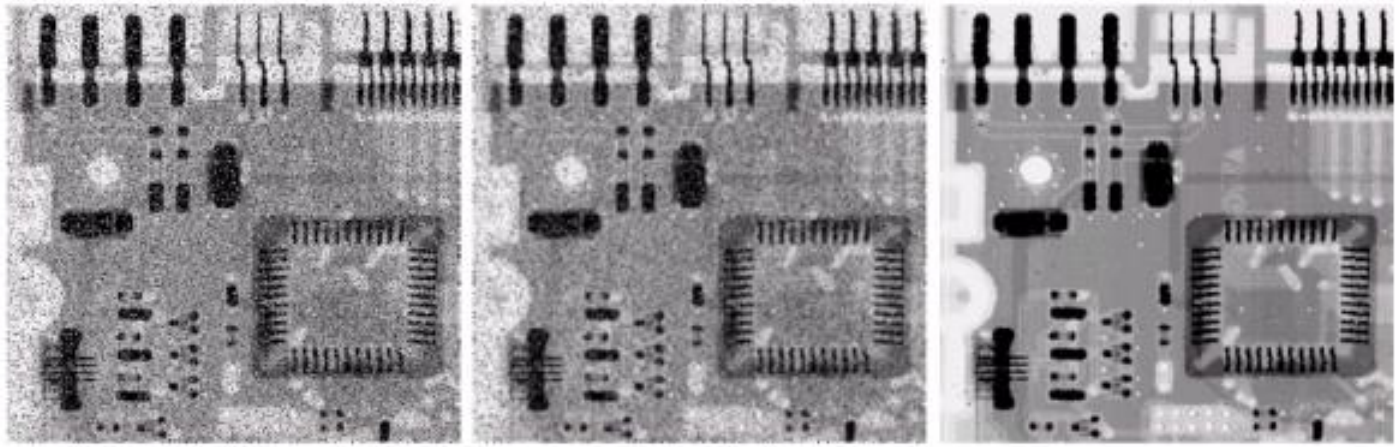
Weighted averaging filter

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

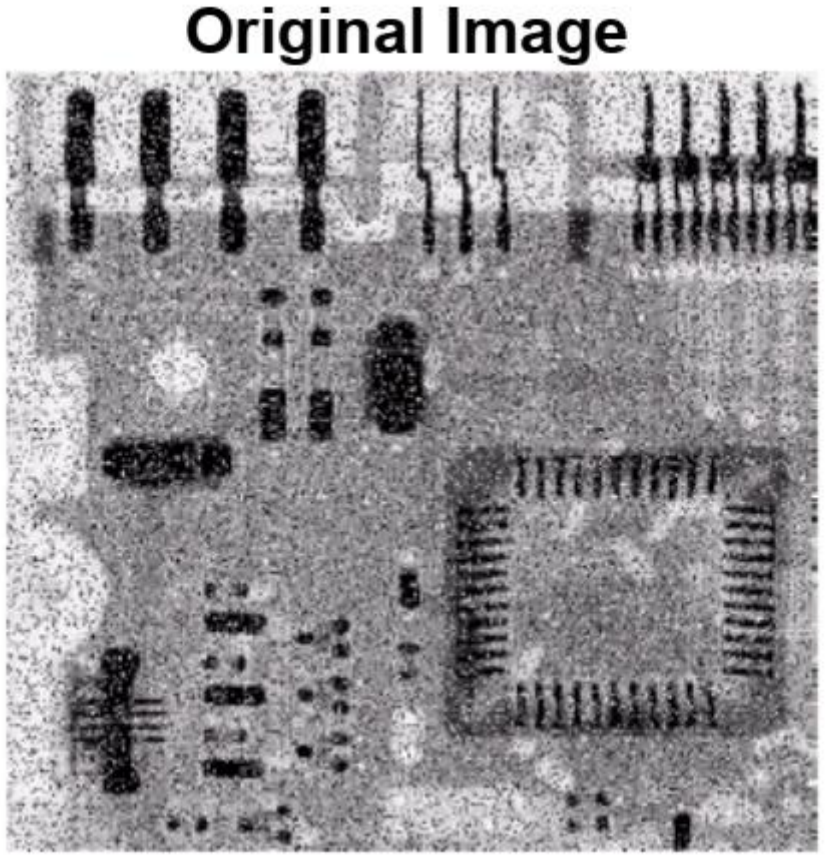


Original Image With Noise | Image After Averaging Filter | Image After Median Filter

Filtering is often used to remove noise from images

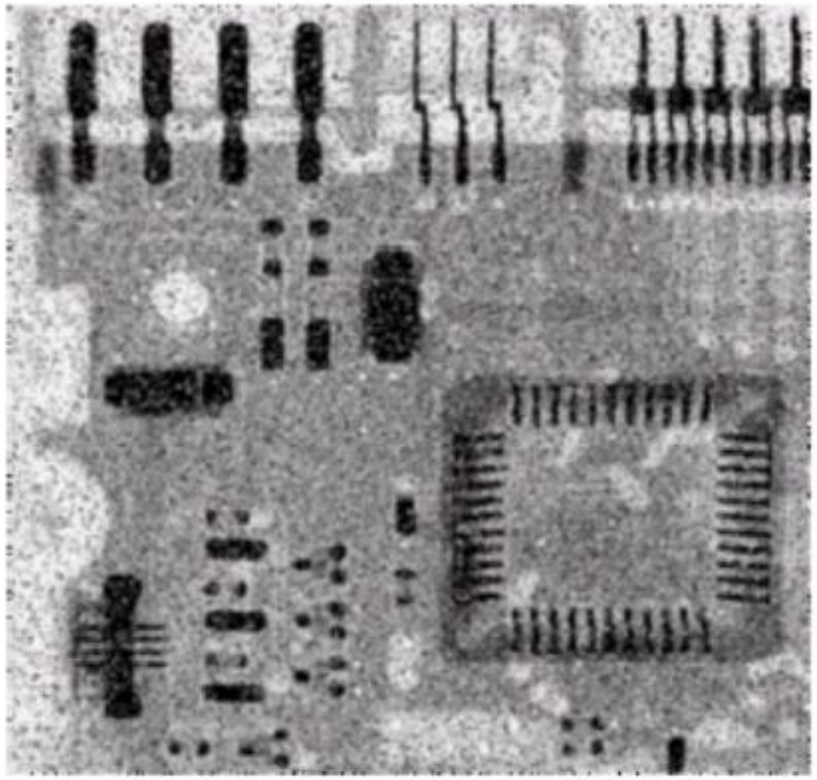
Sometimes a median filter works better than an averaging filter

# Averaging Filter Vs. Median Filter Example

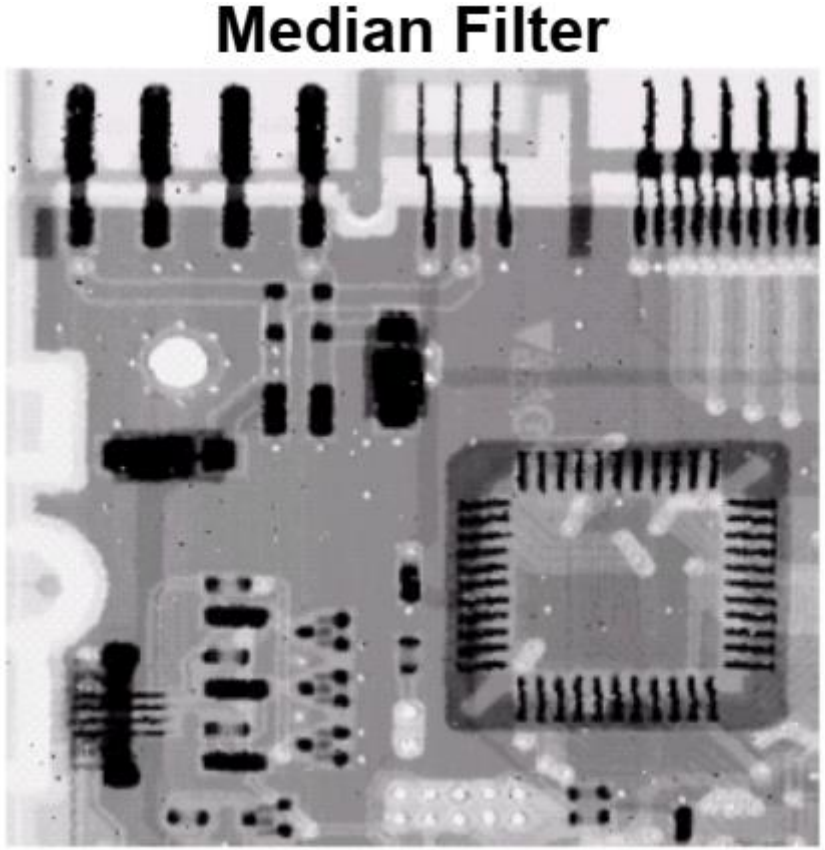


# Averaging Filter Vs. Median Filter Example

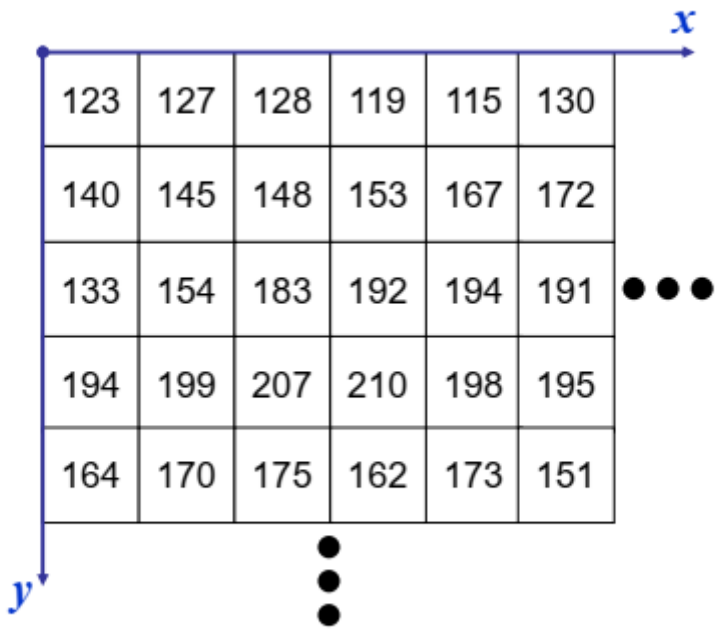
**Average Filter**



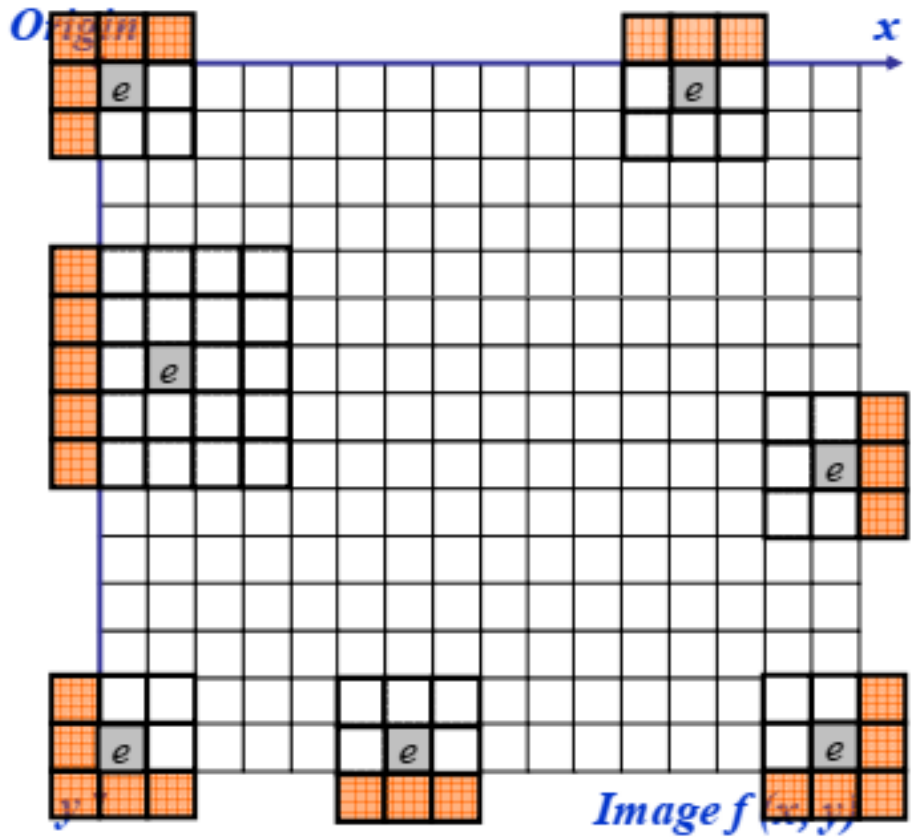
# Averaging Filter Vs. Median Filter Example



# Simple Neighbourhood Operations Example



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

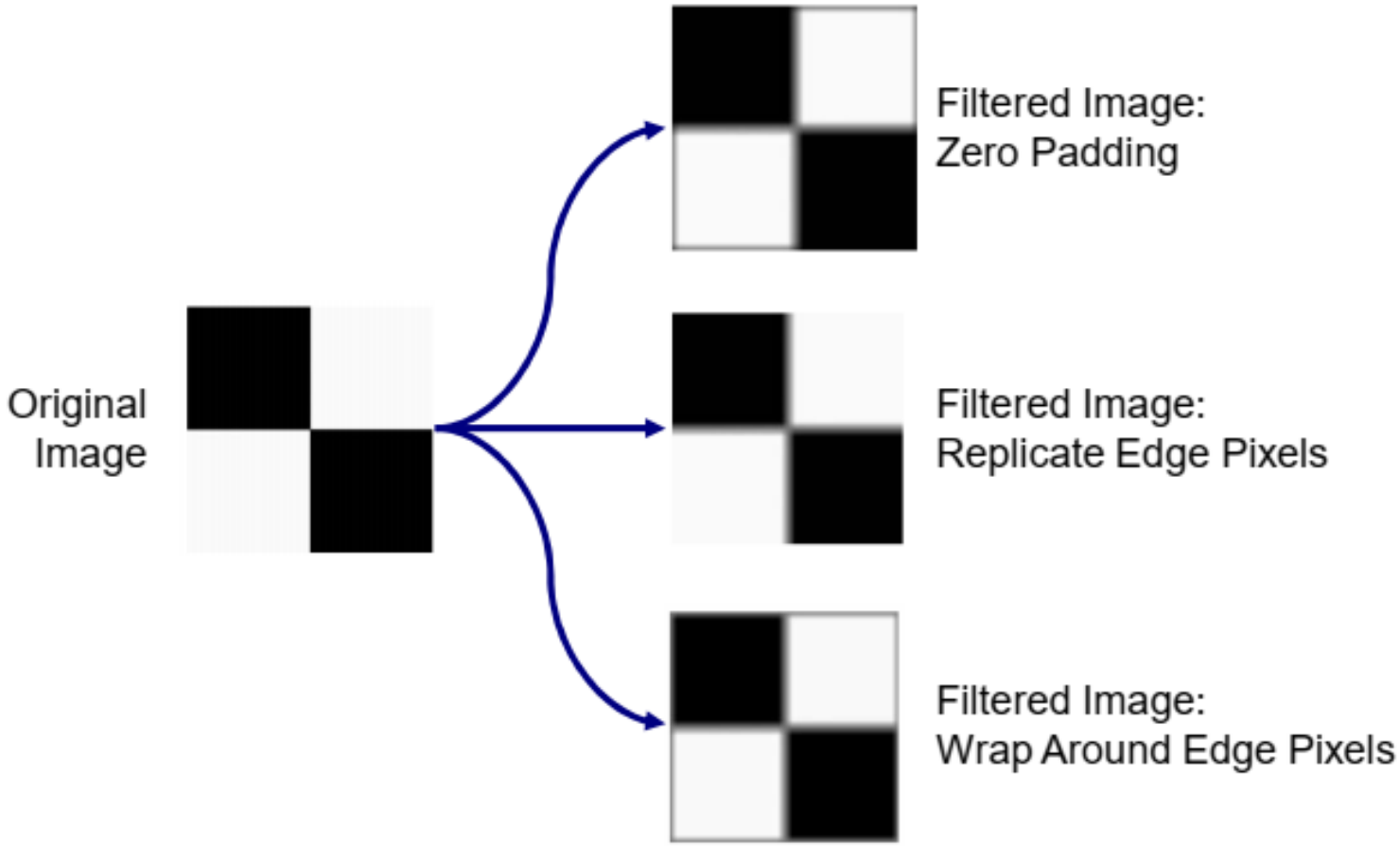


# Strange Things Happen At The Edges!

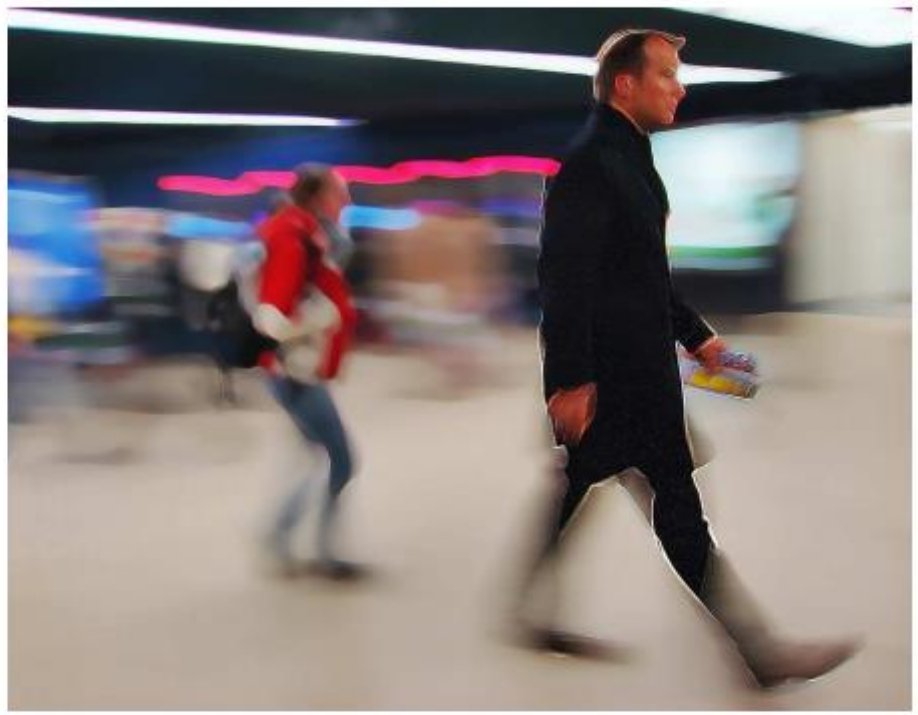
There are a few approaches to dealing with missing edge pixels:

- 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

# Strange Things Happen At The Edges!



# sharpen !

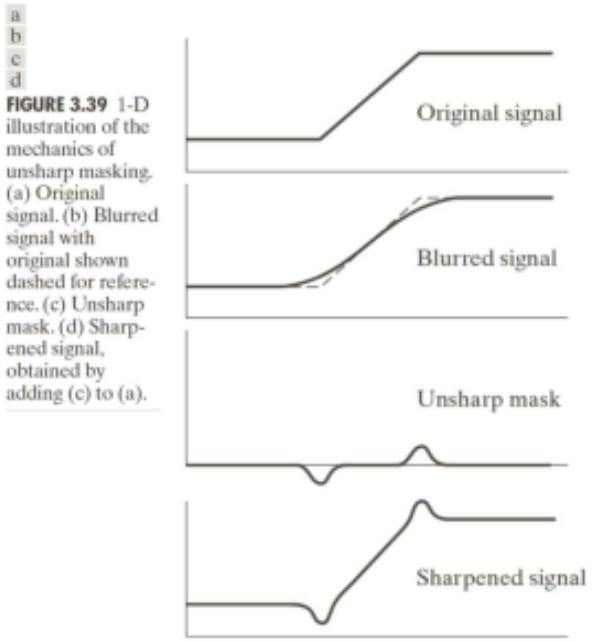


Unlike smoothing the principal objective of **sharpening** is to **highlight transition in intensity**.

http://flickr.com/photos/t_schnitzlein/87607390/

# Unsharp masking

- **Unsharp masking** is an image manipulation technique for increasing the apparent sharpness of photographic images.
- The "unsharp" of the name derives from the fact that the technique uses a blurred, or "unsharp", positive to create a "mask" of the original image. The unsharped mask is then combined with the negative, creating a resulting image sharper than the original.




a
b
c
d

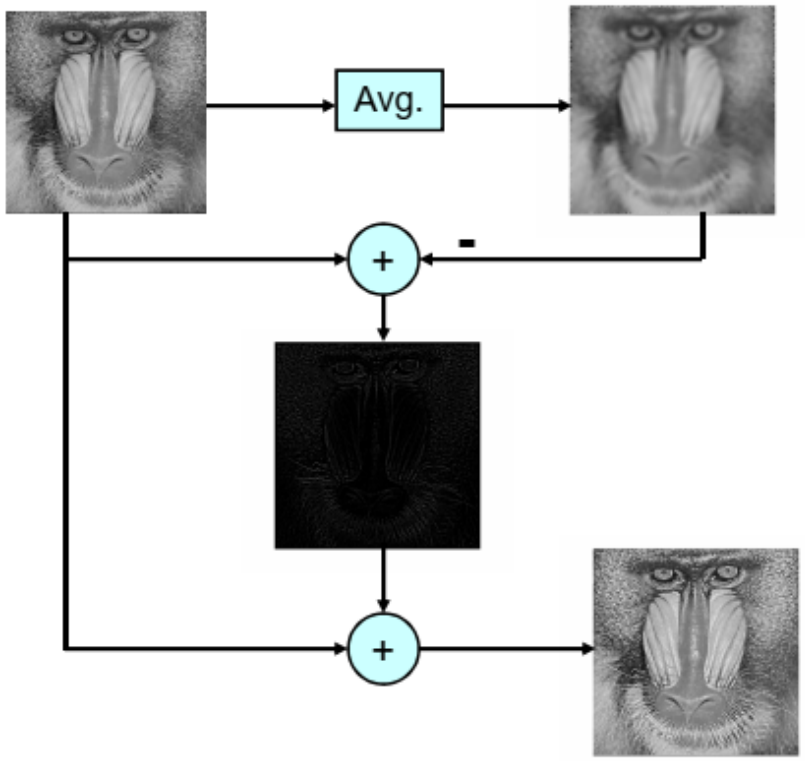
**FIGURE 3.39** 1-D illustration of the mechanics of unsharp masking. (a) Original signal. (b) Blurred signal with original shown dashed for reference. (c) Unsharp mask. (d) Sharpened signal, obtained by adding (c) to (a).

- Steps
  1. Blur the image
  2. Subtract the blurred version from the original (this is called the mask)
  3. Add the "mask" to the original

# High-boost filtering



## high-boost filtering




Let $f'(x)$ is the blurred image.

$$g_{mask}(x,y) = f(x,y) - f'(x,y)$$

$$g(x,y) = f(x,y) + A * g_{mask}(x,y)$$

Unsharp mask:

high-boost with A=1