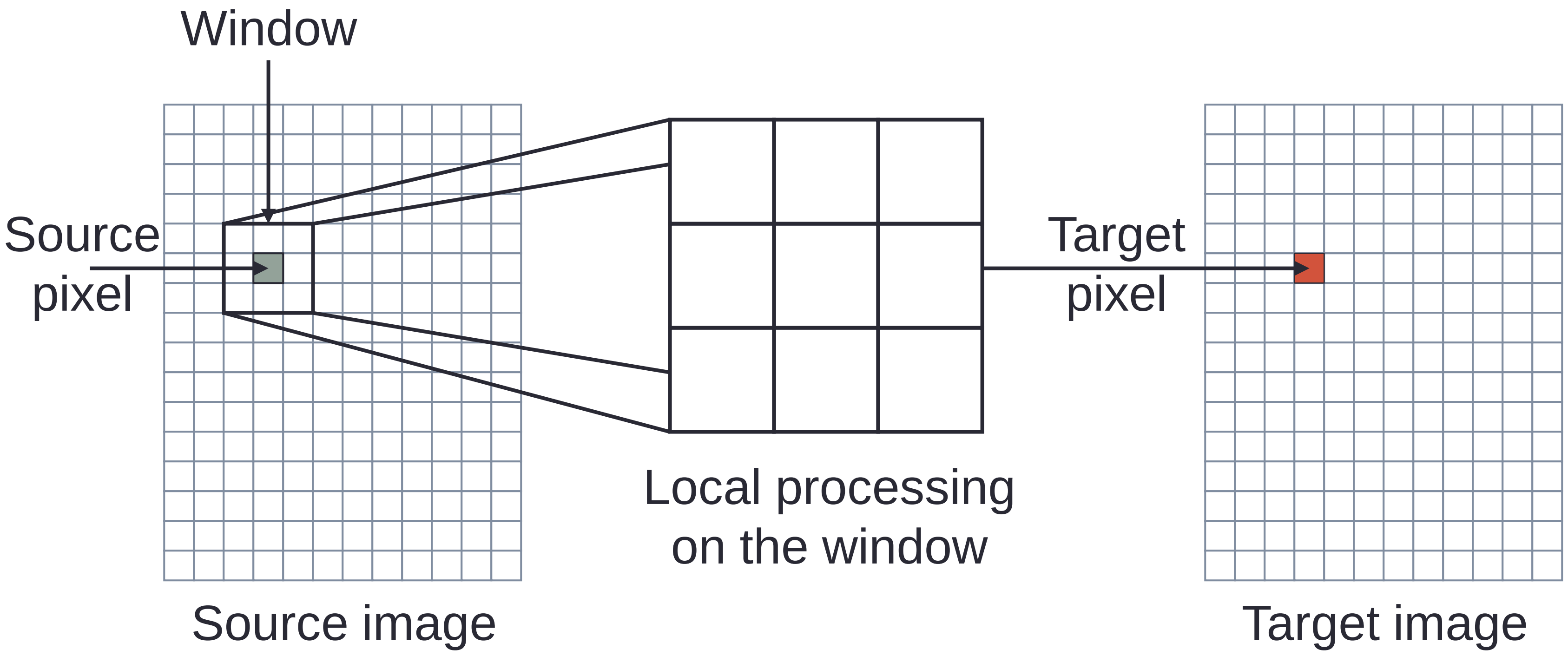
COMP2005

Linear Filters :

- Convolution, Mean Filtering and Noise
- Gaussian Filtering

Spatial Filtering



Why?

- Intensity Transforms read and affect only a single pixel, their power is limited
- Images are spatially organised data structures, many important attributes vary slowly across the image
  - Object identity
  - Viewed surface orientation, colour, etc
  - Illumination
- Processes restricted to a small, compact area have access to more information but are still likely to consider a single object, surface, illumination pattern, etc.

Image Noise

- Noise = small errors in image values
- Imperfect sensors introduce noise
- Image compression methods are lossy: repeated coding & decoding adds noise

Original

JPEG

Difference (Enhanced)

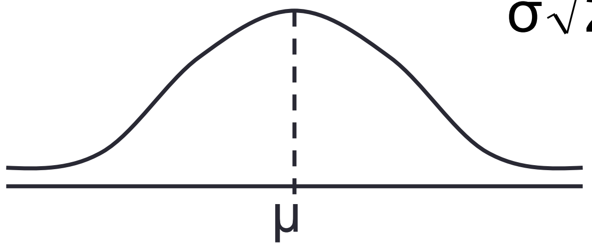
- Noise is often modelled as additive:

  Recorded value = true value + random noise value

Gaussian Noise

- Sensors often give a measurement a little off the true value
  - On average they give the right value
  - They tend to give values near the right value rather than far from it
- We model this with a Gaussian
  - Mean (μ) = 0
  - Variance (σ^2) indicates how much noise there is

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$



Gaussian Noise

The level of noise is related to the Gaussian parameter σ




$\sigma = 1$ $\sigma = 10$ $\sigma = 20$

Image with varying degrees of Gaussian noise added

Noise Reduction

- If you have multiple images, taking the mean value of each pixel will reduce noise
  - Noise is randomly added to each value
  - Mean value added is 0
  - If you average a large set of estimates of the same pixel the random noise values will cancel out

42 43 44 41 40 42 42 44 40

➔ 42

- Given only a single image, averaging over a local region has a similar effect

42 43 44

41 40 42 ➔ 42

42 44 40

- Ideally, we would choose the region to only include pixels that should have the same value
- **We need a spatial filter……**

Spatial Filtering: Convolution

- Many filters follow a similar pattern - multiplying each image value by a corresponding filter entry, and summing the results

$F_{(-1,-1)}$	$F_{(0,-1)}$	$F_{(+1,-1)}$
$F_{(-1,0)}$	$F_{(0,0)}$	$F_{(+1,0)}$
$F_{(-1,+1)}$	$F_{(0,+1)}$	$F_{(+1,+1)}$

Filter Window

$P_{(x-1,y-1)}$	$P_{(x,y-1)}$	$P_{(x+1,y-1)}$
$P_{(x-1,y)}$	$P_{(x,y)}$	$P_{(x+1,y)}$
$P_{(x-1,y+1)}$	$P_{(x,y+1)}$	$P_{(x+1,y+1)}$

Picture Window

$$
\begin{aligned}
& F_{(-1,-1)} \times P_{(x-1,y-1)} \\
& + F_{(0,-1)} \times P_{(x,y-1)} \\
& + F_{(+1,-1)} \times P_{(x+1,y-1)} \\
& + F_{(-1,0)} \times P_{(x-1,y)} \\
& + \ldots \\
& + F_{(+1,+1)} \times P_{(x+1,y+1)}
\end{aligned}
$$

Result

Filtering

- More generally, with a filter with radius *r*
  - $p_{x,y}$ is the original image value at (x,y)
  - $p'_{x,y}$ is the new image value at (x,y)

$$p'(x,y) = \sum_{dx=-r}^{+r} \sum_{dy=-r}^{+r} f_{dx,dy} \times p_{x+dx,y+dy}$$

- Many, though not all, filters work this way, e.g. the mean filter:

3 × 3 mean filter

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25

5 × 5 mean filter

The Mean Filter

Original

Gaussian

Key Points

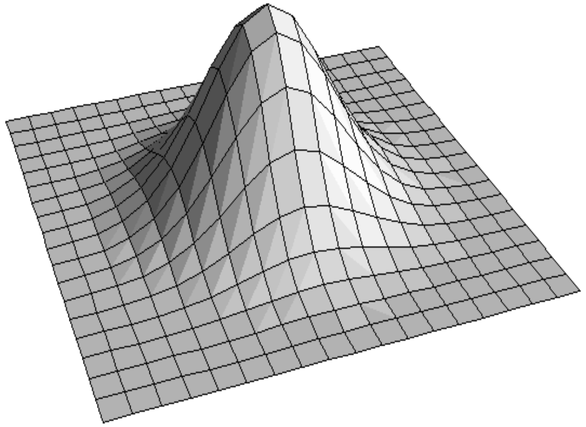
- Spatial filters operate on local image regions
- Many can be formulated as convolution with a suitable mask
- Noise reduction via mean filtering is a classic example

COMP2005

Gaussian Filtering

Gaussian Filters

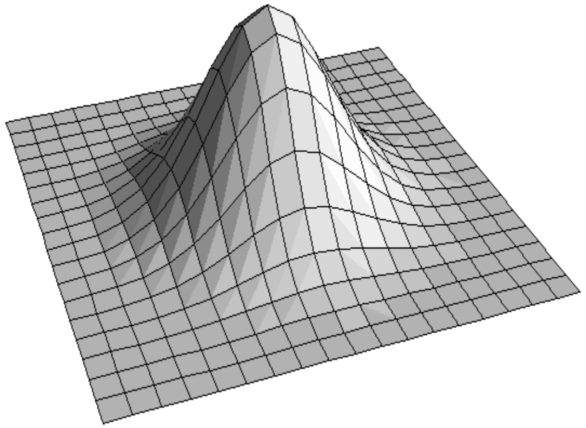
- Convolution with a mask whose weights are determined by a 2D Gaussian function
- Higher weight is given to pixels near the source pixel
- These are more likely to lie on the same object as the source pixel



$$P(x,y) = \frac{1}{\sigma^2 2\pi} e^{\frac{-(x^2+y^2)}{2\sigma^2}}$$

Discrete Gaussian Fitlers

- The Gaussian
  - Extends infinitely in all directions, but we want to process just a local window
  - Has a volume underneath it of 1, which we want to maintain



- We can approximate the Gaussian with a discrete filter
  - We restrict ourselves to a square window and sample the Gaussian function
  - We normalise the result so that the filter entries add to 1

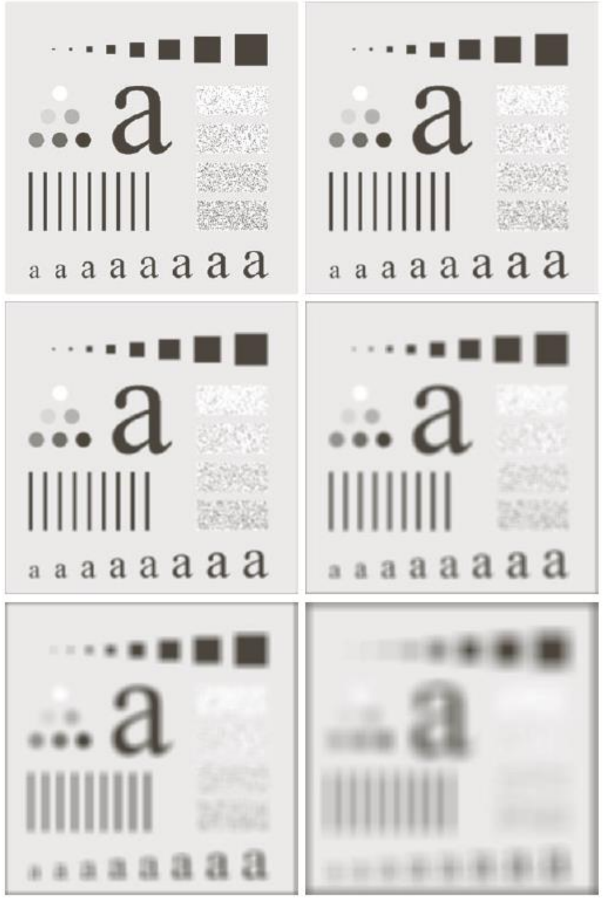
Example

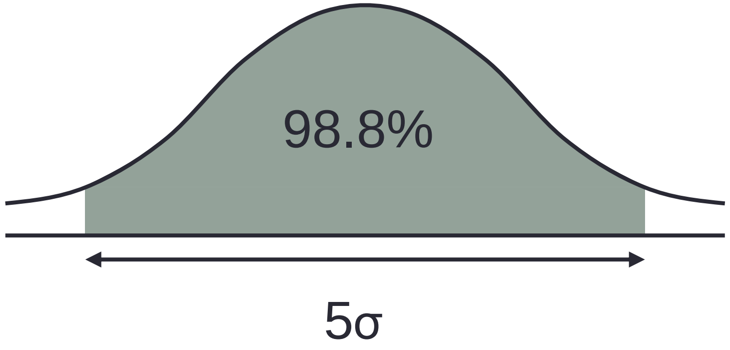
- Suppose we want to use a 5x5 window to apply a Gaussian filter with $\sigma^2 = 1$
  - The centre of the window has x = y = 0
  - We sample the Gaussian at each point
  - We then normalise it

$$
\begin{array}{c}
\begin{array}{c|c|c|c|c|c|}
\cline{2-6}
-2 & 0.00 & 0.01 & 0.02 & 0.01 & 0.00 \\ \cline{2-6}
-1 & 0.01 & 0.06 & 0.10 & 0.06 & 0.01 \\ \cline{2-6}
0 & 0.02 & 0.10 & 0.16 & 0.10 & 0.02 \\ \cline{2-6}
1 & 0.01 & 0.06 & 0.10 & 0.06 & 0.01 \\ \cline{2-6}
2 & 0.00 & 0.01 & 0.02 & 0.01 & 0.00 \\ \cline{2-6}
\multicolumn{1}{c}{} & \multicolumn{1}{c}{-2} & \multicolumn{1}{c}{-1} & \multicolumn{1}{c}{0} & \multicolumn{1}{c}{1} & \multicolumn{1}{c}{2}
\end{array}
\end{array}
\times \frac{1}{0.96}
$$

Gaussian Filters

- How big should the filter window be?
  - With Gaussian filters this depends on the variance (σ^2)
  - Under a Gaussian curve 98% of the area lies within 2σ of the mean
  - A filter width of 5σ gives more than 98% of the values we want





The Gaussian Filter

Original

Gaussian

Separable Filters

- The Gaussian filter is separable
  - A 2D Gaussian is equivalent to two 1D Gaussians
  - First you filter with a 'horizontal' Gaussian
  - Then with a 'vertical' Gaussian.....

0.06
0.24
0.40
0.24
0.06

0.06	0.24	0.40	0.24	0.06

$$
\begin{aligned}
P(x,y) &= \frac{1}{\sigma^2 2\pi} e^{\frac{-(x^2+y^2)}{2\sigma^2}} \\
&= \frac{1}{\sigma\sqrt{2\pi}} \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-x^2}{2\sigma^2}} e^{\frac{-y^2}{2\sigma^2}} \\
&= \left(\frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-x^2}{2\sigma^2}} \right) \left(\frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-y^2}{2\sigma^2}} \right) \\
&= P(x) \times P(y)
\end{aligned}
$$

Separable Filters

- The separated filter is more efficient
  - Given an N×N image and a n×n filter we need to do $O(N^2n^2)$ operations
  - Applying two n×1 filters to a N×N image takes $O(2N^2n)$ operations
- Example
  - A 600×400 image and a 5×5 filter
  - Applying it directly takes around 6,000,000 operations
  - Using a separable filter takes around 2,400,000 - less than half as many

Key Points

- Like mean filtering, and Gaussian smoothing can be used to remove additive noise
- Gaussian smoothing emphasises pixels near the source pixel, where it is more likely image properties are fixed
- Gaussian smoothing is separable, and so efficient

University of Nottingham
UK | CHINA | MALAYSIA

COMP-2032

Introduction to Image Processing

Lecture 3B

Non-Linear Filters

Learning Outcomes

IDENTIFY

1. Median Filtering
2. Anisotropic Diffusion
3. Bilateral Filtering

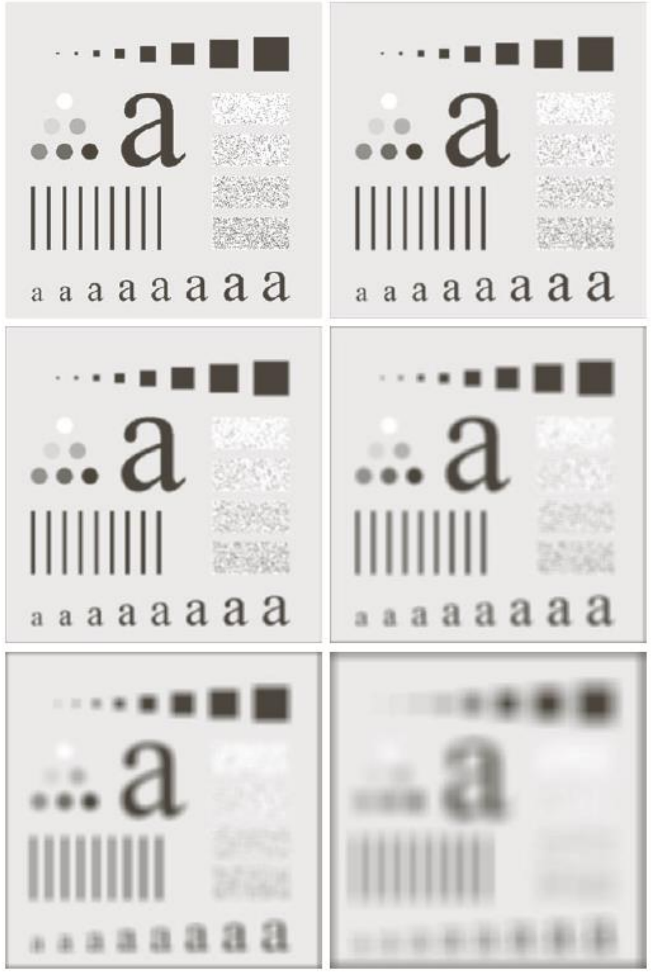
Non-Linear Filters

- Convolution with a mask of weights compute a linear function of a set of pixel values
- Many operations can be implemented this way, but not all:

- *Median filtering*
- *Anisotropic diffusion/Bilateral filtering*

Linear filters smooth sharp image changes, **nonlinear filters** tend to preserve or even enhance them

Difference



Gaussian Smoothing

ACK: Prof. Tony Pridmore, UNUK

Median Filtering

Salt and Pepper Noise

Sometimes sensors either fail to respond or saturate in error

- A false saturation gives a white spot in the image (salt)
- A failed response gives a black spot in the image (pepper)
- Sometimes called **speckle noise**

1%

10%

20%

An image with varying amounts of salt and pepper noise added

ACK: Prof. Tony Pridmore, UNUK

Reducing Salt and Pepper Noise

Original | Salt & Pepper | Original | Salt & Pepper

ACK: Prof. Tony Pridmore, UNUK

Mean Filtering | **Gaussian Filtering**

The Median Filter

Alternative: Median Filter

- Statistically the median is the middle value in a set
- Each pixel is set to the median value in a local window
- Result is a real pixel value, not a combination
- Noise pixels are outliers
- Noise would have to affect >1/2 the pixels to appear in the output

123	124	125
129	127	9
126	123	131

123	124	125	129	127	9	126	123	131

Find the values in a local window

9	123	123	124	125	126	127	129	131

Sort them

9	123	123	124	**125**	126	127	129	131

Pick the middle one

A mean filter would give 113

ACK: Prof. Tony Pridmore, UNUK

The Median Filter



Original

Salt & Pepper

Gaussian



ACK: Prof. Tony Pridmore, UNUK

Anisotropic Diffusion

Anisotropic Diffusion

- Median filtering is good given small regions of speckle noise, less good if edges are important
- There exist explicit edge-preserving smoothing ops

Diffusion

- Spreading out
- Mean and Gaussian filters can be seen as diffusion processes

Anisotropic

Not the same in all directions

Basic IDEA

- Mean and Gaussian filters make each pixel more like its neighbours
- Anisotropic diffusion makes each pixel more like those neighbours **that it is already similar to**

ACK: Prof. Tony Pridmore, UNUK

Anisotropic Diffusion

We have a similar function, *s(p,q)*

- *s(p,q)* has values in the range from 0 to 1
- If the pixels *p* and *q* are similar then *s(p,q)* is close to 1
- If the pixel *p* and *q* are different then *s(p,q)* is close to 0

$$p' = \frac{\sum q \times s(p,q)}{\sum s(p,q)}$$

We use *s(p,q)* to compute a weighted average of pixel values

- The new value at a pixel *p*, is based on all its neighbours, *q*

ACK: Prof. Tony Pridmore, UNUK

The Similarity Function

- The smoothing function, *s(p,q)* needs to be found
- If *d* is the difference between *p* and *q* and *D* is the maximum possible difference we can use:

$$\frac{D-d}{D}$$

$$s(p,q) = e^{\left(\frac{p-q}{K}\right)^2}$$

Other functions often used include:

$$s(p,q) = \frac{1}{1+\left(\frac{p-q}{K}\right)^2}$$

K determines the amount of smoothing

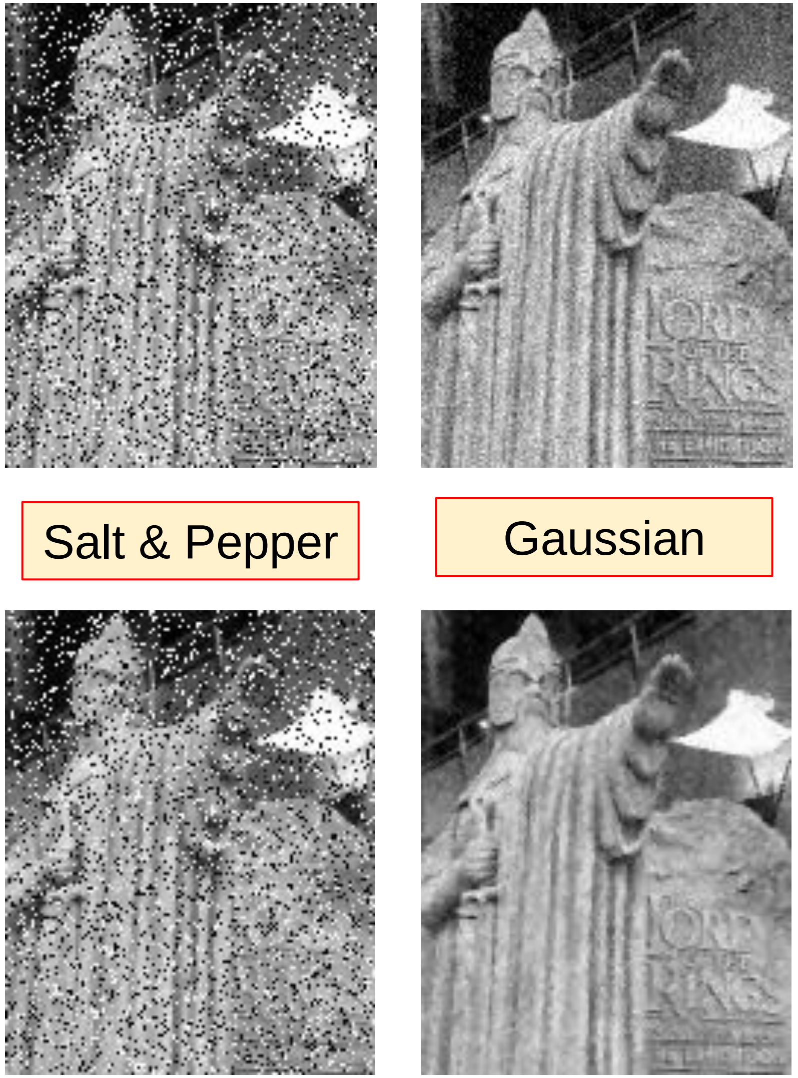
ACK: Prof. Tony Pridmore, UNUK

Anisotropic Diffusion

The examples here used the similarity function:

$$s(p,q) = \frac{1}{1 + \left(\frac{p-q}{K}\right)^2}$$

With K = 25



ACK: Prof. Tony Pridmore, UNUK

Anisotropic Diffusion

A higher value of **K** gives greater smoothing, but **edges are still (quite) sharp**

K = 5

K = 25

K = 50

K = 100

ACK: Prof. Tony Pridmore, UNUK

Anisotropic Diffusion

We can apply the filter repeatedly to give greater smoothing

Original

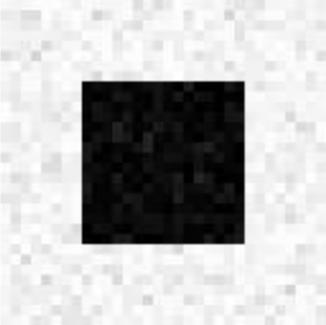
1 Iteration

2 Iteration

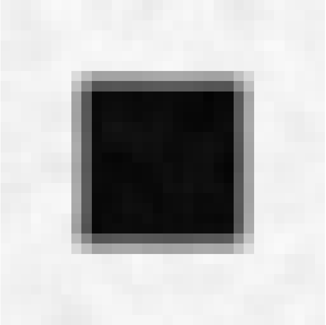
3 Iteration

ACK: Prof. Tony Pridmore, UNUK

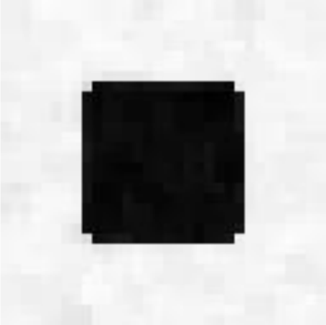
Reducing Noise near Edges



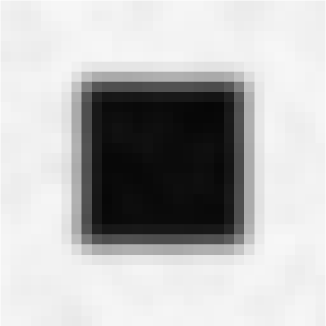
Noise Image



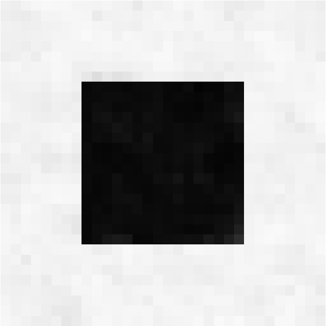
Mean



Median



Gaussian



Anisotropic Diffusion

Anisotropic diffusion is to mean filtering as ???? Is to Gaussian filtering…

ACK: Prof. Tony Pridmore, UNUK

University of Nottingham
UK | CHINA | MALAYSIA

Bilateral Filtering

Bilateral Filtering

- Anisotropic Diffusion is related to mean filtering
- If the similarity function is always 1 we get a mean filter

$$p' = \frac{\sum q \times s(p,q)}{\sum s(p,q)}$$

Sums pixel values in a region

Counts pixel values in a region

Bilateral filters modify Gaussian smoothing in a similar way

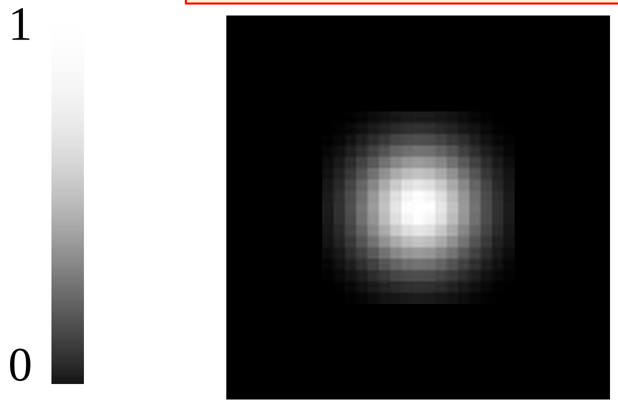
- One Gaussian weights pixels that are near the source
- Another Gaussian weights pixels that have similar intensity to the source pixel

ACK: Prof. Tony Pridmore, UNUK

Gaussian Smoothing Again

$$GB[I]_{\mathbf{p}} = \sum_{\mathbf{q}\in S} G_{\sigma}(\|\mathbf{p}-\mathbf{q}\|)\, I_{\mathbf{q}}$$

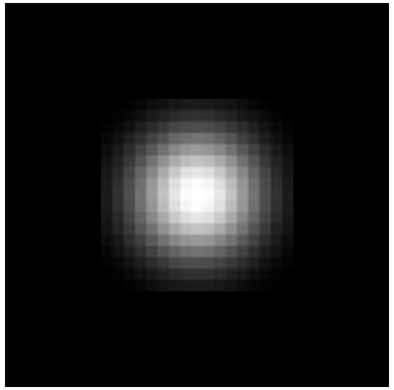
normalized Gaussian function

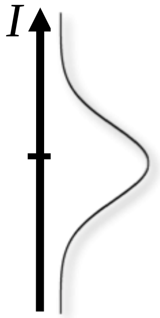


ACK: Prof. Tony Pridmore, UNUK

Bilateral Filtering

$$BF[I]_{\mathbf{p}} = \underbrace{\overset{\text{new}}{\frac{1}{W_{\mathbf{p}}}}}_{\text{normalization factor}} \sum_{\mathbf{q}\in S} \underbrace{\overset{\text{not new}}{G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)}}_{\textbf{\textit{space}}\text{ weight}} \underbrace{\overset{\text{new}}{G_{\sigma_r}(|I_{\mathbf{p}}-I_{\mathbf{q}}|)}}_{\textbf{\textit{range}}\text{ weight}} I_{\mathbf{q}}$$

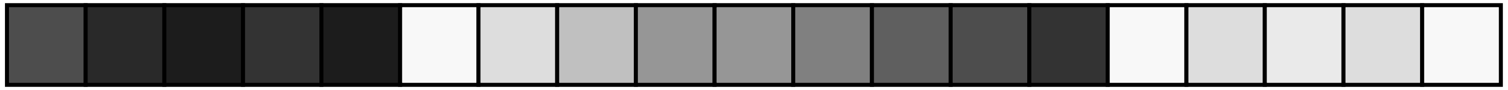




ACK: Prof. Tony Pridmore, UNUK

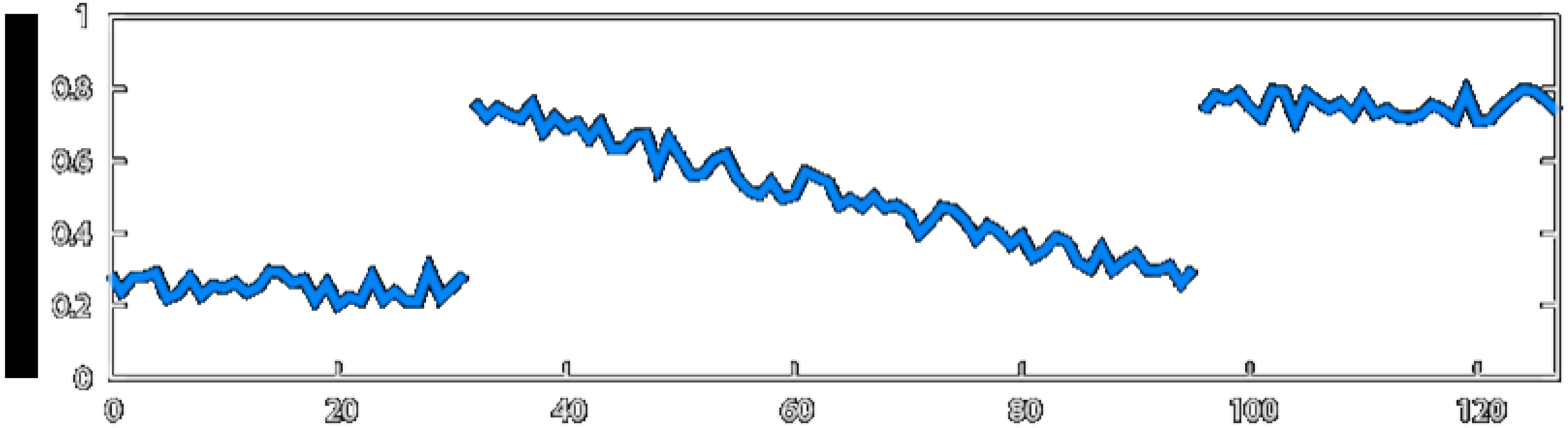
A One Dimensional Example

1D image = line of pixels



Better visualised as a plot

pixel intensity

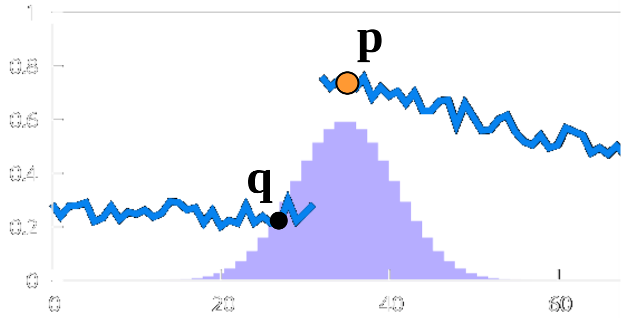


pixel position

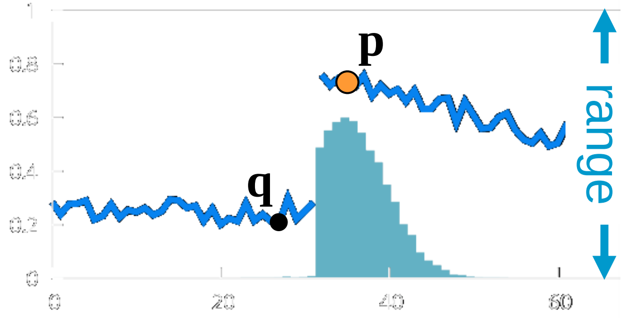
ACK: Prof. Tony Pridmore, UNUK

Gaussian & Bilateral Smoothing

Gaussian blur



Bilateral filter

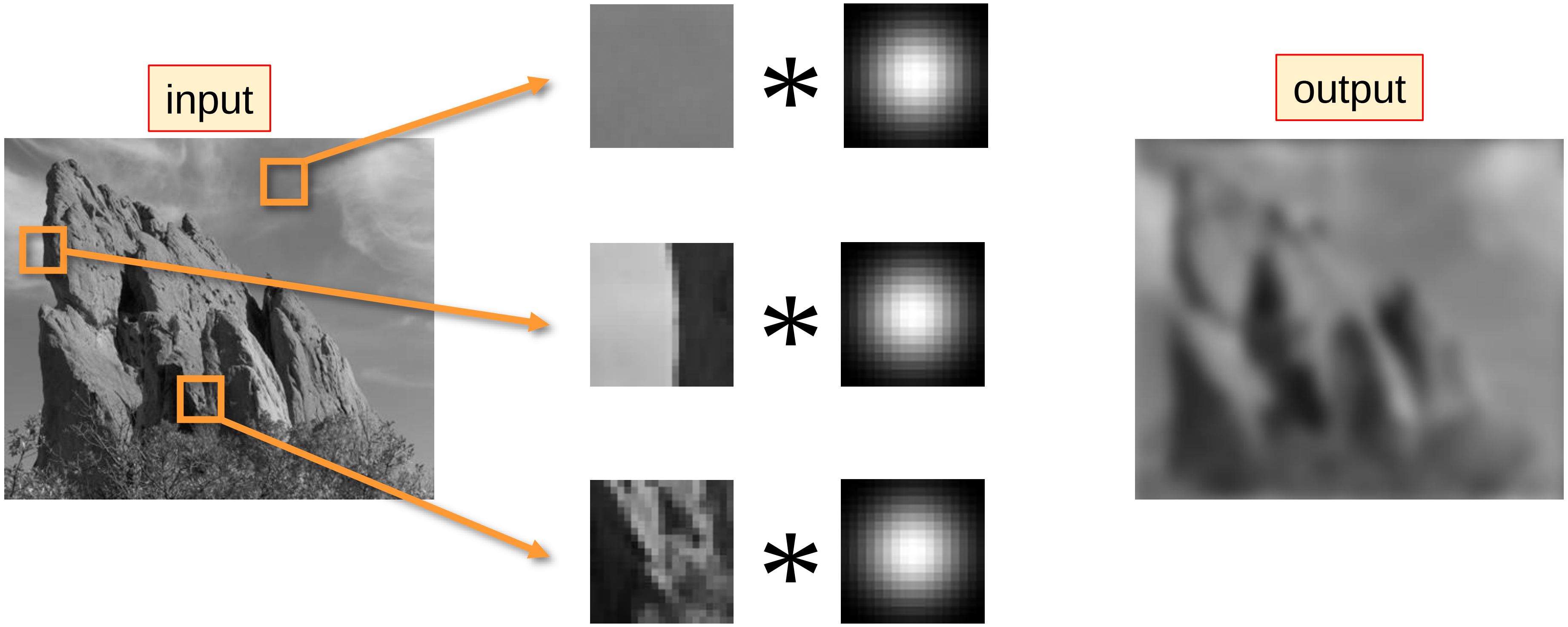


$$GB[I]_{\mathbf{p}} = \sum_{\mathbf{q}\in S} G_{\sigma}(\underbrace{\|\mathbf{p}-\mathbf{q}\|}_{\text{space}})\, I_{\mathbf{q}}$$

$$BF[I]_{\mathbf{p}} = \underbrace{\frac{1}{W_{\mathbf{p}}}}_{\text{normalization}} \sum_{\mathbf{q}\in S} \underbrace{G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)}_{\text{space}}\, \underbrace{G_{\sigma_r}(|I_{\mathbf{p}}-I_{\mathbf{q}}|)}_{\text{range}}\, I_{\mathbf{q}}$$

ACK: Prof. Tony Pridmore, UNUK

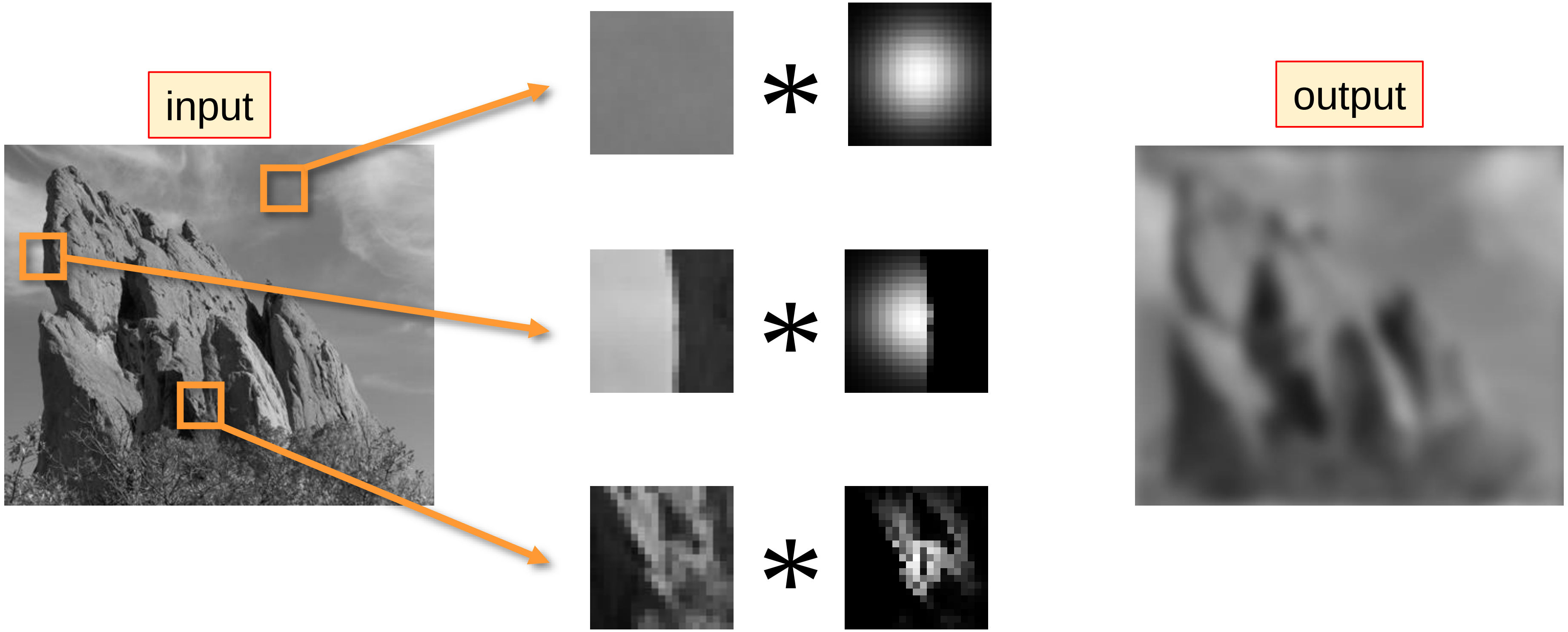
In 2D: Gaussian Smoothing



Same Gaussian kernel everywhere.

ACK: Prof. Tony Pridmore, UNUK

In 2D: Bilateral Filtering



The kernel shape depends on the image content.

ACK: Prof. Tony Pridmore, UNUK

Summary

1. Median Filtering
2. Anisotropic Diffusion
3. Bilateral Filtering

Questions

University of Nottingham
UK | CHINA | MALAYSIA

NEXT:

Thresholding
&
Binary Images