Image and video processing

Image filtering

Dr. Miles Hansard

Today's agenda

- Noise
- PSNR
- Convolution
- Image smoothing and noise reduction

Today's agenda

- Noise
- PSNR
- Convolution
- Image smoothing and noise reduction

Image & video processing flow diagram

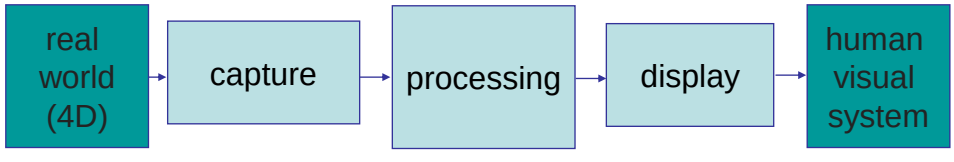


Image noise

original image

noisy image

Causes

- Incident Signal-to-Noise Ratio (SNR)
  - Power ratio
  - Source signal strength
  - Propagation losses
  - Background noise
  - Temperature noise at detector

Sensor induced noise

- Sensor induced noise
  - Interference among adjacent detectors
  - Thermal noise in infra-red detectors
    - Cooled to reduce
  - Leakage noise
    - Conductance at surface of detector
  - Shot noise
    - Electron devices

Also analog-to-digital conversion noise, and quantum noise.

Noise Removal – De-noising

- De-noising
  - Select or build a low-pass filter
  - Slide the filter over each successive sample point
  - Compute the convolution integral at that point
    (i.e., the area under the product curve)
    (the weighted average of the current pixel and its nearby neighbors)
  - Most filters are
    - symmetric around their origin and
    - fall off rapidly from their center

Note1: For 2D images/3D surfaces → the filter is a 2D image/3D surface

Note2: For digital images → we do not compute the integral,
but the sum of discrete values of the filter function

Convolution example (1D)

- Convolve these two sequences

Today's agenda

- Noise
- PSNR
- Convolution
- Image smoothing and noise reduction

Image quality measurement: SNR

- Signal-to-Noise Ratio (SNR)
  - estimates of the quality of a processed image *I'(x, y)* compared with an original image *I(x, y)*
  - Basic idea → to compute a single number that reflects the quality of a distorted image
  - The higher the value of the metric → the better the quality of the distorted images

NB In general SNR measures do not reflect human subjective perception

PSNR

- Peak signal-to-noise (distortion) ratio PSNR
  - For a given source image *I(x, y)* of dimension N x M and a distorted image *I'(x, y)*, the error or distortion is computed as follows:
    - Mean squared error (MSE) of the reconstructed image

$$MSE = \frac{\sum [I(x, y) - I'(x, y)]^2}{NM}$$

    - PSNR in decibels (dB)

$$PSNR = 20 \log_{10} \left(255 / \sqrt{MSE} \right)$$

PSNR

- PSNR values
  - typically range between 20 and 40
  - are usually estimated with two decimal points (e.g., 25.47)
  - the actual value is not meaningful, but the comparison between two values for different reconstructed images gives one measure of quality
- Importance of a PSNR difference
  - The MPEG committee used an informal threshold of 0.5 dB PSNR to decide whether to incorporate a coding optimization because they believed that an improvement of that magnitude would be visible

Distortion errors

- Visualisation of errors
  - a technique to visualize errors is to construct an error image which shows the pixel-by-pixel errors
  - to create the pixel-by-pixel image difference between the distorted and original images
    - problem: zero difference is black → most errors are small numbers → shades of black → hard to see!
    - solution: multiply the difference image by a constant → increase the visible difference

$$E(x, y) = 255\left[\left|I(x, y) - I'(x, y)\right| - \min\right] / (\max - \min)$$

Today's agenda

- Noise
- PSNR
- Convolution
- Image smoothing and noise reduction

Convolution: applications

- Deconvolution
  - Remove effects of previously applied linear operations
- Noise removal
  - Filtering to separate noise from signal for estimating noise
  - Feature detection
  - Periodic noise removal
- Feature enhancement
  - Using high pass filter, for instance

1D continuous convolution

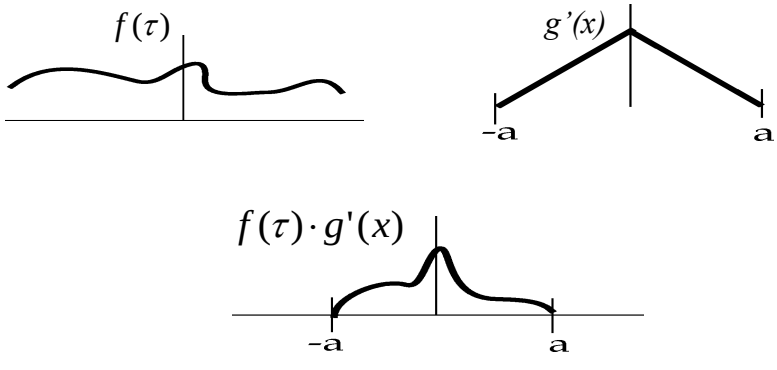
• If a signal r(t) is input into a linear system, the output is the convolution of the input signal with the transfer function h(t)

$$c(t) = \int_{-\infty}^{+\infty} h(t-\tau)\, r(\tau)\, d\tau$$

- A useful 'function' is the **Dirac delta**, which is like a 'spike', defined as:
- $\delta(x) = \infty$ if $x = 0$
- $\delta(x) = 0$ if $x \neq 0$
- $\int_{-\infty}^{\infty} \delta(x)\, dx = 1$
- Convolution of a function with $\delta(x)$ returns the original function. Convolution with $\delta(x - t)$ performs a *shift*.

Example: 1D continuous

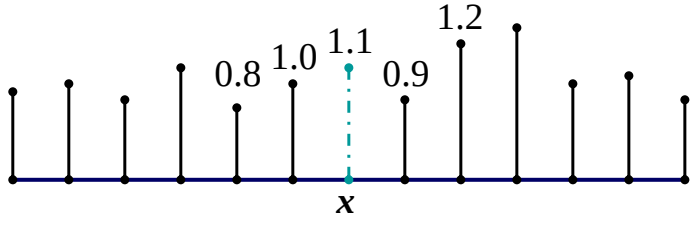
• Filtering by convolution
  – computes the area under the product curve $h(x) = f(\tau) \cdot g'(x)$



Discrete 1-D convolution

- Input
  - The discrete signal r with m sampling points
- Convolution kernel
  - Filter h of length n
- Output
  - c is a sequence of length m
  - the i^{th} element is $c(i) = r(i) * h(i) = \sum_j r(j)h(i-j)$

Example: 1D discrete

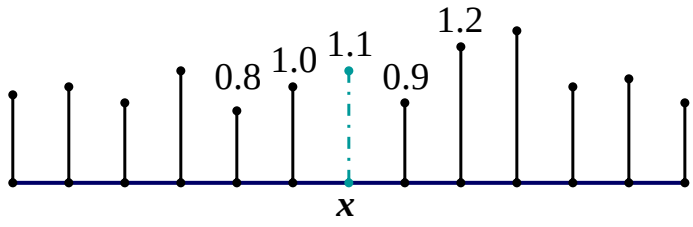


Filter
c=(1/9, 2/9, 1/3, 2/9, 1/9)
c(0)=1/3, c(1)=c(-1)=2/9, c(2)=c(-2)=1/9

Amplitude or intensity value at x=1.1

Convolution (filter response at x)

Example: 1D discrete



Filter

c=(1/9, 2/9, 1/3, 2/9, 1/9)

c(0)=1/3, c(1)=c(-1)=2/9, c(2)=c(-2)=1/9

Amplitude or intensity value at x=1.1

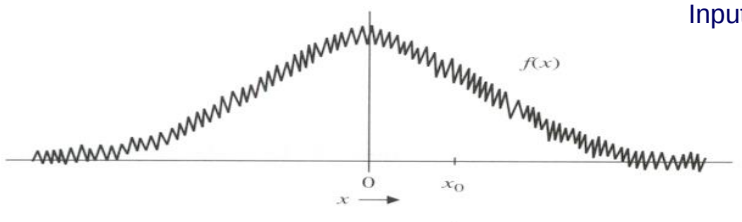
Convolution (filter response at x)

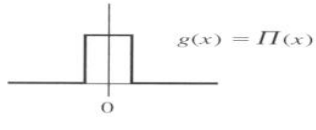
$$\sum_{i=x-2}^{x+2} I(i).c(i-x)$$ =(0.8)x(1/9)+(1.0)x(2/9)+(1.1)x(1/3)+(0.9)x(2/9)+ +(1.2)x(1/9)=1.011

Queen Mary University of London

Example: 1D

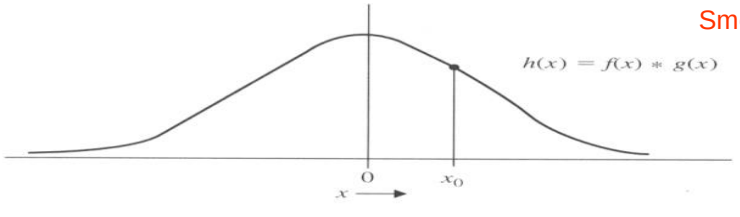
Input noisy signal



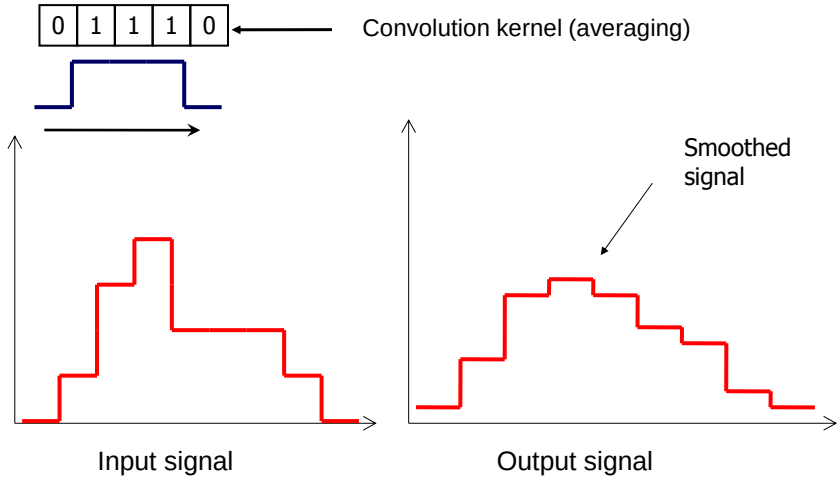


Rectangular kernel
(average filter)

Smoothed output



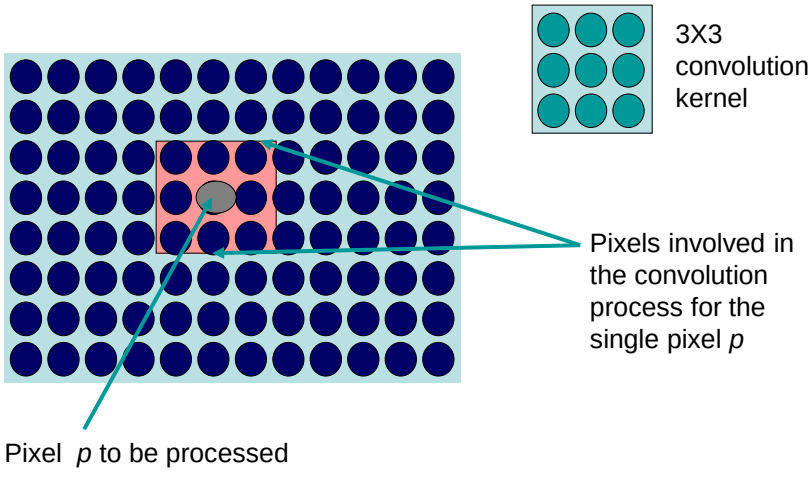
Example: 1D discrete



Convolution operator: properties

- Linearity: $f * (\alpha h + \beta g) = \alpha f*h + \beta f*g$
- Associativity: $(f * g) * h = f * (g * h)$
- Derivative: $d/dt (f * g) = f' * g = f * g'$

Discrete 2D convolution

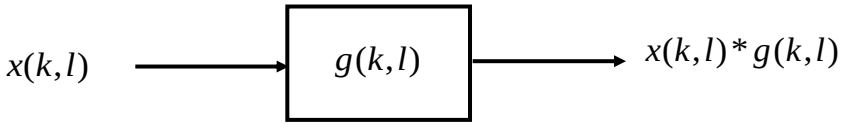


2D convolution

- 2D convolution

$$x(k,l) * g(k,l) = \sum_{k'=-\infty}^{+\infty} \sum_{l'=-\infty}^{+\infty} x(k',l')g(k-k',l-l')$$

- Notation



2D convolution: properties

- Commutativity

$$x(k,l) * g(k,l) = \sum_{k'=-\infty}^{+\infty} \sum_{l'=-\infty}^{+\infty} x(k',l') g(k-k', l-l')$$

$$= \sum_{k'=-\infty}^{+\infty} \sum_{l'=-\infty}^{+\infty} g(k',l') x(k-k', l-l') = g(k,l) * x(k,l)$$

2D convolution: properties

- Associativity

$$[x(k,l) * g(k,l)] * h(k,l) = x(k,l) * [g(k,l) * h(k,l)]$$

- Linearity

$$x(k,l) * [g(k,l) + h(k,l)] = x(k,l) * g(k,l) + x(k,l) * h(k,l)$$

Choices for convolution

- Determine the most appropriate shape of the window
- Determine the size of the window
- Solve the border problem

Size of the window

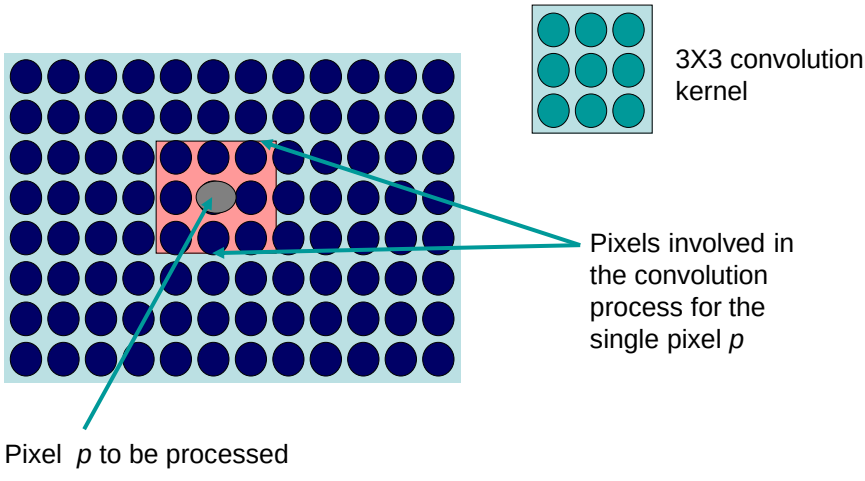
- The limits of the sums are defined by the size of the window (i.e., size of the filter) → $M_g \times N_g$

$$x(k,l) * g(k,l) = \sum_{k'=0}^{M_g-1} \sum_{l'=0}^{N_g-1} g(k',l')x(k-k',l-l')$$

Convolution procedure

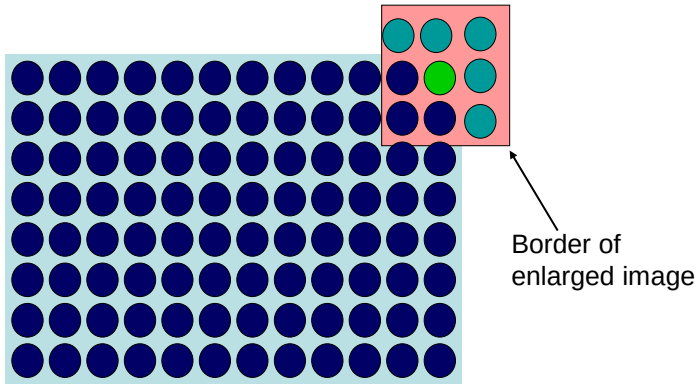
- Positioning
  - Place the centre of the window over the pixel position to be filtered
  - $M_g \times N_g$ pixels will be covered by the filter's window (mask, kernel)
- Multiply
  - the original pixel values of the image by the filter values corresponding to their location
- Add
  - Together the obtained multiplication results
  - The result of this addition is the convolution result

Discrete 2D convolution



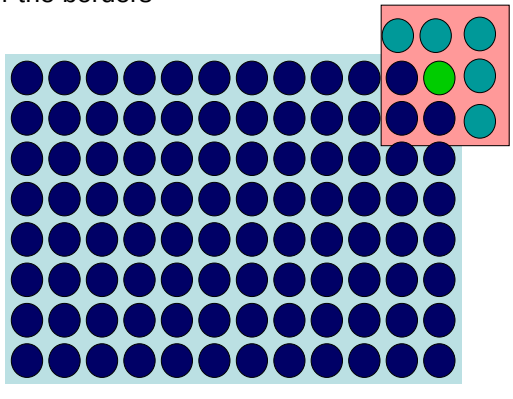
Border problem

- The values of the pixels outside the image but involved in the convolution process need to be estimated



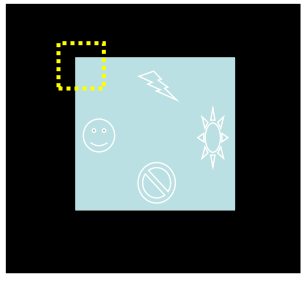
Border problem: solutions

- Change filter size along the border
- Enlarge image
  - Fill with zeros
  - Periodic extension of the image
  - Mirror the borders



Filling with zeros

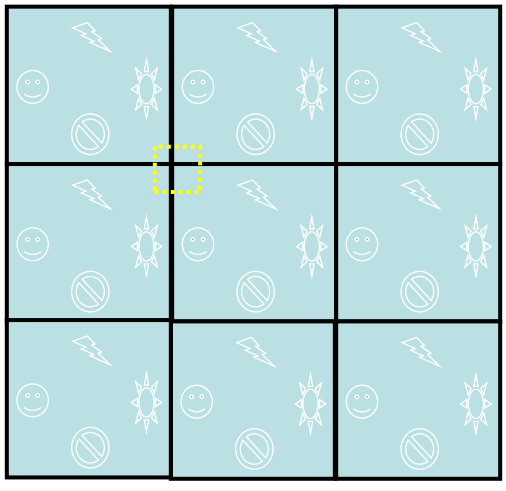
- Simple
- Generate border effects



Periodic extension

- Facilitate the software implementation of the filtering
- Coherent with the hypothesis of the Fourier Transform
- Better results than those obtained with filling with zeros

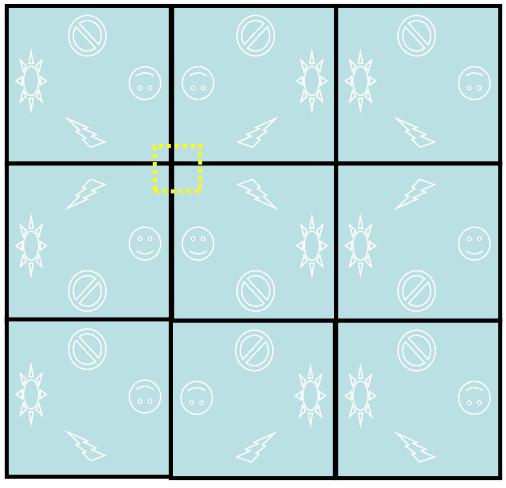
Periodic extension



Mirroring

- More complex than the previous 2 solutions
- Better to eliminate the border effects

Mirroring



2D spatial filtering: summary

- Output of the filter
  - value calculated from the convolution with a local neighborhood in the input image
- Local neighborhood
  - enclosed in a window (mask, kernel) of $M_g \times N_g$ pixels
- Filtering
  - performed by shifting the window over the whole image

Today's agenda

- Noise
- PSNR
- Convolution
- Image smoothing and noise reduction

Spatial domain filtering

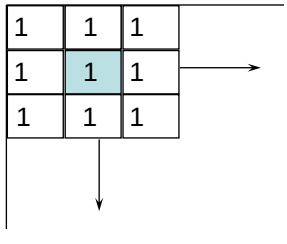
- Linear filters
  - Smoothing filters
    - mean filters
    - Gaussian filters
  - Edge enhancing filters
    - Sobel operator
    - Prewitt operator
    - Laplace operator
    - ….
- Non-linear filters
  - Median, Min, Max

Low-pass filtering

- Low-pass filtering
  - filter-out high frequencies
  - simplest linear filter: average= 1/n
  - example
    - Convolving the discrete signal in the previous example with the average filter. What is the response at x?
  - most popular filter: Gaussian

Smoothing filters

- Linear filters
- Adaptive filters (steerable)
- Non-linear filters
- Averaging → the simplest linear filter



Mean filter

1/9 x

1	1	1
1	1	1
1	1	1

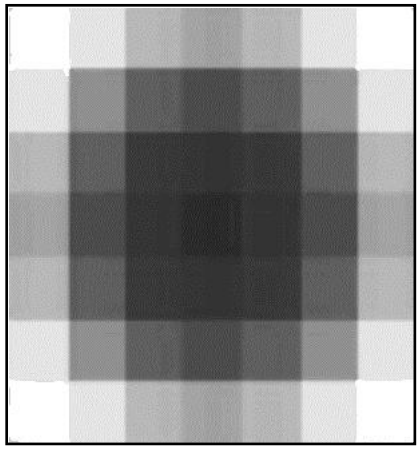
- Need for normalization
  - To conserve the total “energy” of the image (sum of all grey levels)
- Quick
- Severe edge blurring

Mean filter: results

original image 5x5 filter 9x9 filter

Gaussian filters

- 2D Gaussian kernel
  - (in the figure): the darker a pixel, the higher the filter value
  - the weighting values decrease proportionally to the distance from the center (exponentially)



Gaussian filter

- Gaussian filter
  - separable

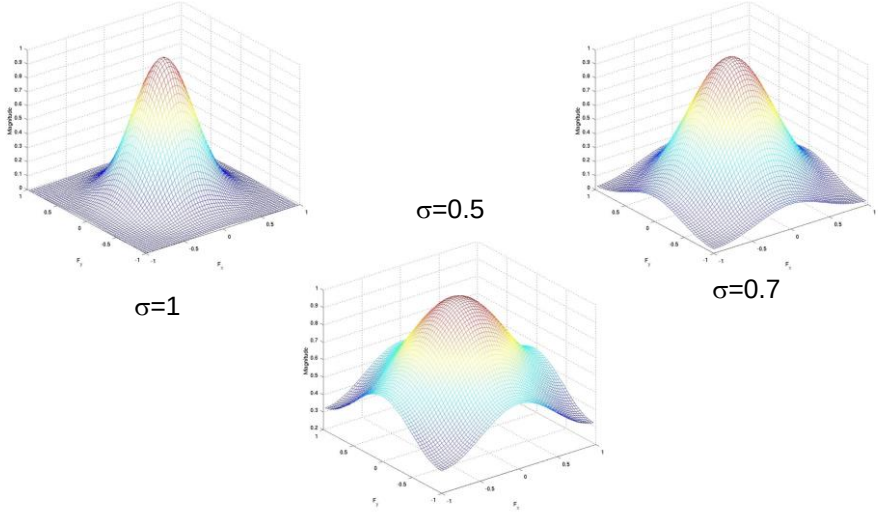
$$G_\sigma(k,l) = \frac{1}{2\sigma^2\pi} e^{-\frac{k^2+l^2}{2\sigma^2}}$$

- The Fourier transform of a Gaussian is another Gaussian, of reciprocal width.
- The convolution of two Gaussians is another Gaussian.

$$G_\sigma(k,l) = G_\sigma(k)G_\sigma(l) \qquad G_\sigma(k) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{k^2}{2\sigma^2}}$$

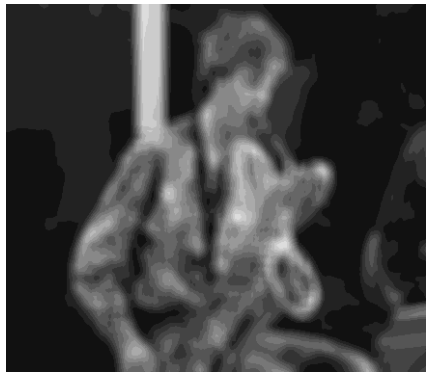
- We can perform the 2D convolution by:
  - First filtering the rows of the input image.
  - Then filtering the columns of the result.
  - This is much faster than using the 2D kernel.

Gaussian filter 7x7



Gaussian smoothing: results

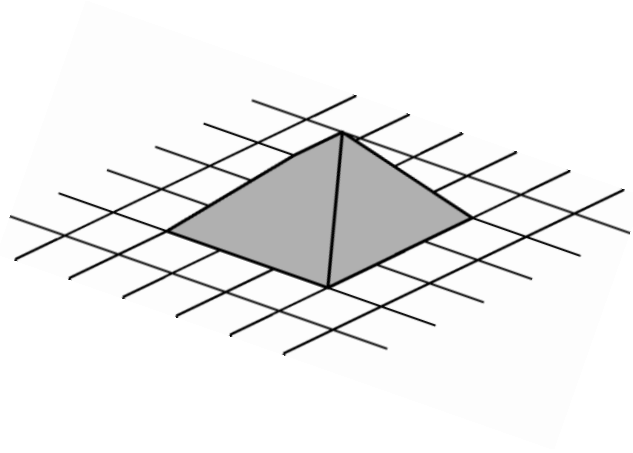
original image



13x13 filter kernel

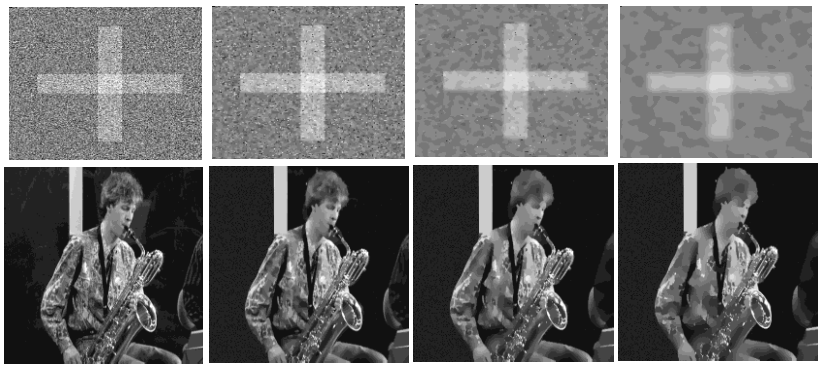
Linear pyramidal filter

- Simpler than the Gaussian filter
  - The kernel elements decrease linearly (not exponentially as in the Gaussian)

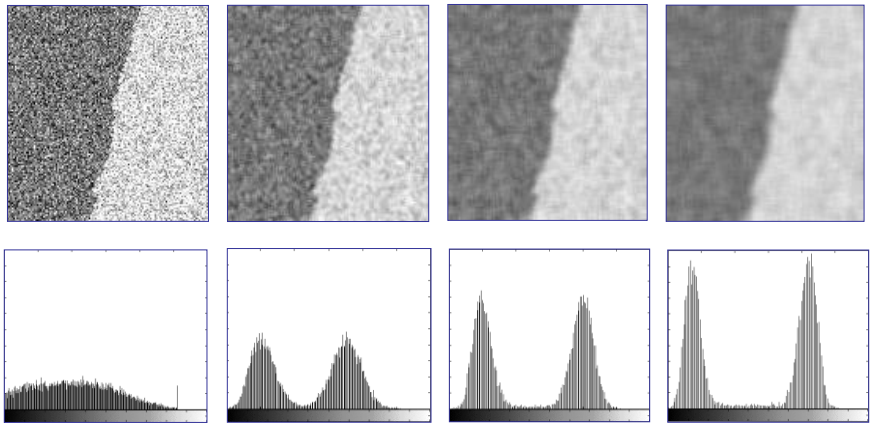


Non-linear de-noising and smoothing

- Advanced filters
  - remove high frequency components (noise) while keeping edges sharp



Noise in histogram thresholding



What did we learn today?

- Noise
- PSNR
- Convolution
- Image smoothing and noise reduction