ETH zürich

PAUL SCHERRER INSTITUT
PSI

Image filters

Anders Kaestner :: Laboratory for Neutron Scattering and Imaging

Outline :: Filters

Defining filters

Filters

Definition

A filter modifies the information in the image given a neighborhood.
Purpose: Enhance or suppress information.
Mainly two classes of filters will be covered here:

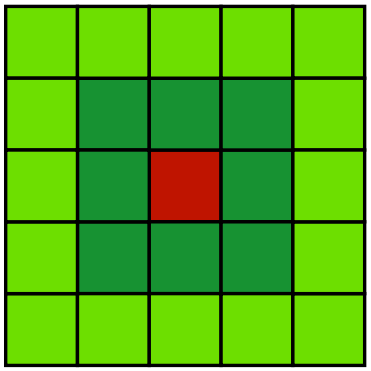
- Linear spatially invariant filters. Computed with convolution
- Non-linear filters

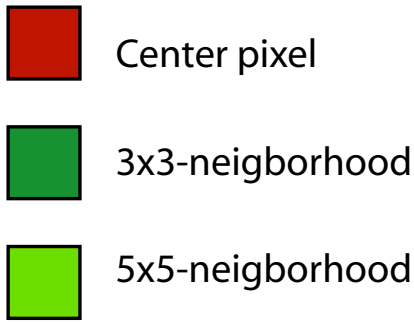
Books that cover filters are e.g. Jähne (2002) or Jain (1989)

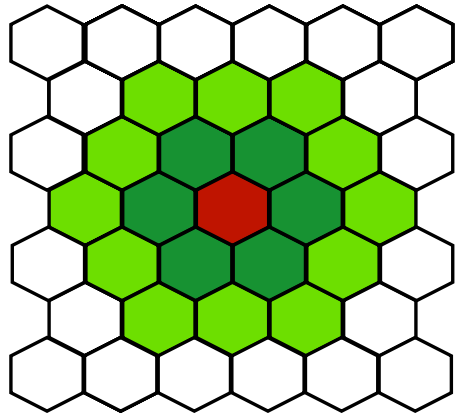
Neighborhoods

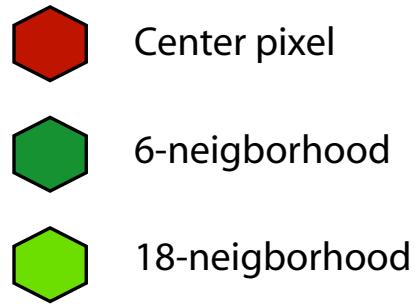
Definition

A neighborhood of a pixel/voxel are the adjacent pixels. Neighborhoods are defined differently on different grids









Linear spatially invariant filters

Definitions

Continuous

$$g(\mathbf{x}) = h * f(\mathbf{x}) = \int_{\Omega_h} f(\mathbf{x} - \boldsymbol{\tau})\, h(\boldsymbol{\tau})\, d\boldsymbol{\tau} \tag{1}$$

Discrete

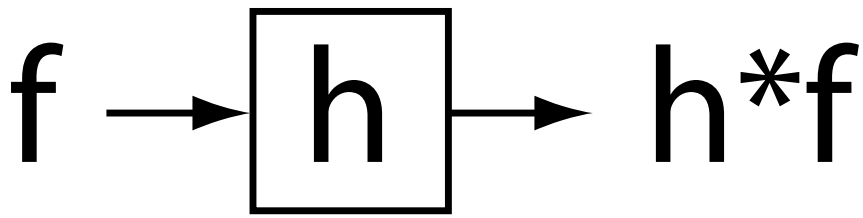
$$g[\mathbf{x}] = h * f[\mathbf{x}] = \sum_{p \in \Omega_h} f[\mathbf{x} - \mathbf{p}]\, h[\mathbf{p}] \tag{2}$$

where

$f(\mathbf{x})$ is the image to filter

h is the convolution kernel of the filter

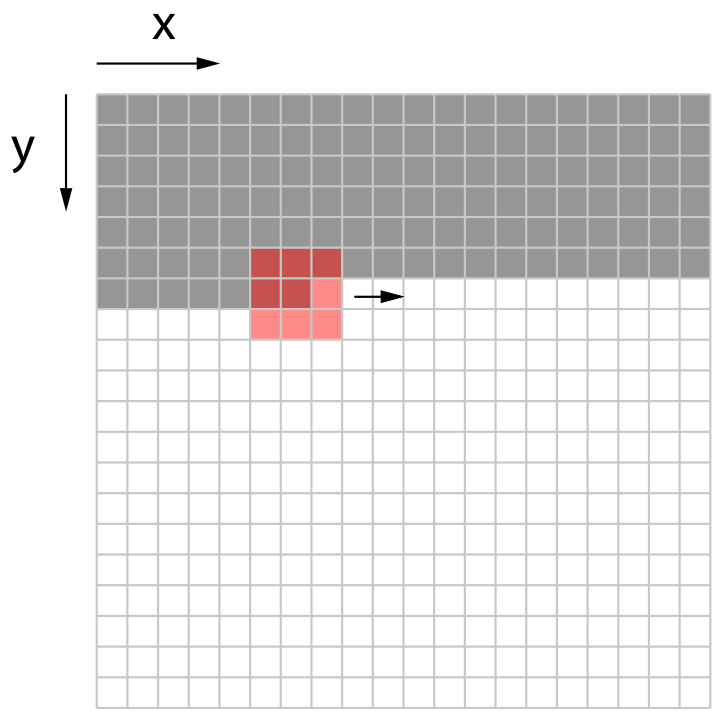
Block diagram



Convolution :: The principle

Discrete convolution is computed as :

$$g[\mathbf{x}] = f[\mathbf{x}] * h = \sum_{\boldsymbol{\tau} \in \Omega} f[\mathbf{x} - \boldsymbol{\tau}]\, h[\boldsymbol{\tau}] \tag{3}$$

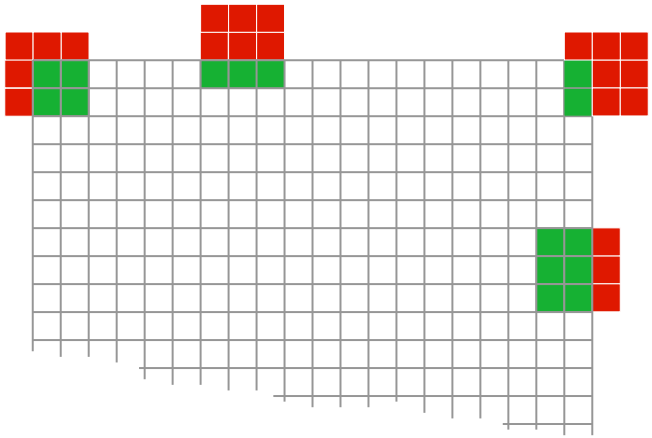


Edge processing

The convolution is defined for infinitely large images

Problem

Discrete images are finite. The convolution kernel has no image support on the edges



Edge processing strategies

- Ignore the edges
- Use only the supported part of the kernel
- Wrap around
- Mirror

Properties of the convolution

Commutative $(a * b) * c = a * (b * c)$

This property is useful if the kernel is separable

Associative $a * (b + c) = a * b + a * c$

L.H.S. is computationally less expensive than R.H.S.

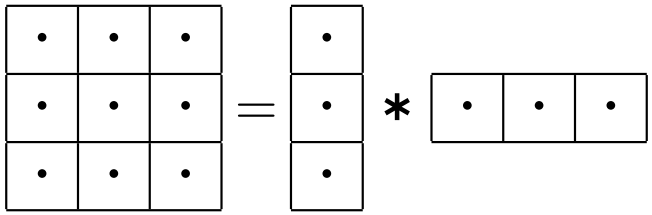
Spatially invariant

- Filter operation only depend on the intensities in $f(p) \quad \forall p \in \Omega_h(x)$
- Independent of position

Euclidean Separability

Definition

A convolution kernel is called separable if it can be split in two or more parts:



and Euclidean separable if each part only depends on one coordinate variable.

Example

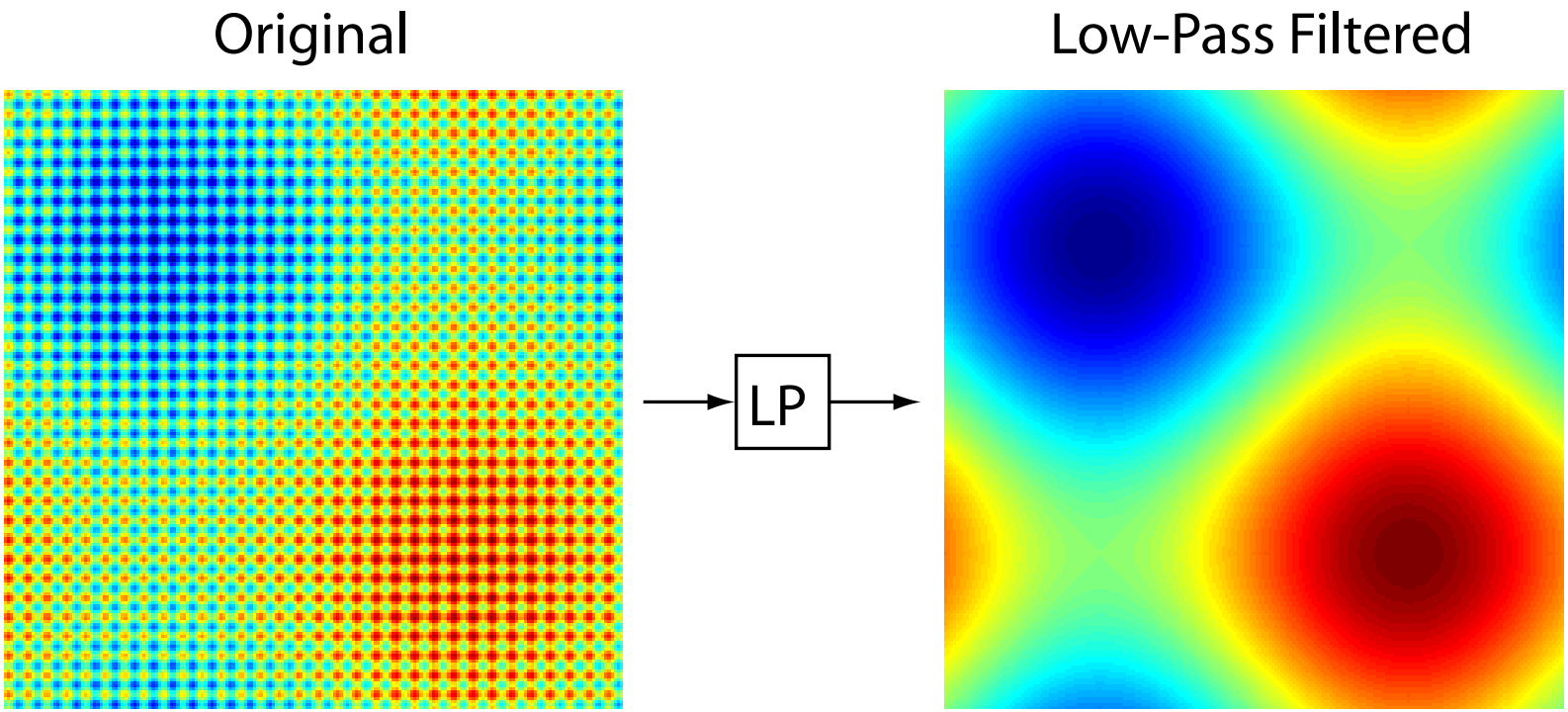
$$e^{-\frac{x^2+y^2}{2\sigma^2}} = e^{-\frac{x^2}{2\sigma^2}} * e^{-\frac{y^2}{2\sigma^2}}$$

Low-pass filters and denoising

Low-Pass Filters

The effects of a *Low-Pass* filter are in general

- slow changes are enhanced
- rapid changes are suppressed

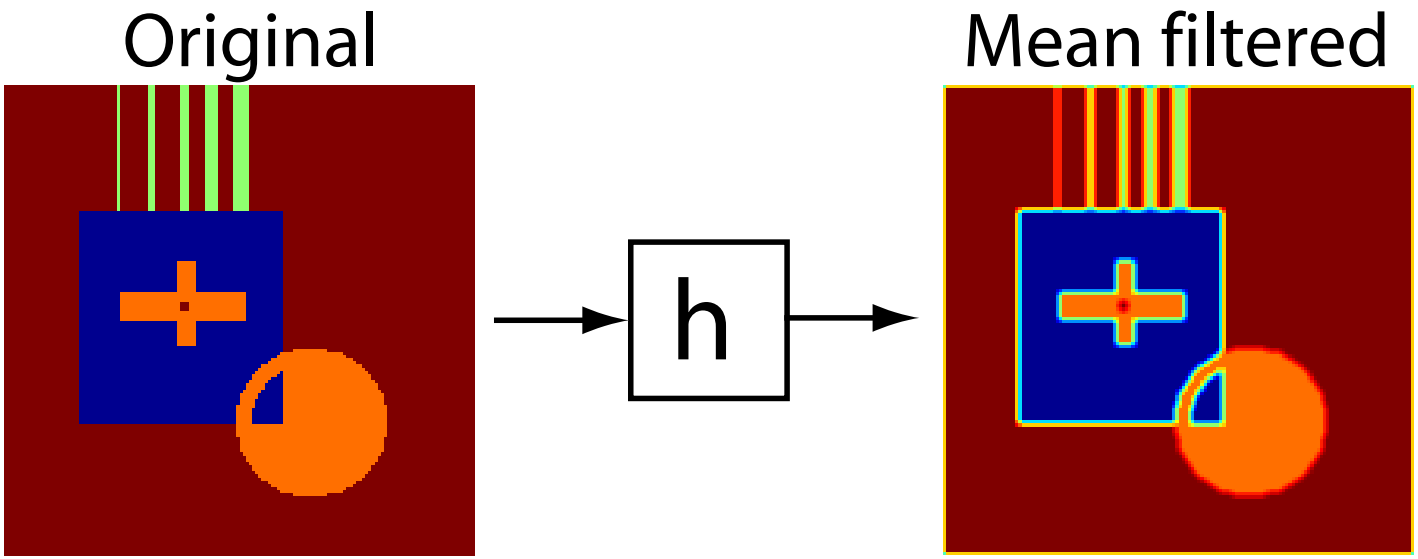


Low pass filters :: box filter

Computes the local average value of the pixel neigborhood
A 3 × 3 box filter is defined as

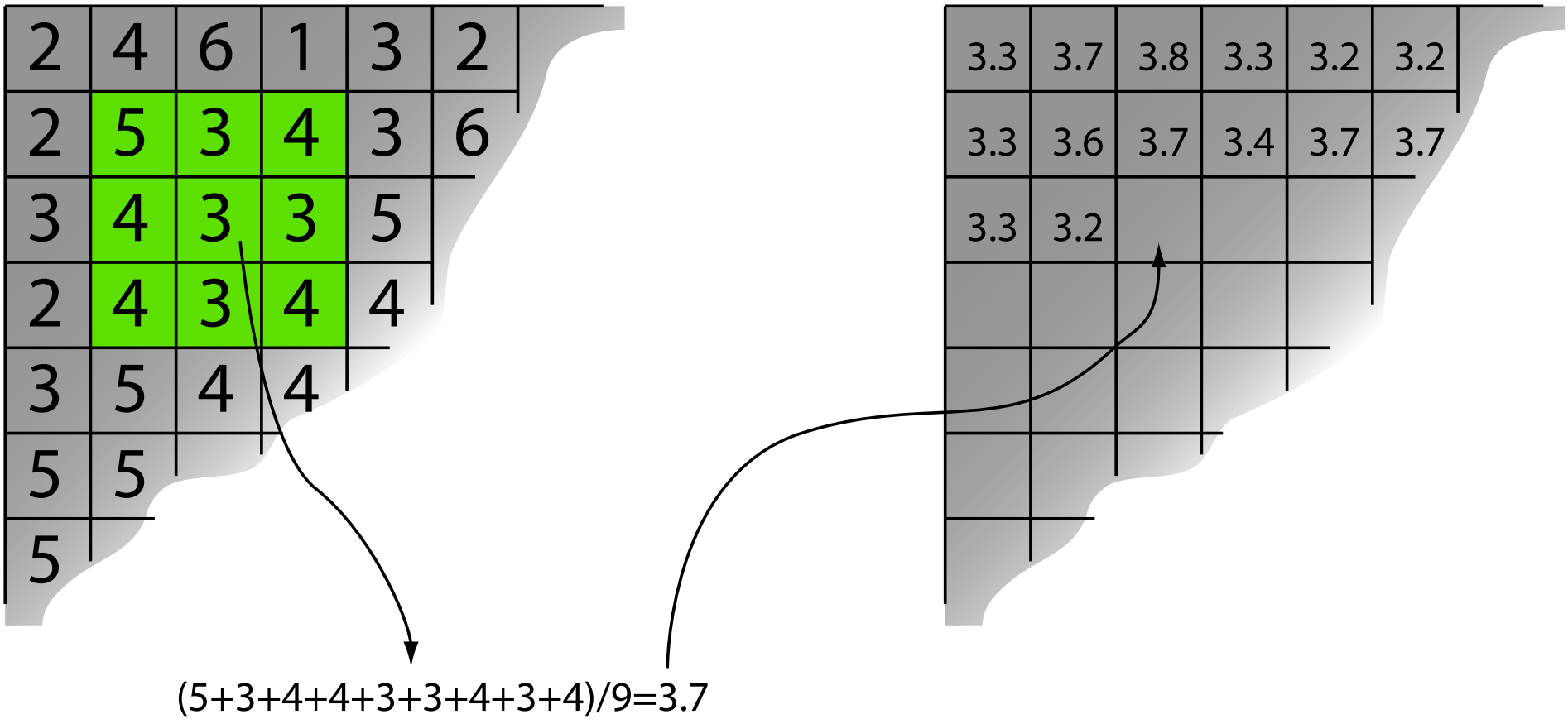
$$h = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = h_x * h_y = \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \end{bmatrix} * \frac{1}{3}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \tag{4}$$

The mean filter is also called a *mean filter*



Observe: The edges are less sharp after the low-pass filter.

Detailed example box filter



Low pass filters :: Binomial Filter

The two kernels $B_x = \frac{1}{2}[\ 1 \quad 1]$ and $B_y = \frac{1}{2}\begin{bmatrix} 1 \\ 1 \end{bmatrix}$

1D filter coefficients are taken from Pascal's triangle

$$\begin{array}{ccccccccc} & & & & 1 & & & & \\ & & & 1 & & 1 & & & \\ & & 1 & & 2 & & 1 & & \\ & 1 & & 3 & & 3 & & 1 & \\ 1 & & 4 & & 6 & & 4 & & 1 \end{array}$$

A 3×3-kernel is defined as

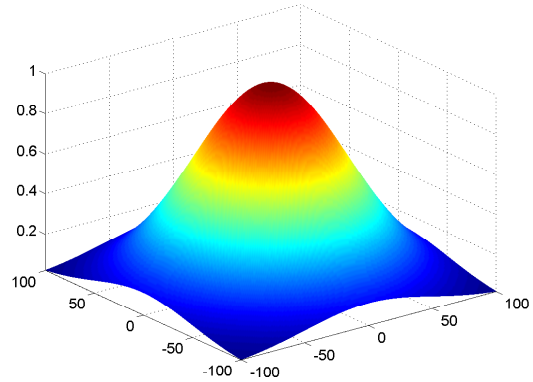
$$B = B_x * B_x * B_y * B_y = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{5}$$

Low-pass filters :: Gaussian filter

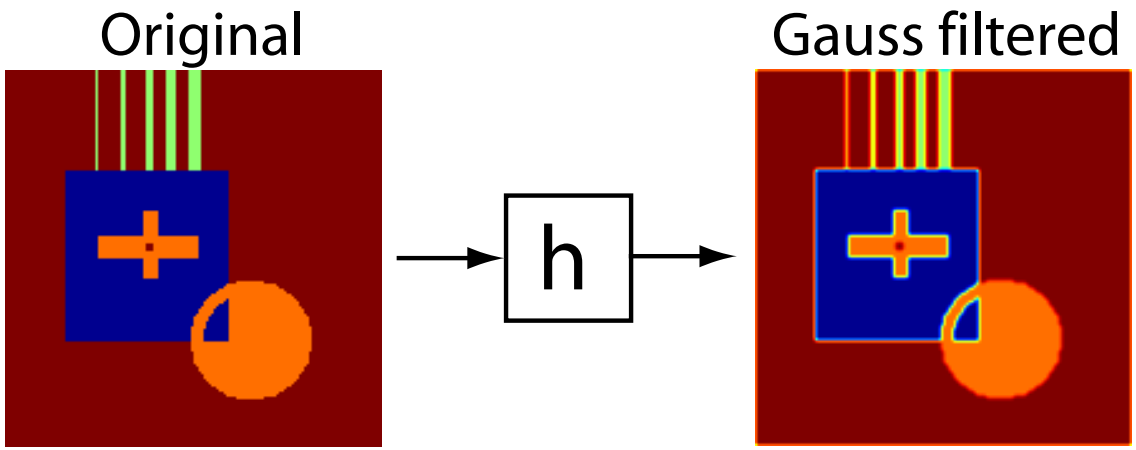
The Gaussian kernel

$$G_\sigma = \frac{1}{(2\pi\sigma)^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad (6)$$

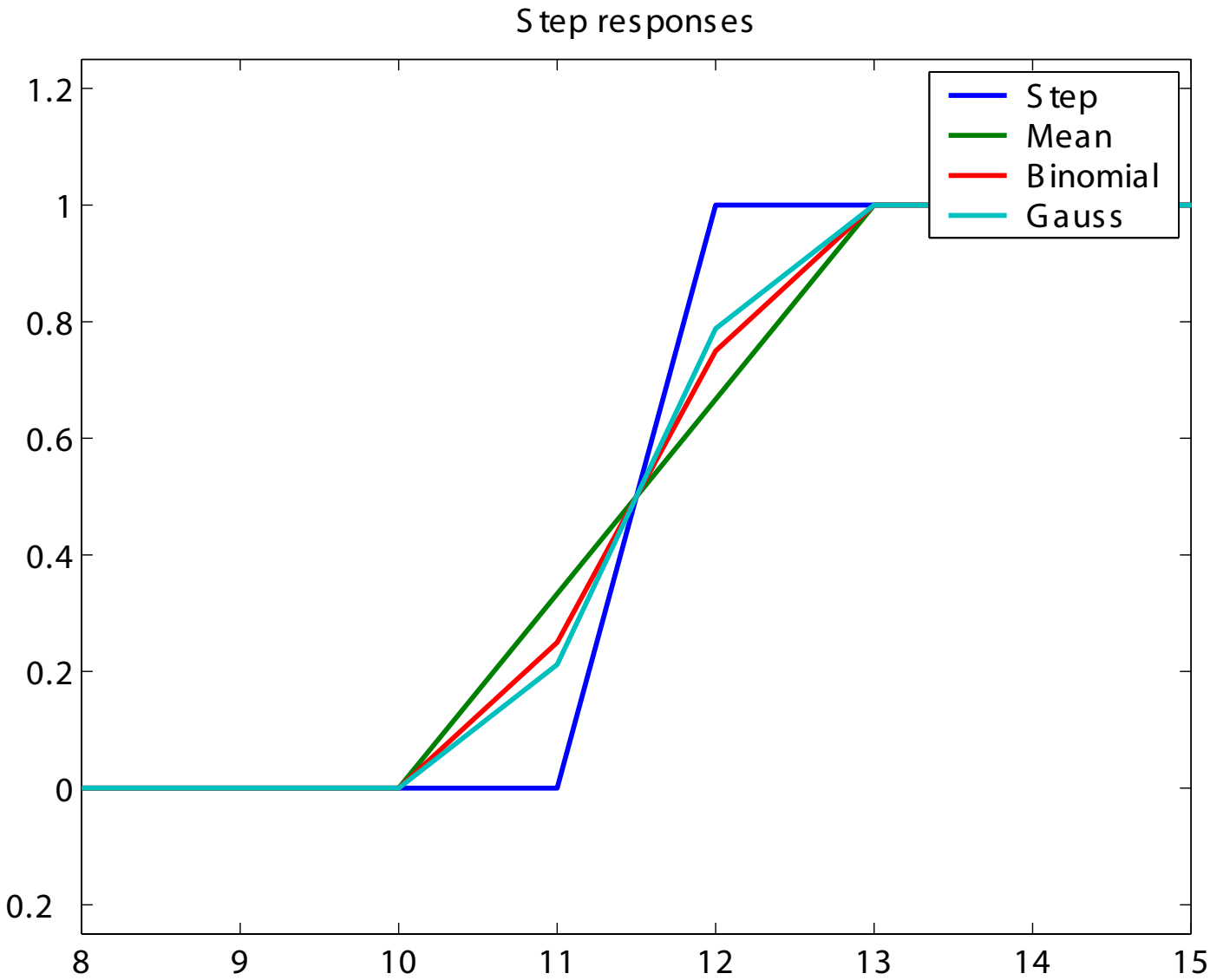
The Gaussian filter kernel size is determined by e.g. $[2n\sigma]$ discrete positions.



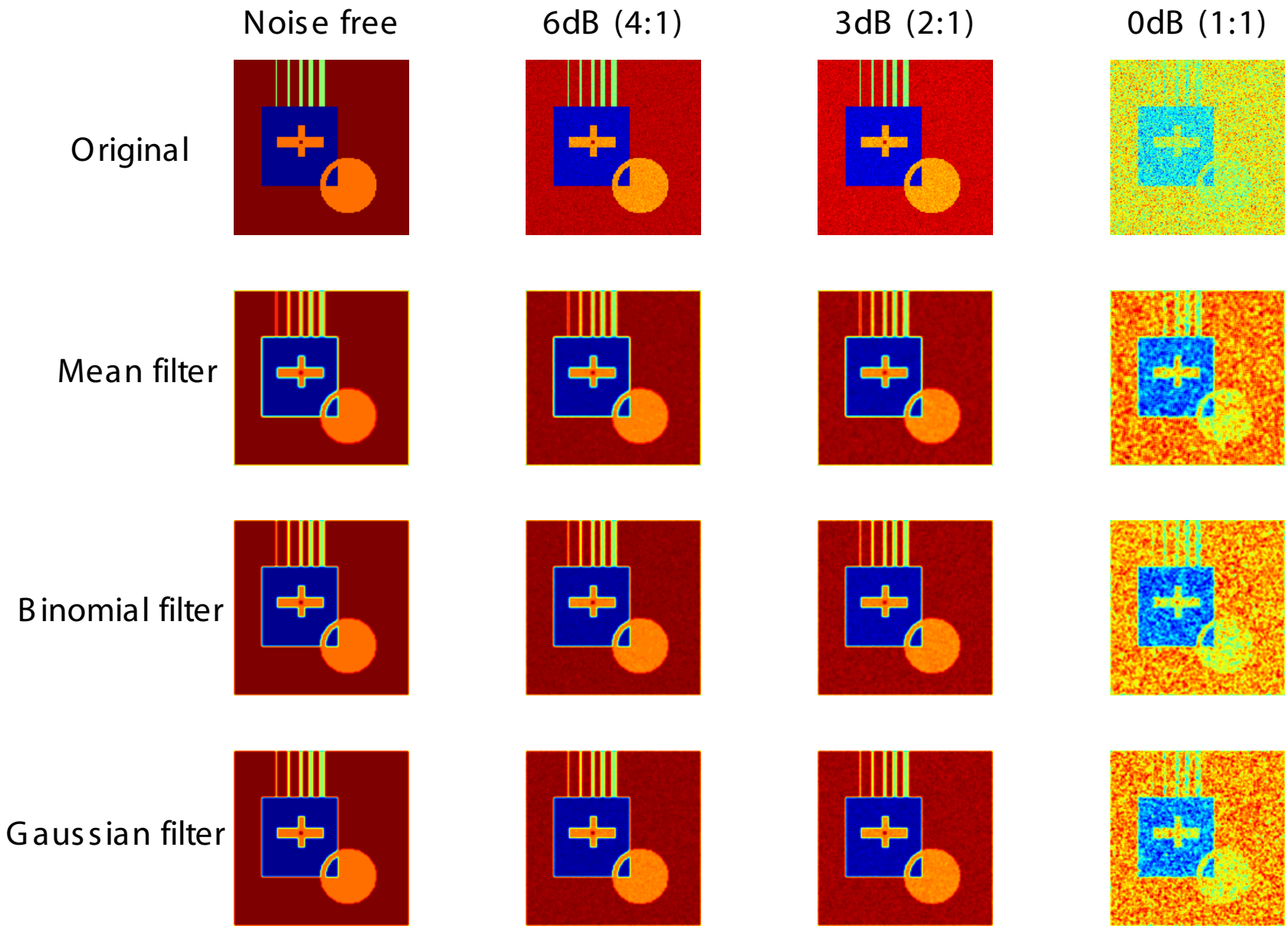
Example



Step response of low-pass filters



Low-Pass Filters with noise

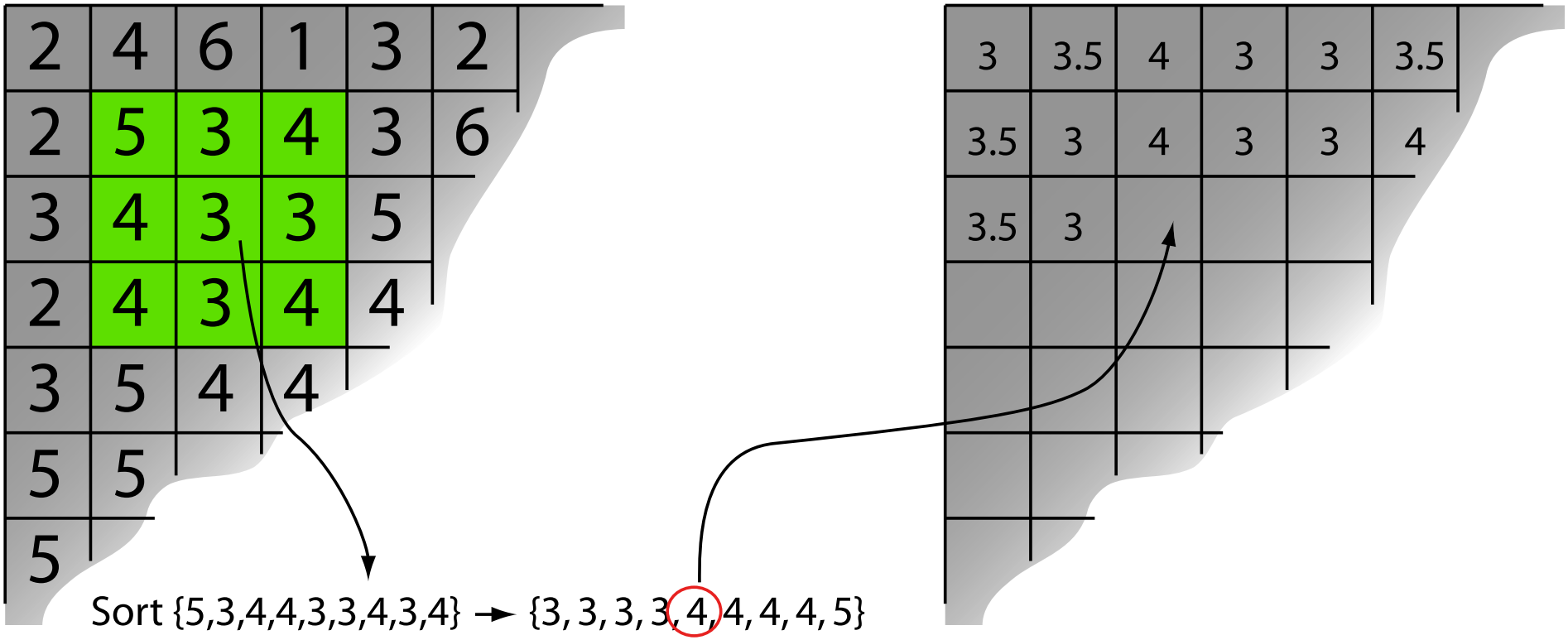


Non-linear filters for denoising

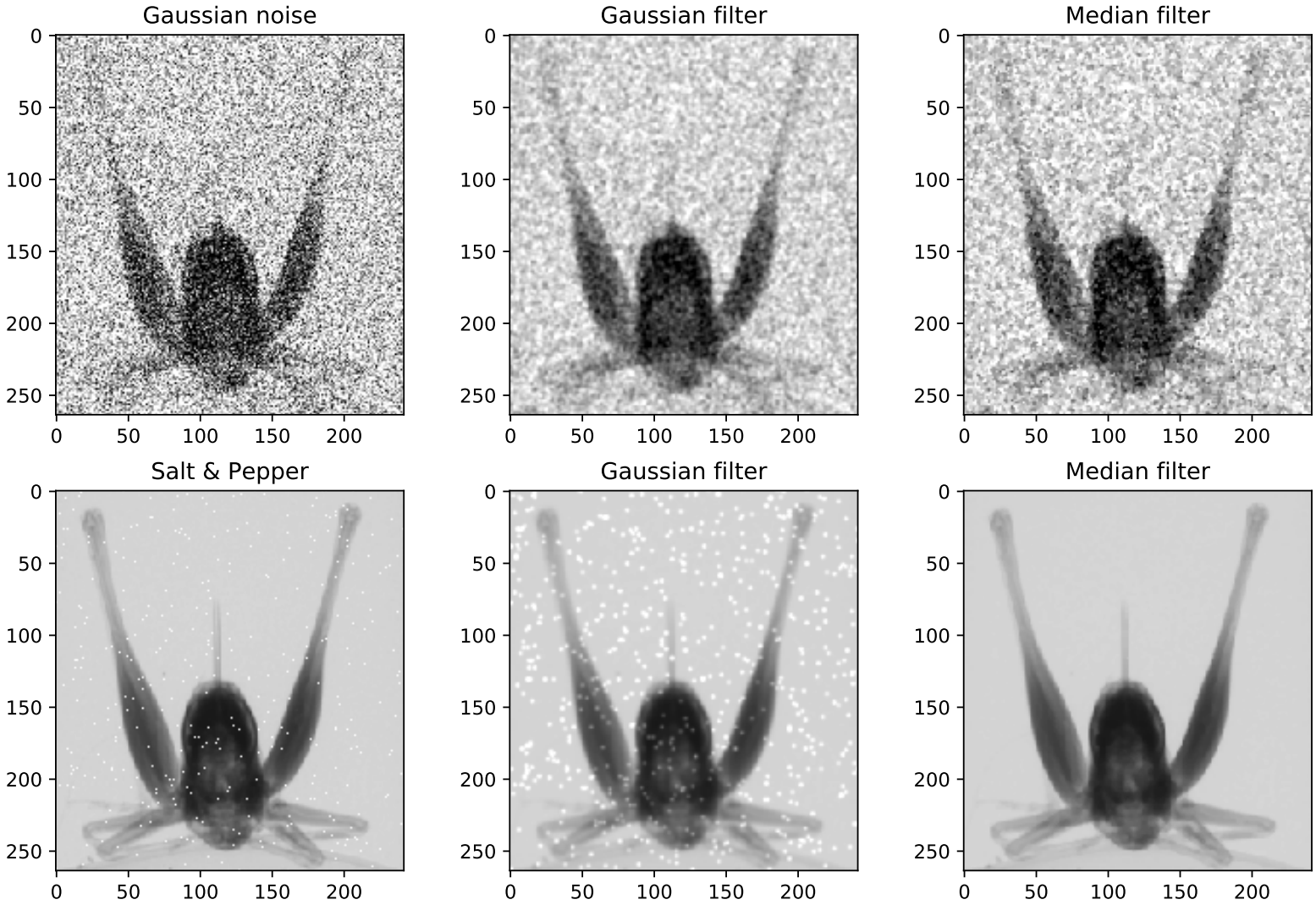
Median filter :: Principle

The median filter is a low-pass type non-linear

- Removing outlier noise such as salt& pepper-noise
- Is gentler to edges than convolution at same smoothing



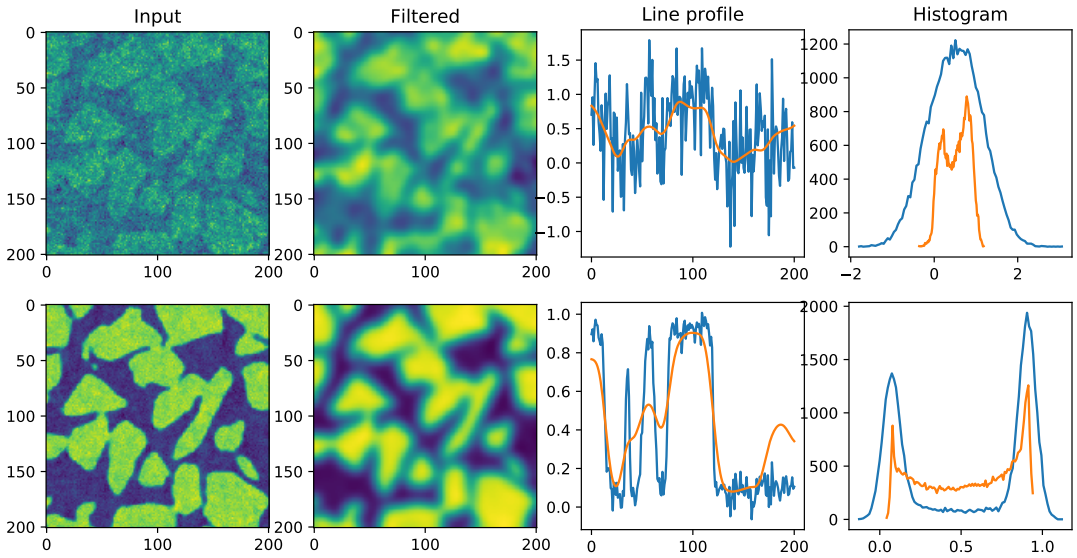
Compare median and box filters



When traditional filters are not enough

In cases with low SNR and artifacts

- Filter kernel increase
- Increased blurring
- Relevant features vanish



Demo python notebook

Advanced filters

New filters have been introduced d to handle 'ugly' images

- Non-local means
- Non-linear convolution
- ROF inverse scale space

Non-local smoothing

The idea

Smoothing normally consider information from the neighborhood like

- Local averages (convolution)
- Gradients and Curvatures (PDE filters)

Non-local smoothing average similiar intensities in a global sense.

- Every filtered pixel is a weighted average of all pixels.
- Weights computed using difference between pixel intensities.

Draw-back the computational complexity $O(N^2)$. Approximate schemes exist.

Example

Demo notebook.

Buades et al. (2005)

Filter definition

The non-local means filter is defined as

$$u(p) = \frac{1}{C(p)} \sum_{q \in \Omega} v(q) f(p, q) \tag{8}$$

where

v **and** *u* input and result images.

$C(p)$ is the sum of all pixel weights as

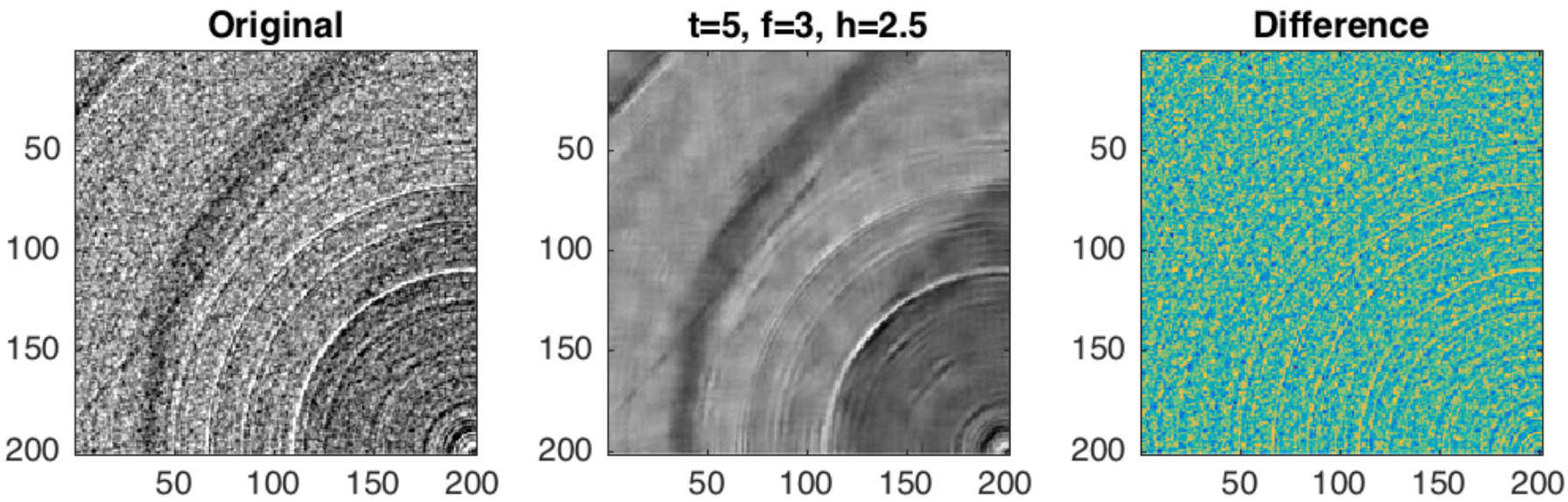
$$C(p) = \sum_{q \in \Omega} f(p, q) \tag{9}$$

$f(p, q)$ is the weighting function

$$f(p, q) = e^{-\frac{|B(q) - B(p)|^2}{h^2}} \tag{10}$$

B(x) is a neighborhood operator e.g. local average around x

Non-local means 2D – Example



Observations

- Good smoothing effect.
- Strong thin lines are preserved.
- Some patchiness related to filter parameter t, *i.e.* the size of Ω_i.

PDE based filters

Non-linear diffusion

Smooth fine features first

$$\frac{\partial u}{\partial t} = \underbrace{G(|\nabla_\sigma u|)}_{\text{Diffusivity}} \nabla^2 u$$

Inverse scale space

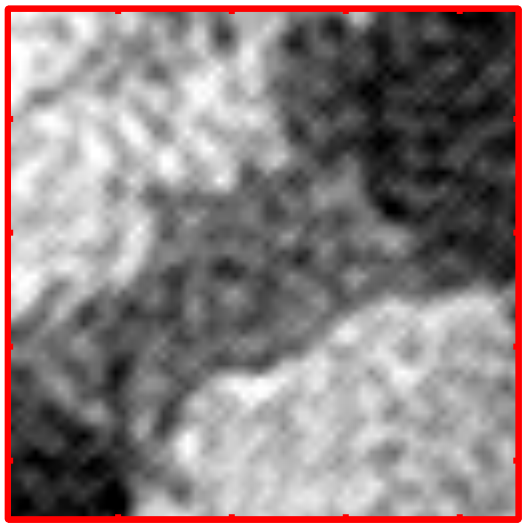
Add wide features first

$$\begin{aligned} \frac{\partial u}{\partial t} &= \operatorname{div}\left(\frac{\nabla u}{|\nabla u|}\right) + \lambda\,(u_0 - u + v) \\ \frac{\partial v}{\partial t} &= \alpha\,(u_0 - u) \end{aligned}$$

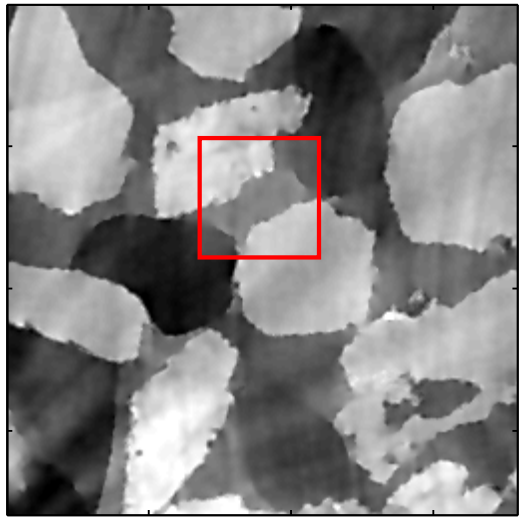
- ∇_σ Gradient smoothed by a Gaussian.
- $G(x)$ is a non-linear function with threshold behavior.
- λ Controls the strength of the filter.
- α Additional time scaling parameter (quality refinement).
- N Number of iterations.
- τ Time increment.

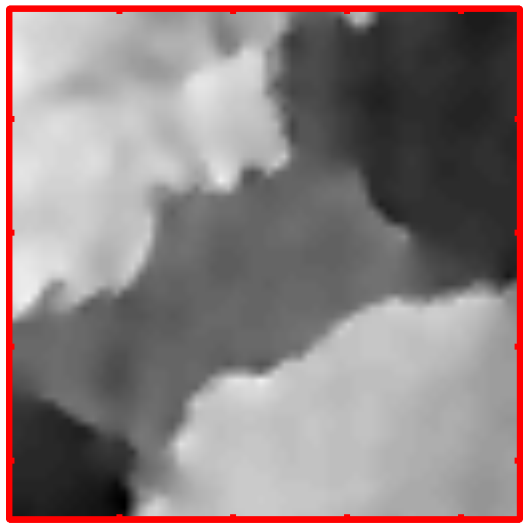
Comparing different filters

Original

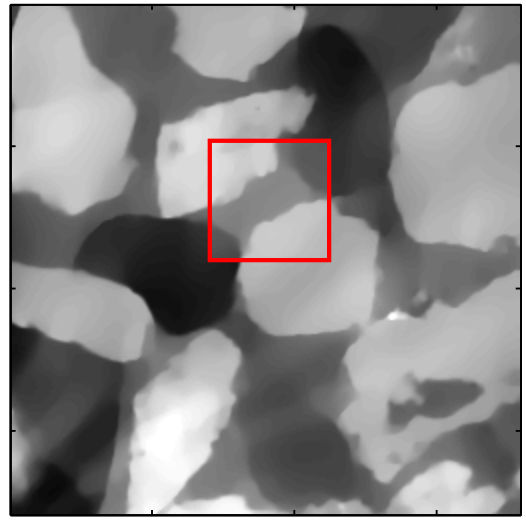


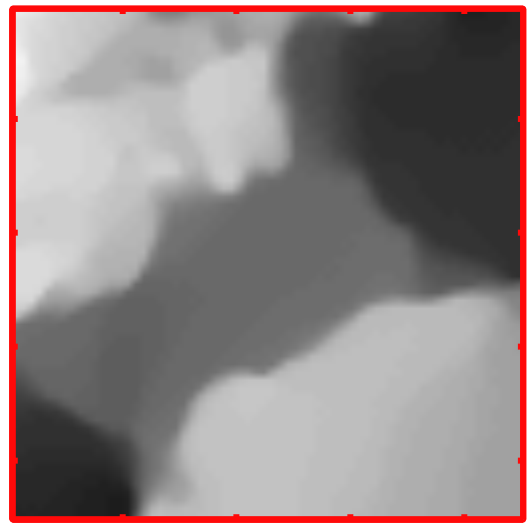
Diffusion filter





ISS filter

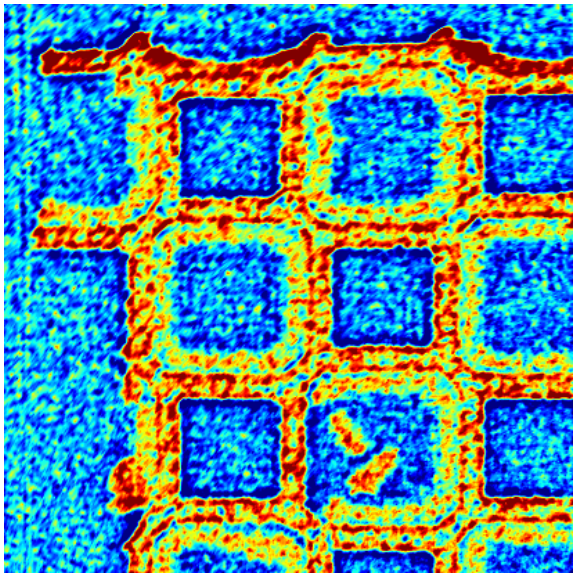




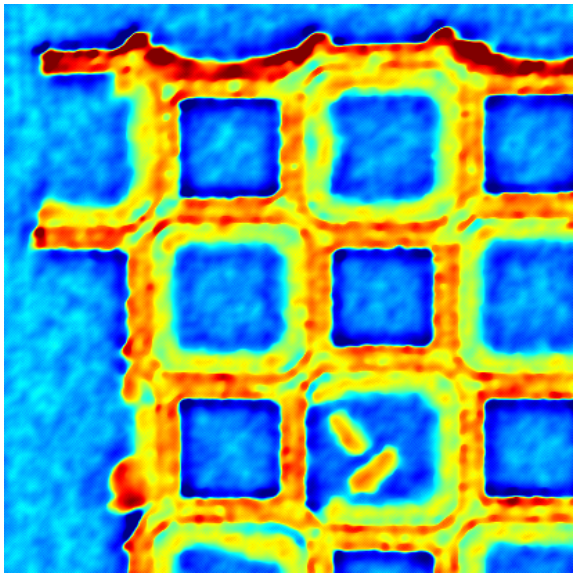
Kaestner *et al.*, AWR, 2008, Kaestner et al. (2008)

An example with the ISS filter

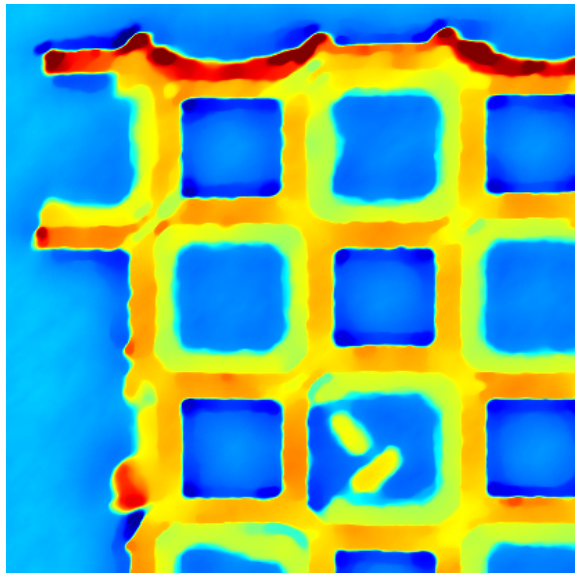
Neutron CT of a diesel particulate filter



Original



100 iterations



1000 iterations

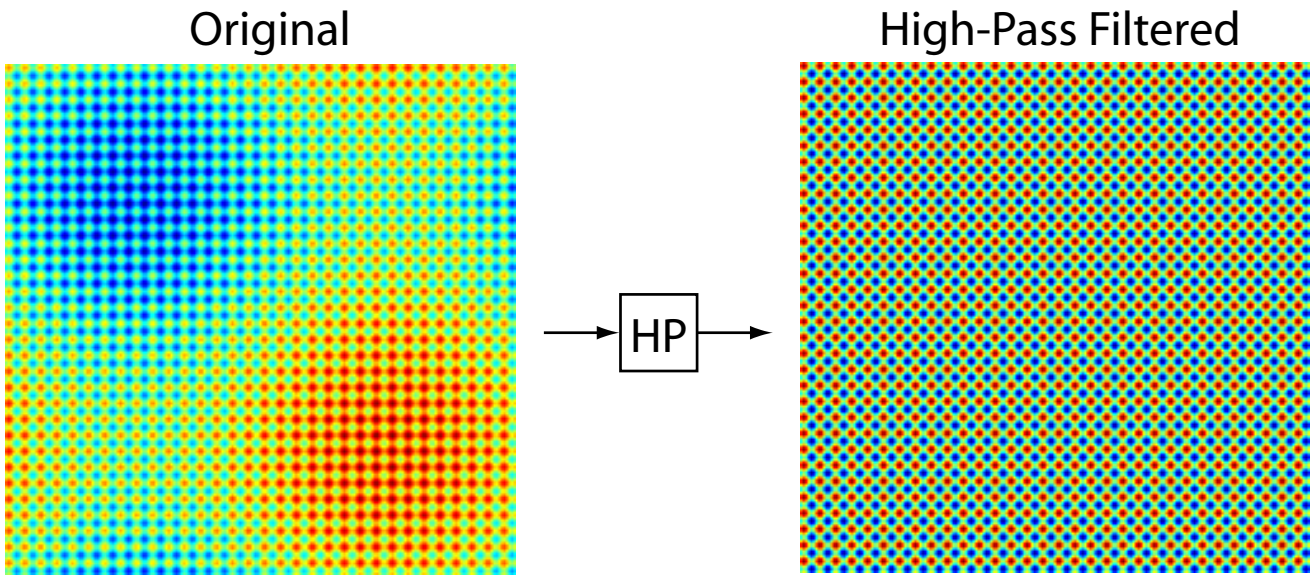
Grünzweig *et al.*, MTZ, 2012, **?**
Kaestner *et al.*, WCNDT-18, 2012, Kaestner et al. (2012)

High-pass filters and edge detection

High-pass Filters

The effects of a *High-Pass* filter are in general

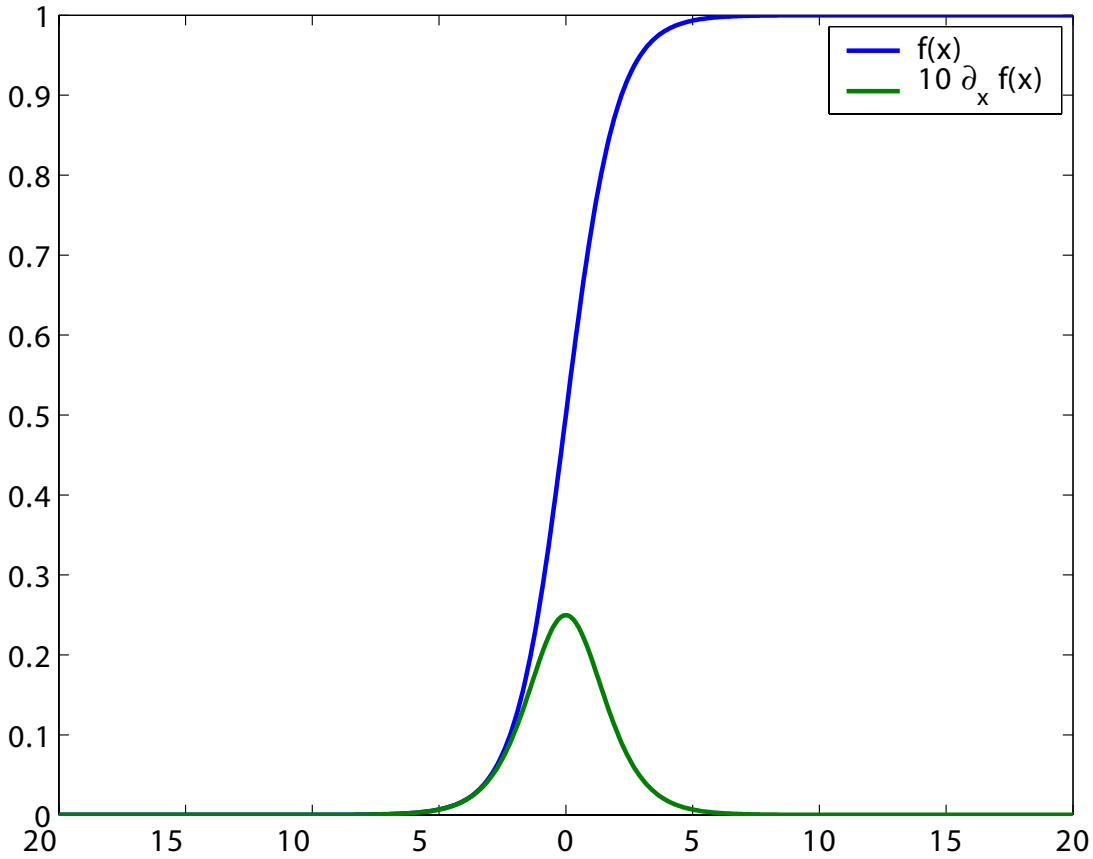
- rapid changes are enhanced
- slow changes are suppressed



High-pass Filters :: Gradients

Properties of the gradient:

- Large response for step changes
- No response for constant levels

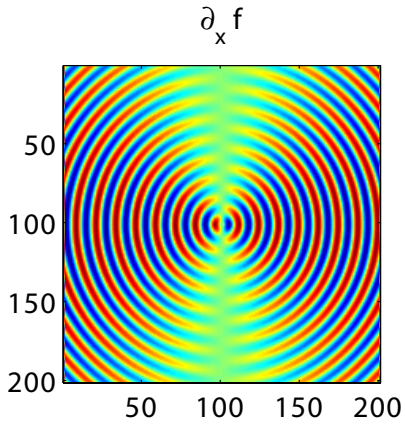


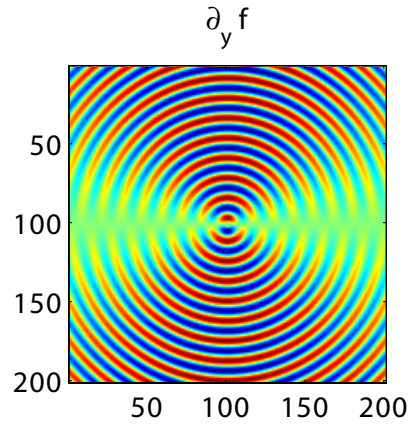
Different gradients

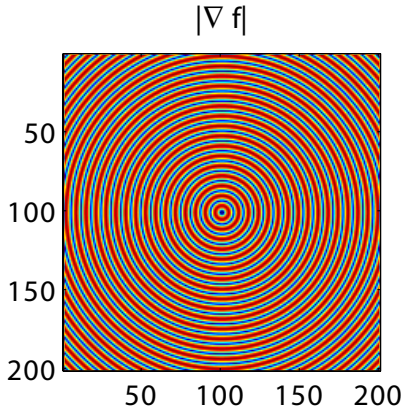
$$D_x = D_y^T = \frac{1}{2}\begin{bmatrix} 1 & -1 \end{bmatrix}$$

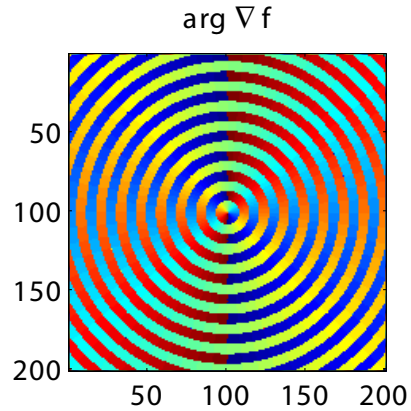
$$D_x = D_y^T = \frac{1}{2}\begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

$$D_x = D_y^T = \frac{1}{32}\begin{bmatrix} 3 & 0 & -3 \\ 10 & 0 & -10 \\ 3 & 0 & -3 \end{bmatrix}$$





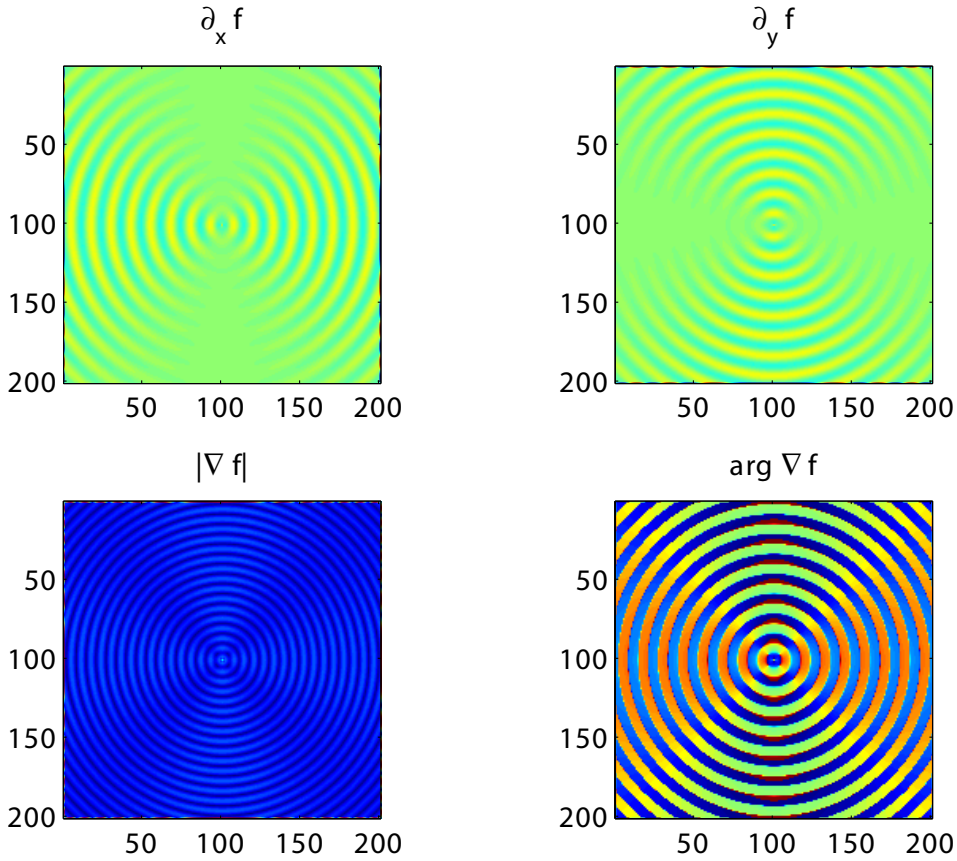




Jähne (2002)

Laplacian ($\nabla \cdot \nabla f$)

$$L_x = \frac{1}{4}\begin{bmatrix} 1 & -2 & 1 \end{bmatrix}, \qquad L_y = \frac{1}{4}\begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$



Gradient of a Real Image

Original

∂_y

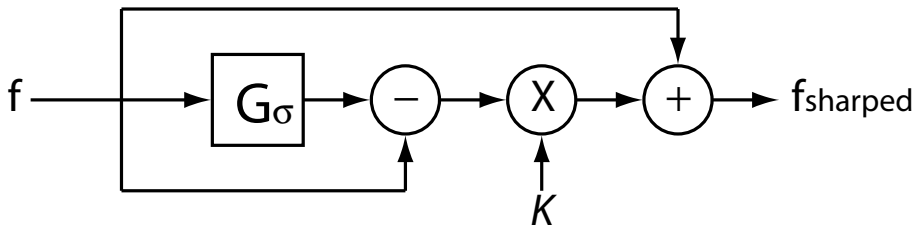
∂_x

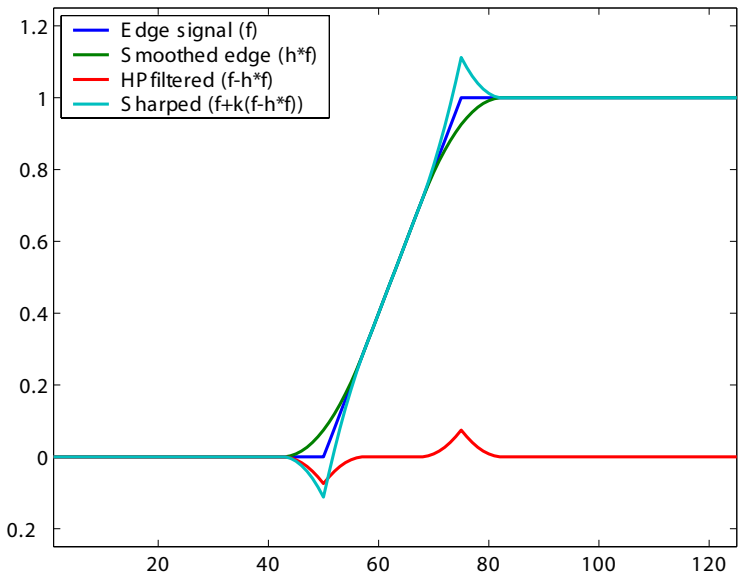
Sharpening Images

A way to enhance the edge contrast is the *unsharp mask* filter:

$$f_{sharped} = f + K\,(f - G_\sigma * f) \tag{11}$$

Block diagram

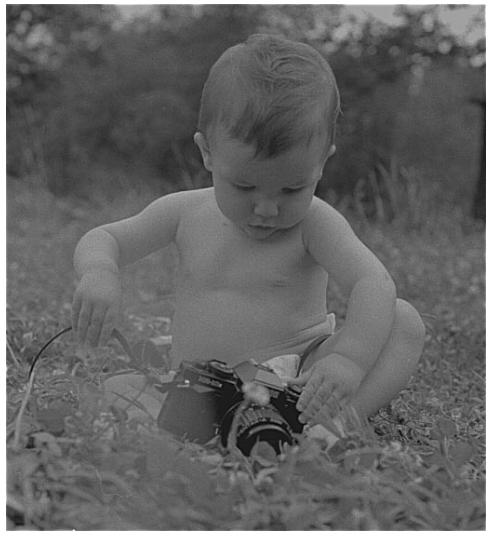




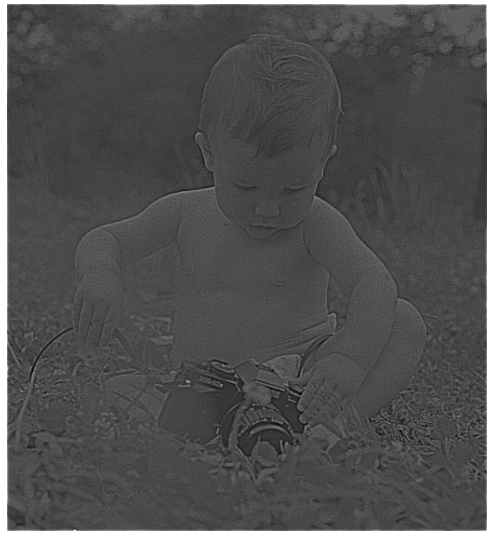
Sharpening Images – 2D Example

Original

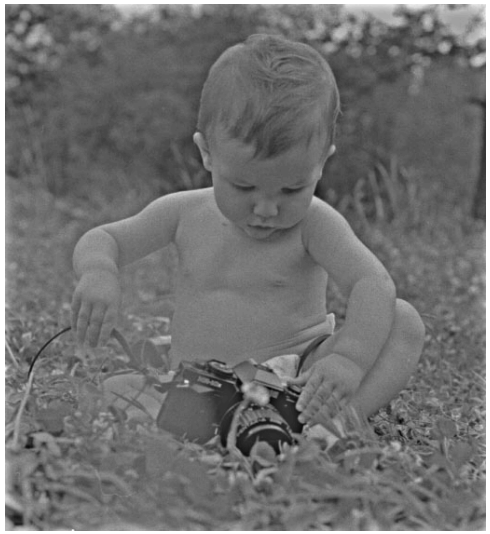
σ=1.5, K=15



σ=1.5, K=10



σ=10, K=15

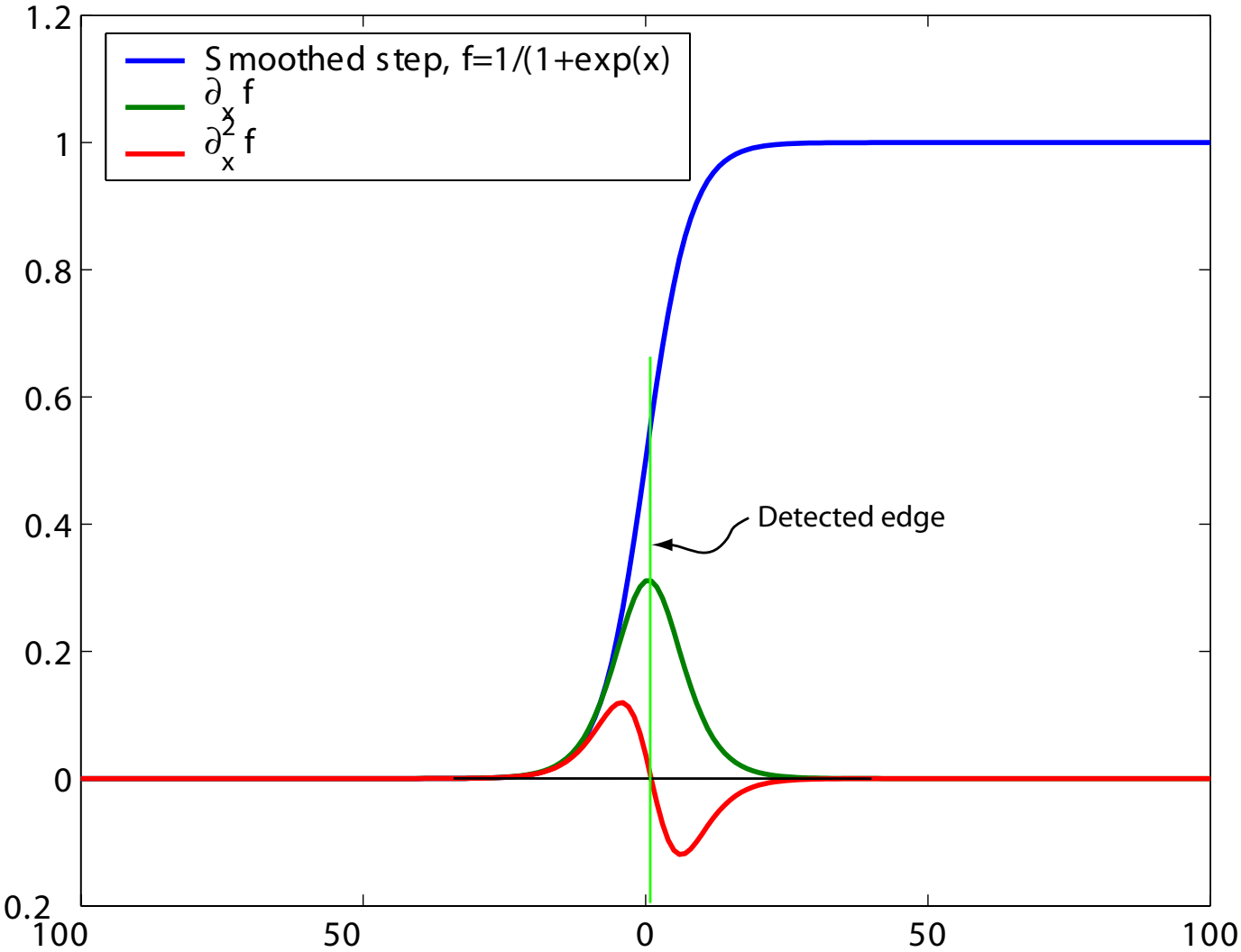


σ=10, K=10

Edge detection – Introduction

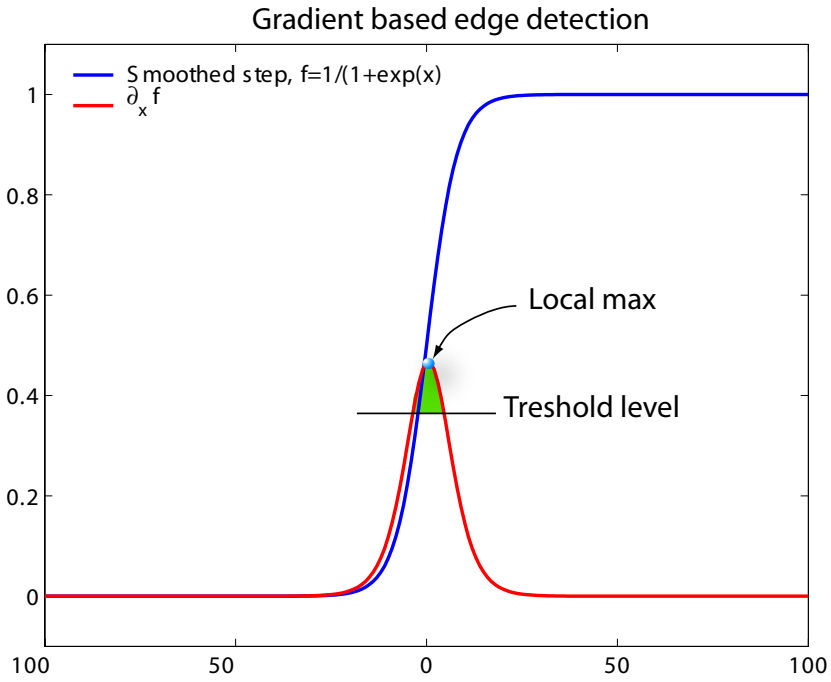
The interesting features of an edge are.

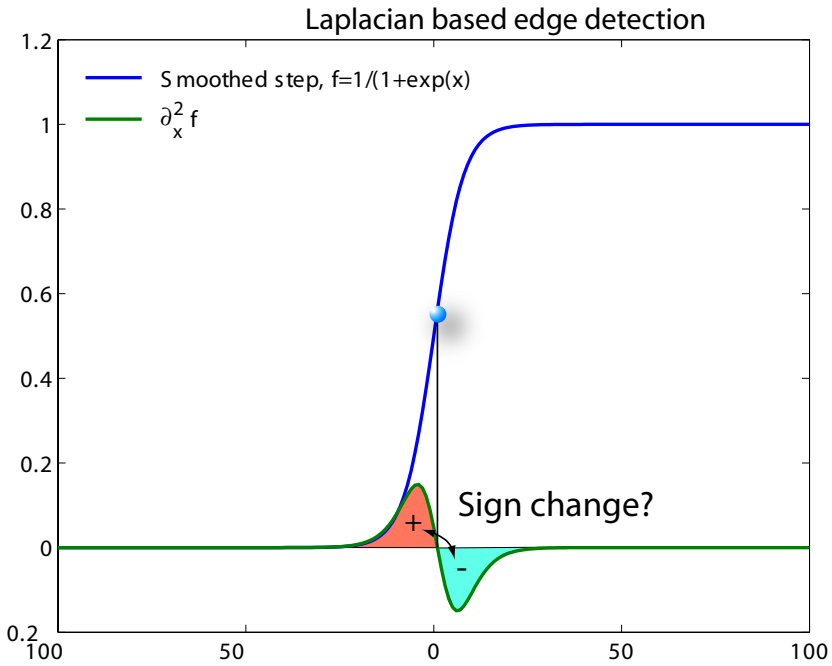
- Edges have a large high frequency component
- The gradient has a larger amplitude at the edge



Edge detection :: Principles

Edges can be detected with first and second derivatives



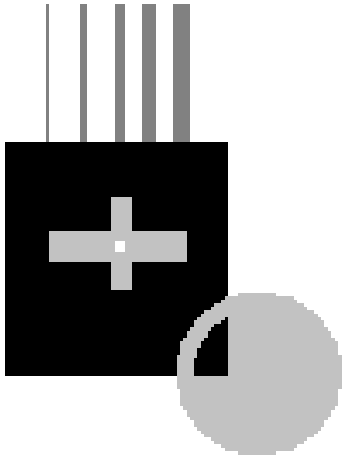


Example: Laplacian based edge detector

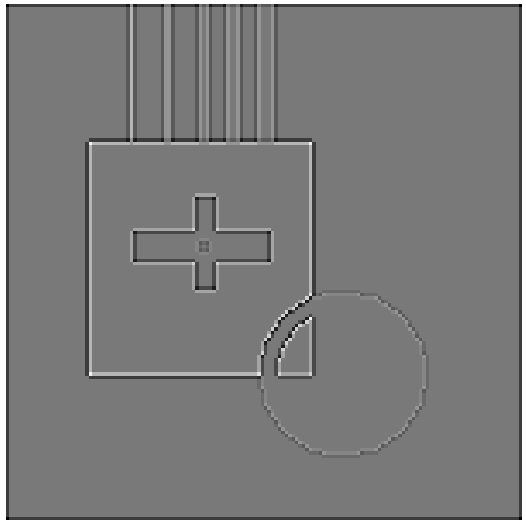
A simple Laplacian-based edge detector:

- Compute $g = L_x * f + L_y * f$, with $L_x = {L_y}^T = \begin{bmatrix} 1 & -2 & 1 \end{bmatrix}$
- Assign $g \underset{\text{Edge}}{\overset{\text{Background}}{\gtrless}} 0$

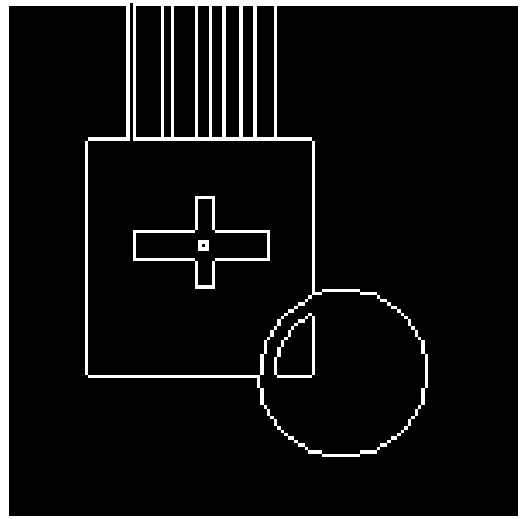
Orignial



$\nabla^2 f$



$\nabla^2 f<0$



Edge detection and noise

Problem

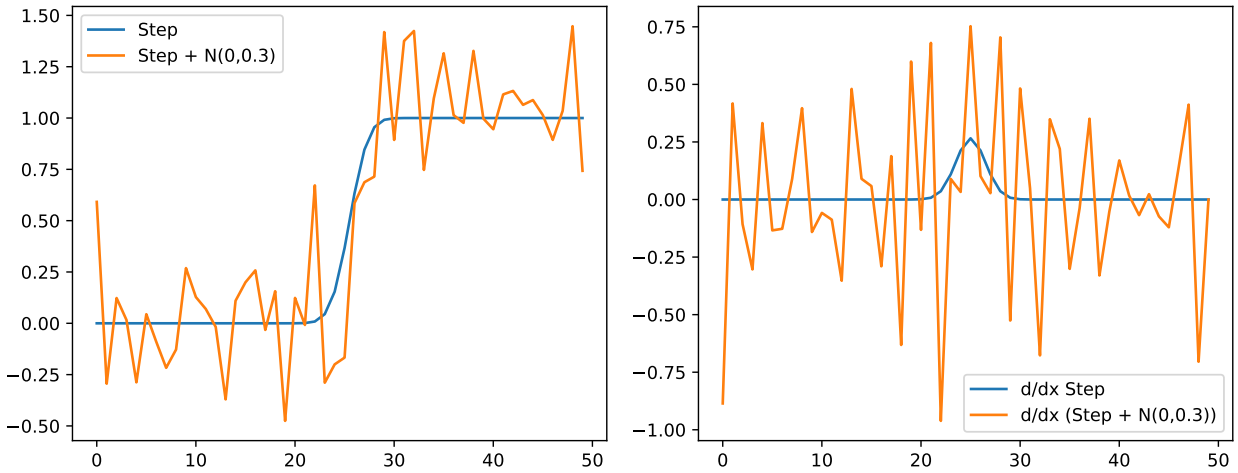
The large problem for an edge detector is noise, it can

- *Hide relevant edges*
- *Result in undesired edges*

Solutions

- Smooth image prior to edge detections
- Track edges (search in the direction of the gradient)

Example



Edge detection :: Implementations

Edge detection robustness is improved by perpendicular smoothing.
Popular edge detectors are:

Sobel edge detector masks

−1	−2	−1
1	2	1

−1		1
−2		2
−1		1

Prewitt edge detector masks

−1	−1	−1
1	1	1

−1		1
−1		1
−1		1

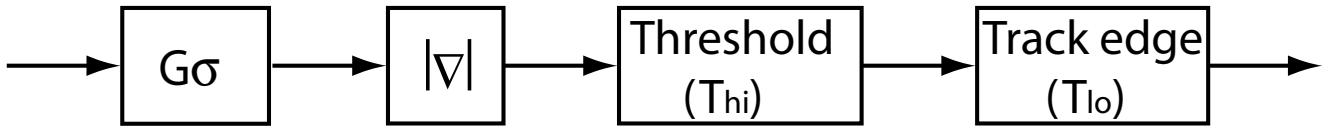
See e.g. Haralik and Shapiro (1992)

Canny edge detector

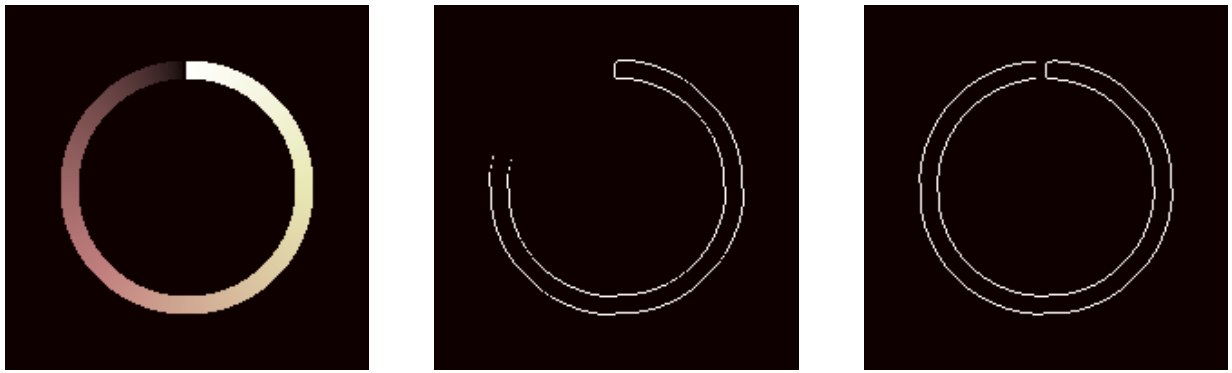
The edge detection performance can be improved by combining

- Smoothing
- Thresholding
- Tracking

Block diagram



Example



Original Sobel Canny

Canny (1986)

Filters in the frequency domain

The Fourier transform

Changing the representation of an image by

- Mapping the image information to a new domain
- The spatial frequency components are explored

$$G(\xi_1, \xi_2) = \mathcal{F}\{g\} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) e^{-i(\xi_1 x + \xi_2 y)} \, dx \, dy$$

Some useful properties of the FT

- Addition

$$a f(x) + b g(x) \Leftrightarrow a F(\omega) + b G(\omega)$$

- Convolution

$$f * g(t) \Leftrightarrow F(\omega) \cdot G(\omega)$$

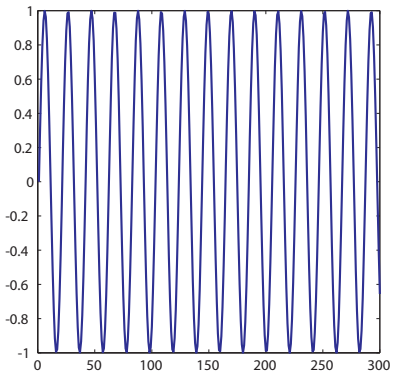
- Symmetry

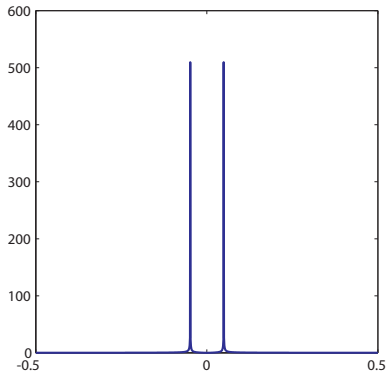
$$x \in \mathbf{R} \Rightarrow F(\omega) = F(-\omega)$$

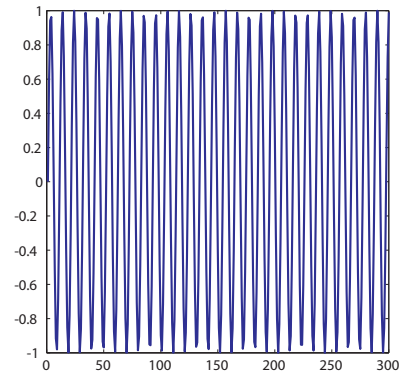
General Jähne Jähne (2002) or Detailed Granlund Granlund and Knutsson (1995)

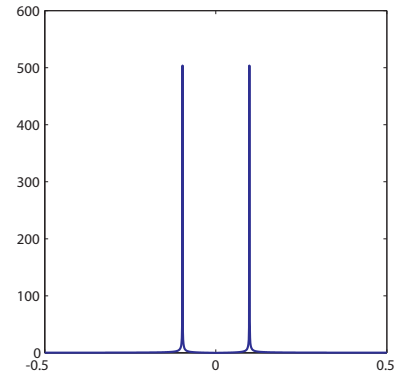
Addition of 1D signals

$f(t)$ and $\mathcal{F}\{f\}$ + $g(t)$ and $\mathcal{F}\{g\}$

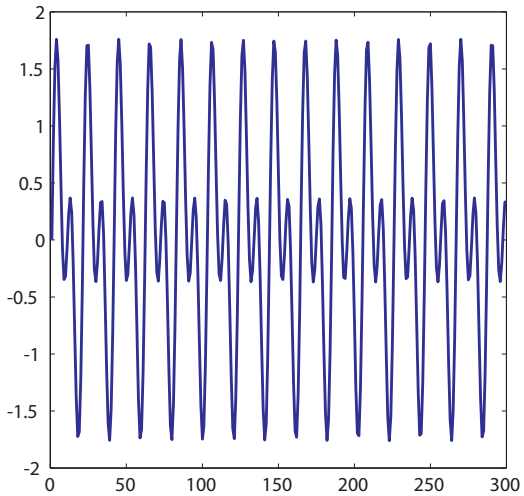


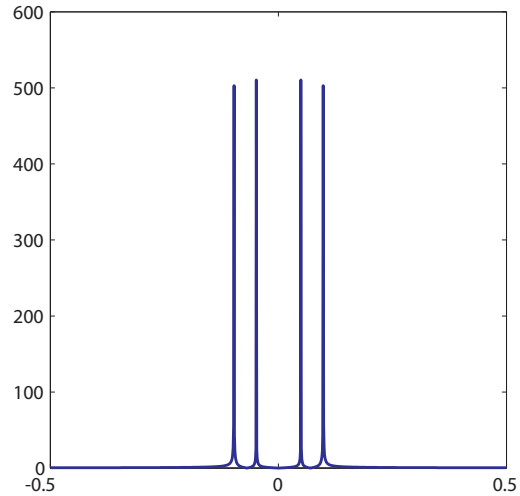




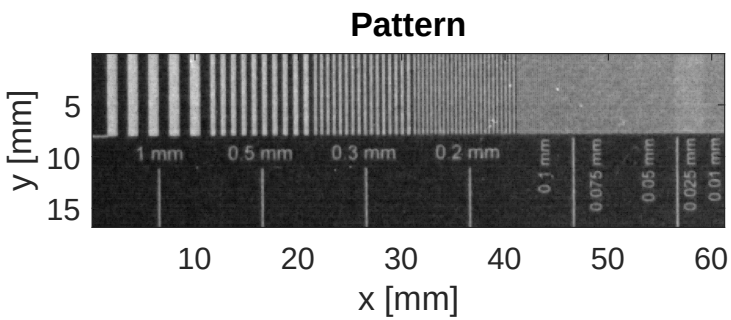


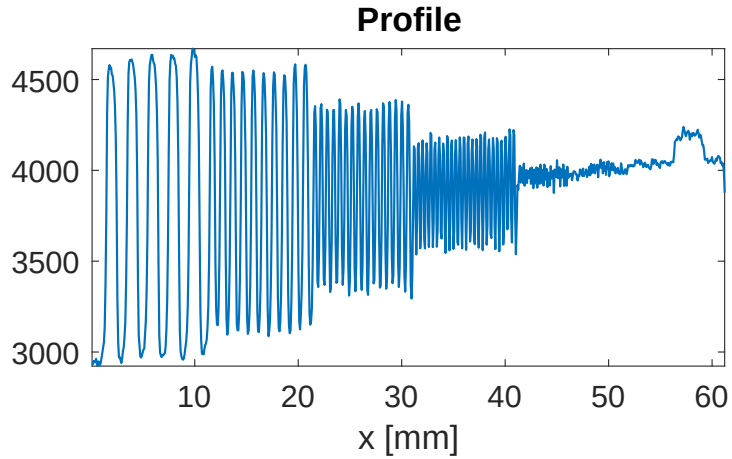
$f(t) + g(t)$ and $\mathcal{F}\{f\} + \mathcal{F}\{g\}$

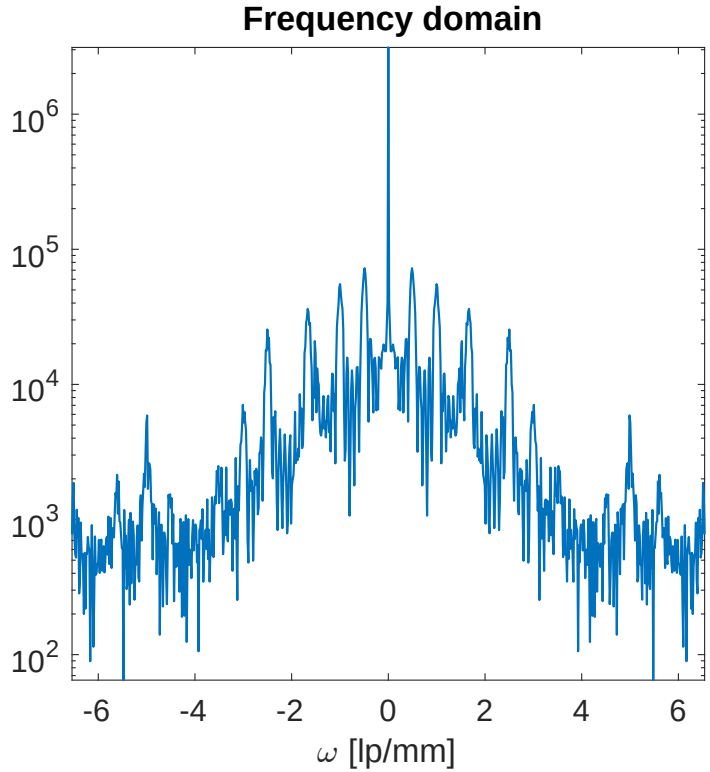




Fourier transform of real line pattern







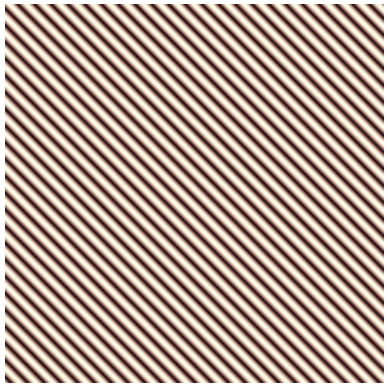
What we can see:

- Peaks for each line frequency
- Smoothing window
- Noise level

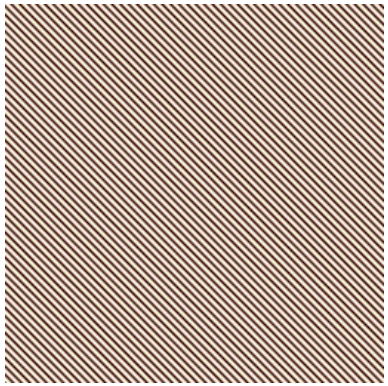
Fourier transform in 2D – Spatial frequencies

Spatial domain

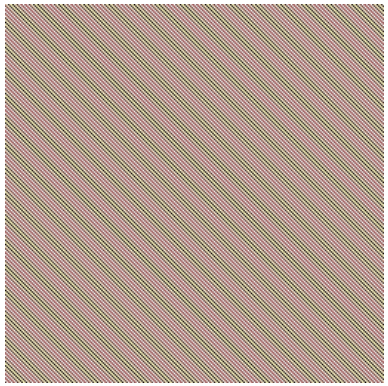
$\omega = 0.05$



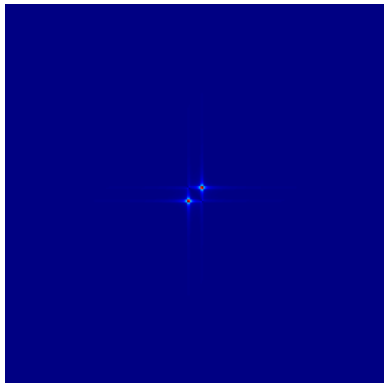
$\omega = 0.25$

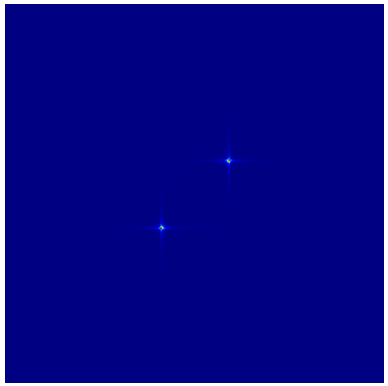


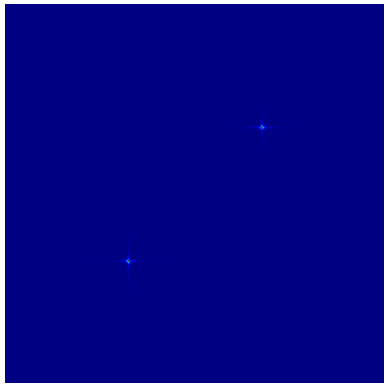
$\omega = 0.5$

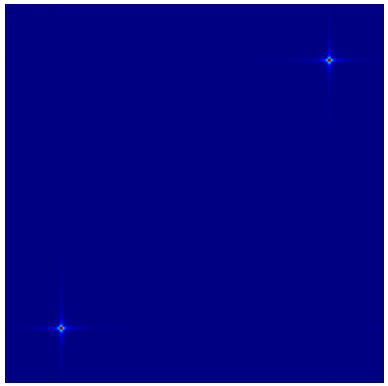


$\omega = 1.0$





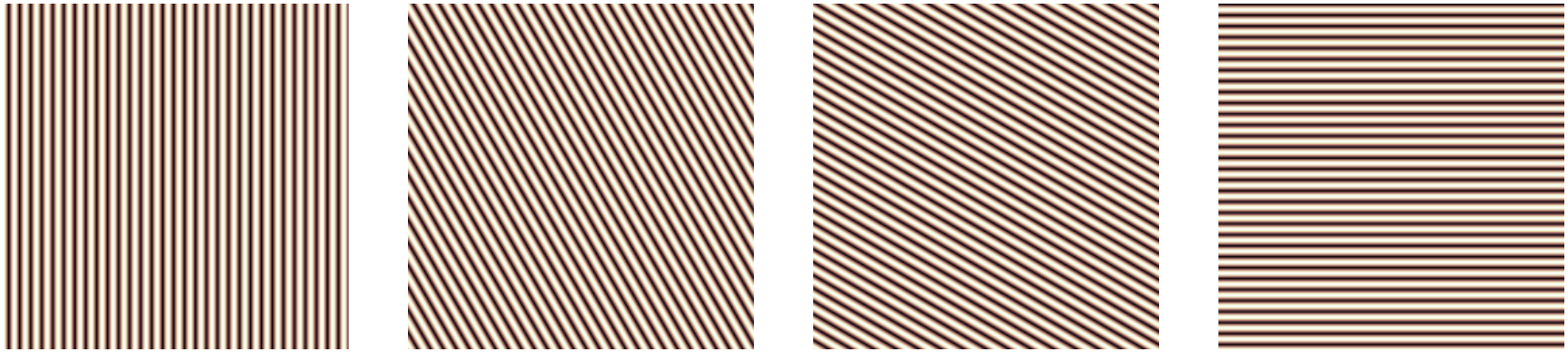




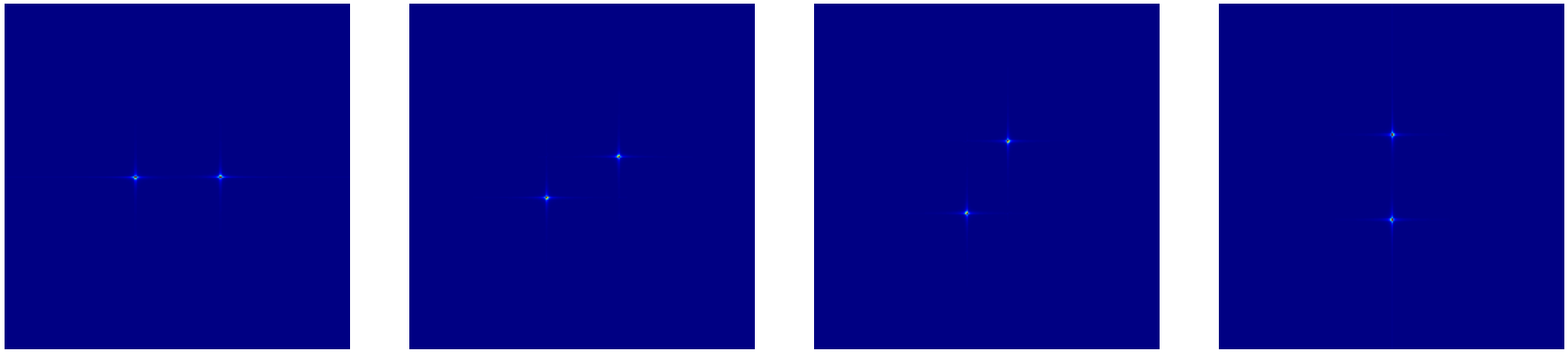
Frequency domain

2D FFT – Orientations

Spatial domain



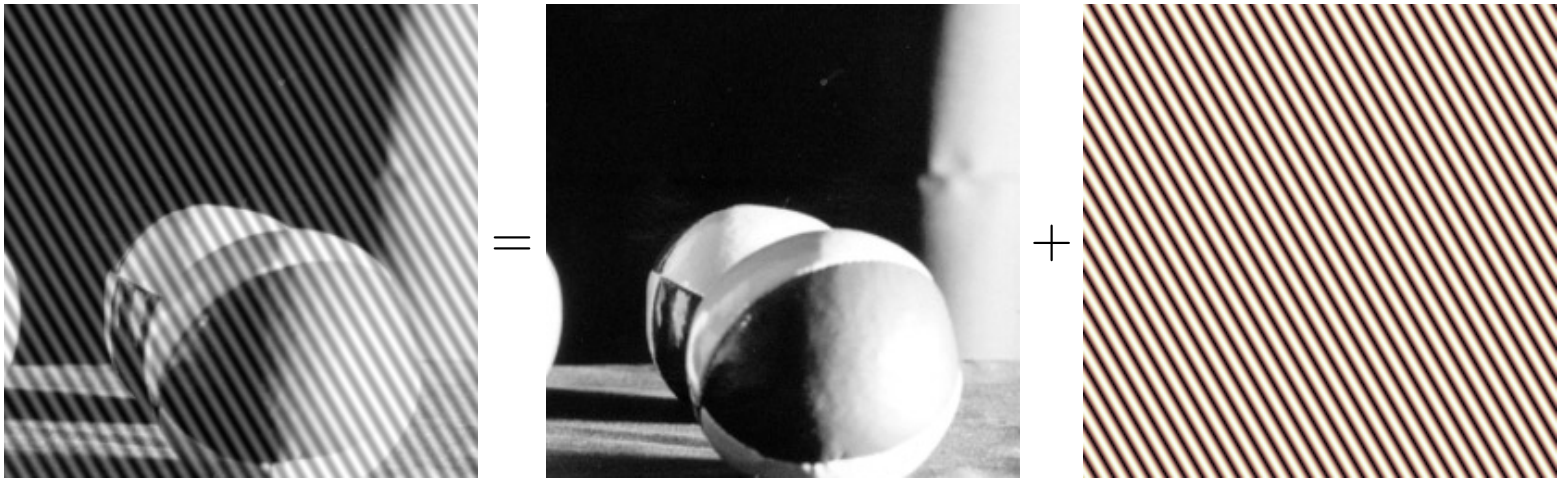
$\theta = 0$ $\theta = 30°$ $\theta = 60°$ $\theta = 90°$



Frequency domain

Orientation sensitive filters

You have



Filter with a band stop filter defined as

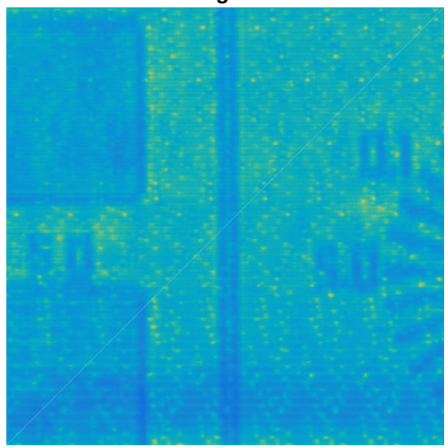
$$H(\omega_1, \omega_2) = \frac{1}{1 + A e^{-\frac{(\omega_1 \pm \omega_c \cos(\alpha))^2 - (\omega_2 \pm \omega_c \sin(\alpha))^2)}{2\sigma^2}}} \tag{12}$$

The image is processed by

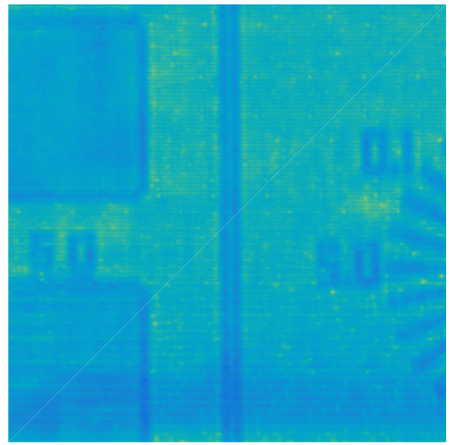
$$f_{filtered} = \mathcal{F}^{-1}\{H \cdot \mathcal{F}\{f_{raw}\}\}$$

Filter cyclic patterns

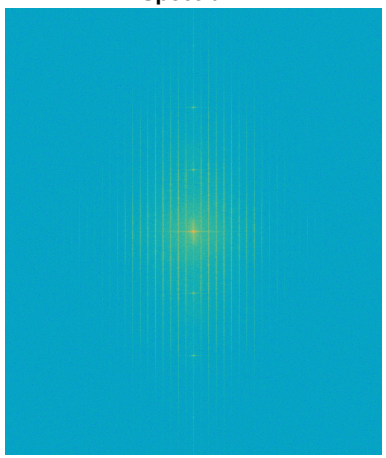
Original



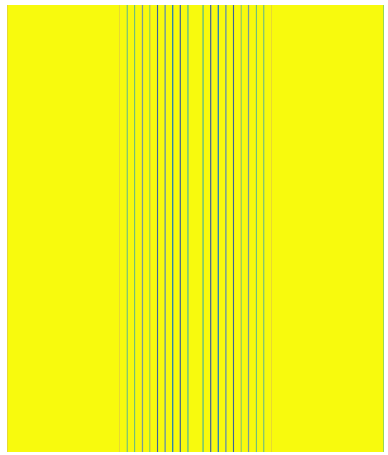
Filtered



Spectrum



Filter



Summary

This part mainly covered spatial filter such as

- Linear spatially invariant filter
  - Low-Pass filters for Smoothing
  - High-Pass filters for Edge enhancement and detection
- Non-linear filters

References I

Buades, A., Coll, B., and Morel, J. (2005). A non-local algorithm for image denoising. In *IEEE Int. Conf. on Computer Vision and Pattern Recognition*.

Canny, J. (1986). A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698.

Granlund, G. H. and Knutsson, H. (1995). *Signal processing for computer vision*. Kluwer Academic Publishers.

Haralik, R. and Shapiro, L. (1992). *Computer and robot vision*. Addison Wesley.

Jähne, B. (2002). *Digital image processing*. Springer Verlag, 2 edition.

Jain, A. (1989). *Fundamentals of digital image processing*. Prentice Hall.

Kaestner, A., Grünzweig, C., and Lehmann, E. (2012). Improved analysis techniques for new applications in neutron imaging. In *Proc 18th World Conference on Nondestructive Testing*, pages 1–10.

Kaestner, A., Lehmann, E., and Stampanoni, M. (2008). Imaging and image processing in porous media research. *Advances in Water Resources*, 31:1174–1187.