

Image filters

Anders Kaestner :: Laboratory for Neutron Scattering and Imaging



1 Defining filters**2 Low pass filters and denoising****3 Non-linear filters for denoising****4 High pass filters****5 Filters in the frequency domain****6 Summary**

Defining filters

Definition

A filter modifies the information in the image given a neighborhood.

Purpose: Enhance or suppress information.

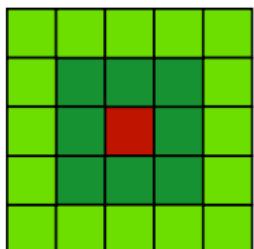
Mainly two classes of filters will be covered here:

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

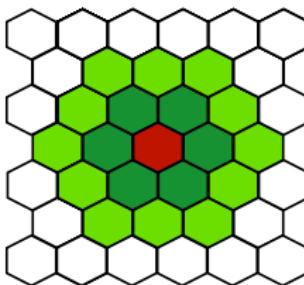
Books that cover filters are e.g. Jähne (2002) or Jain (1989)

Definition

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



- Center pixel
- 3x3-neighborhood
- 5x5-neighborhood



- Center pixel
- 6-neighborhood
- 18-neighborhood

Definitions

Continuous

$$g(x) = h * f(x) = \int_{\Omega_h} f(x - \tau) h(\tau) d\tau \quad (1)$$

Discrete

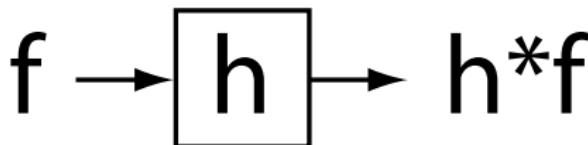
$$g[x] = h * f[x] = \sum_{p \in \Omega_h} f[x - p] h[p] \quad (2)$$

where

$f(x)$ is the image to filter

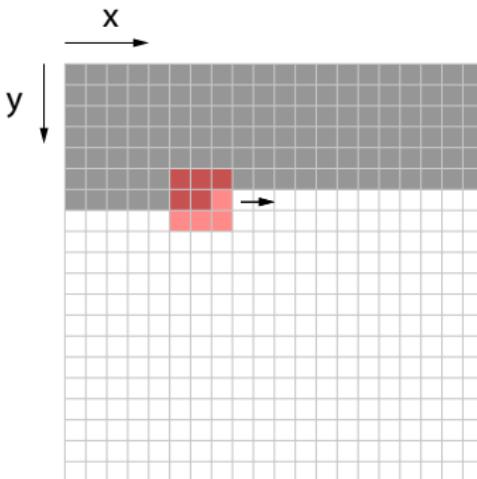
h is the convolution kernel of the filter

Block diagram



Discrete convolution is computed as :

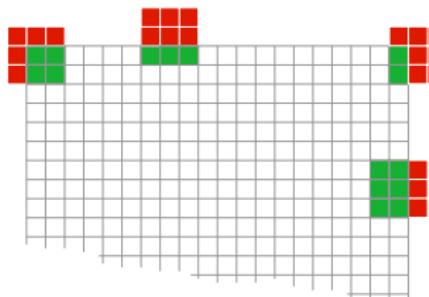
$$g[x] = f[x] * h = \sum_{\tau \in \Omega} f[x - \tau] h[\tau] \quad (3)$$



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

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

Definition

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

$$\begin{array}{|c|c|c|} \hline \cdot & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot \\ \hline \end{array} = \begin{array}{|c|} \hline \cdot \\ \hline \cdot \\ \hline \cdot \\ \hline \end{array} * \begin{array}{|c|c|c|c|} \hline \cdot & \cdot & \cdot & \cdot \\ \hline \end{array}$$

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

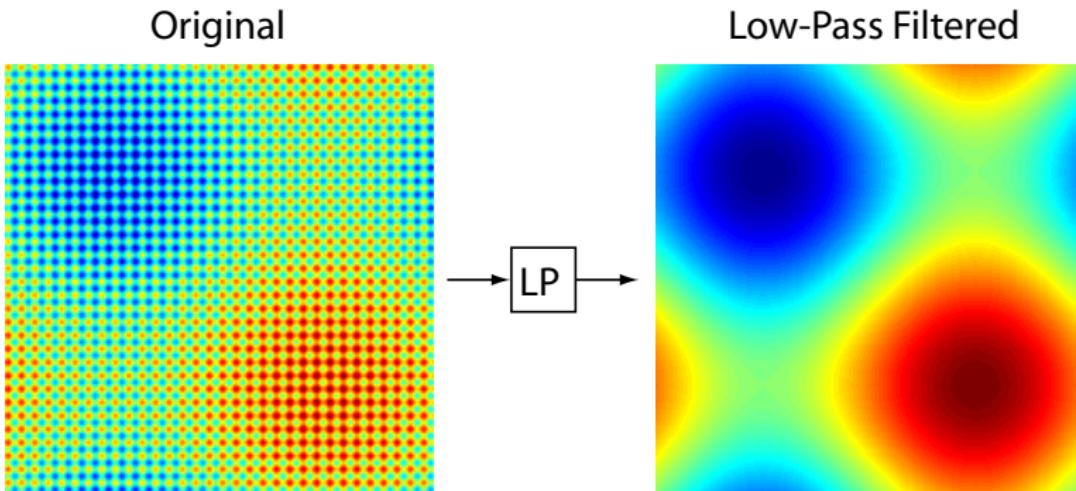
Example

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

Low-pass filters and denoising

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

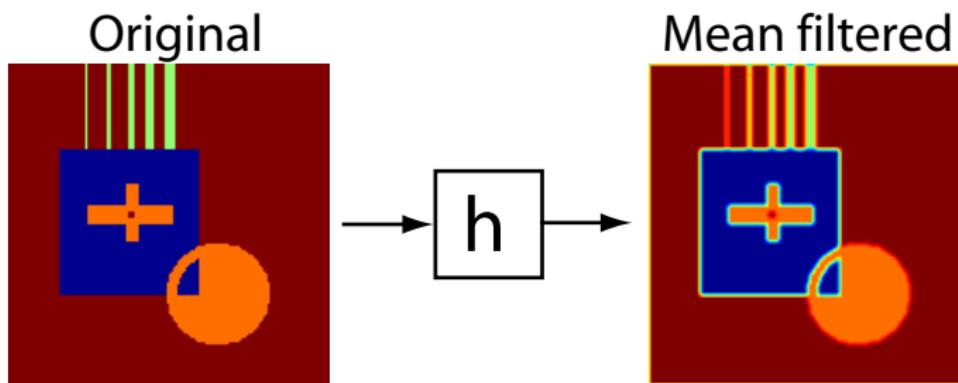
- slow changes are enhanced
- rapid changes are suppressed



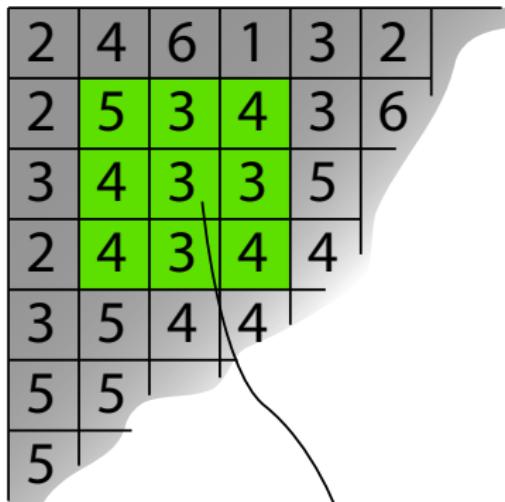
Computes the local average value of the pixel neighborhood
A 3×3 box filter is defined as

$$h = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = h_x * h_y = \frac{1}{3} [1 \ 1 \ 1] * \frac{1}{3} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad (4)$$

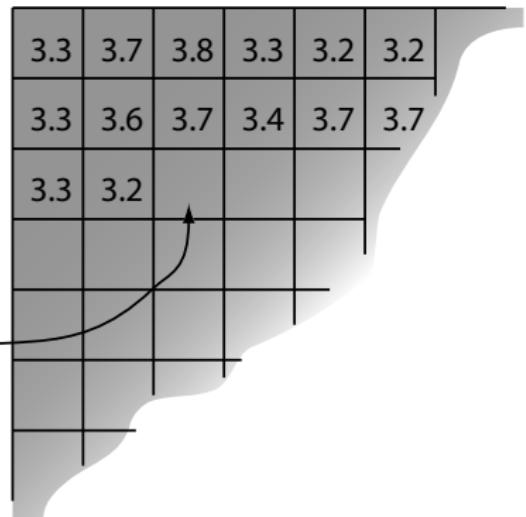
The mean filter is also called a *mean filter*



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



$$(5+3+4+4+3+3+4+3+4)/9=3.7$$



The two kernels $B_x = \frac{1}{2} [\begin{array}{cc} 1 & 1 \end{array}]$ and $B_y = \frac{1}{2} [\begin{array}{c} 1 \\ 1 \end{array}]$

1D filter coefficients are taken from Pascal's triangle

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

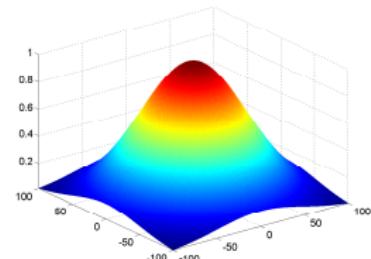
A 3×3 -kernel is defined as

$$B = B_x * B_x * B_y * B_y = \frac{1}{16} \left[\begin{array}{ccc} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{array} \right] \quad (5)$$

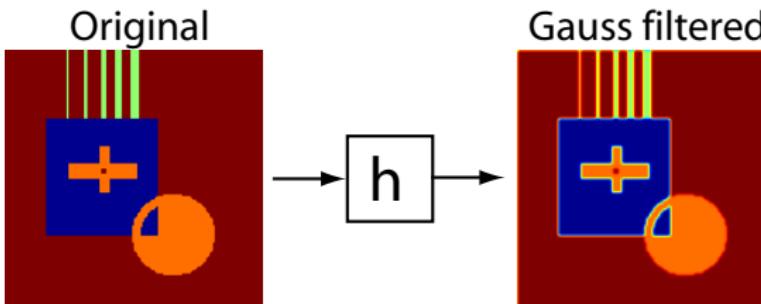
The Gaussian kernel

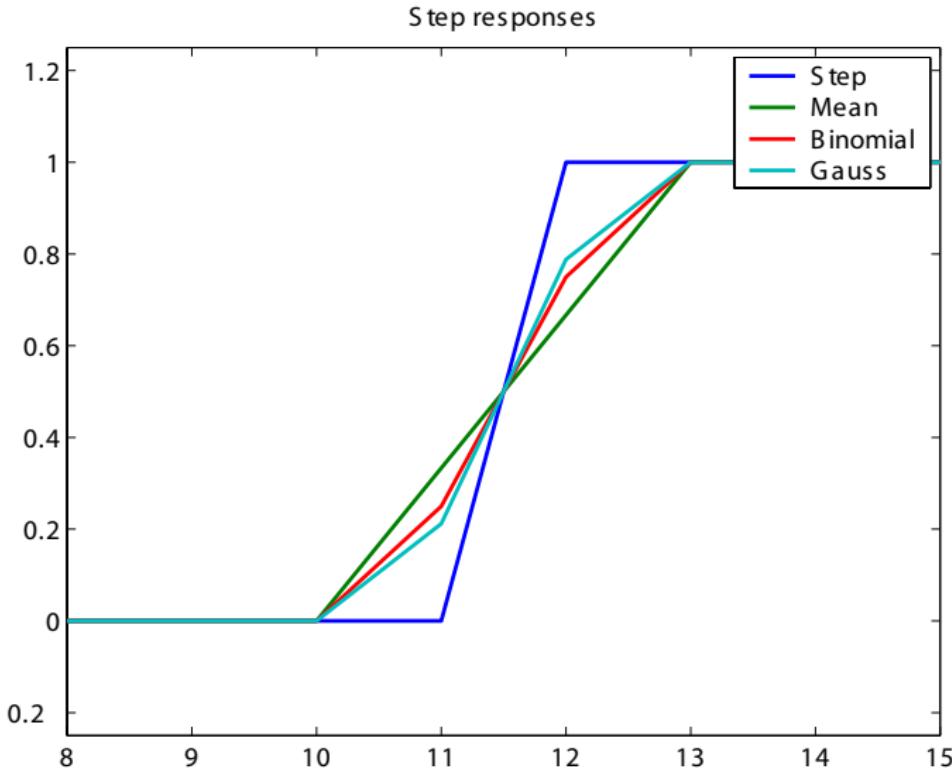
$$G_\sigma = \frac{1}{(2\pi\sigma)^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (6)$$

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

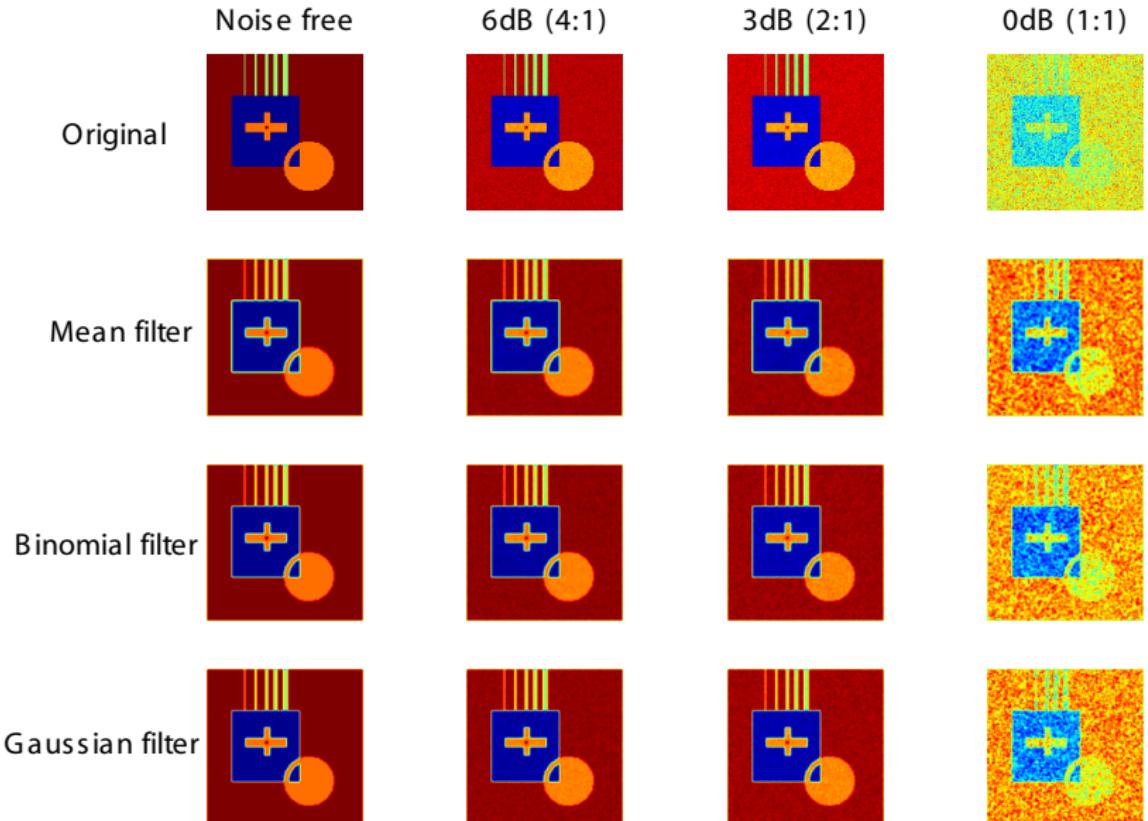


Example





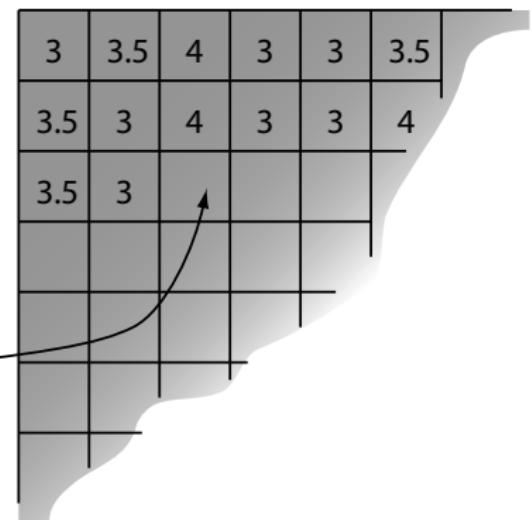
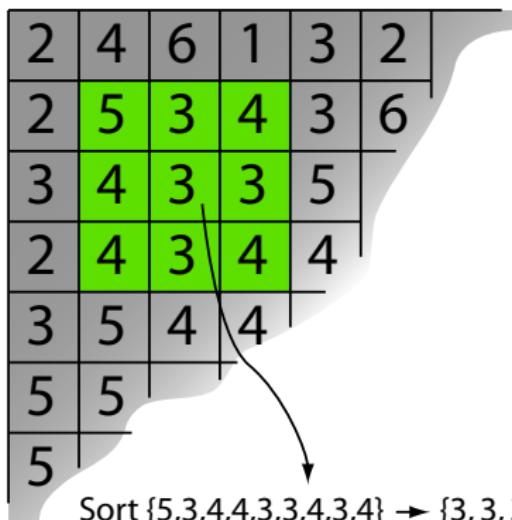
Low-Pass Filters with noise



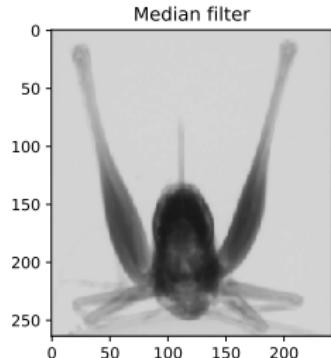
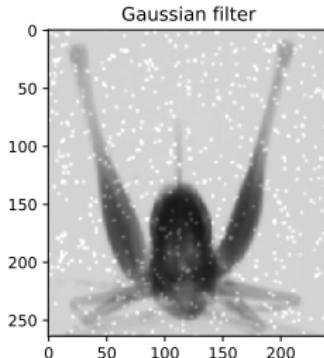
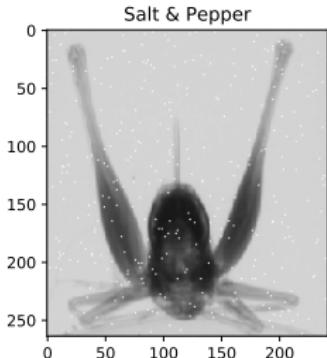
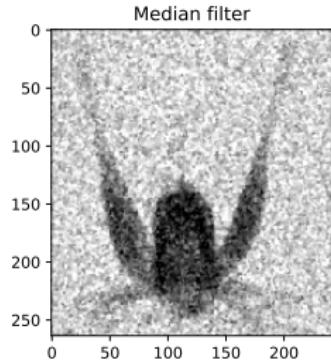
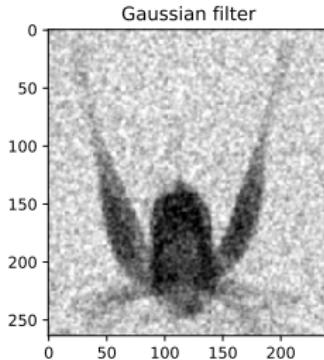
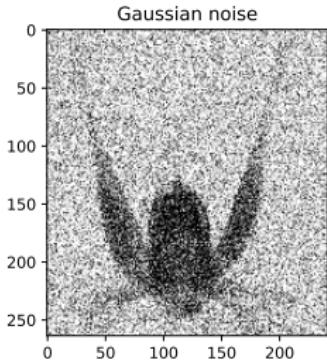
Non-linear filters for denoising

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

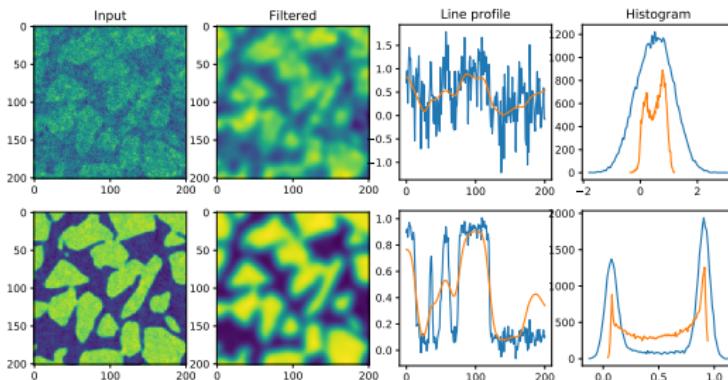


Compare median and box filters



In cases with low SNR and artifacts

- Filter kernel increase
- Increased blurring
- Relevant features vanish



Demo python notebook

New filters have been introduced to handle 'ugly' images

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

The idea

Smoothing normally consider information from the neighborhood like

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

Non-local smoothing average similar 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)

The non-local means filter is defined as

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

where

v and u input and result images.

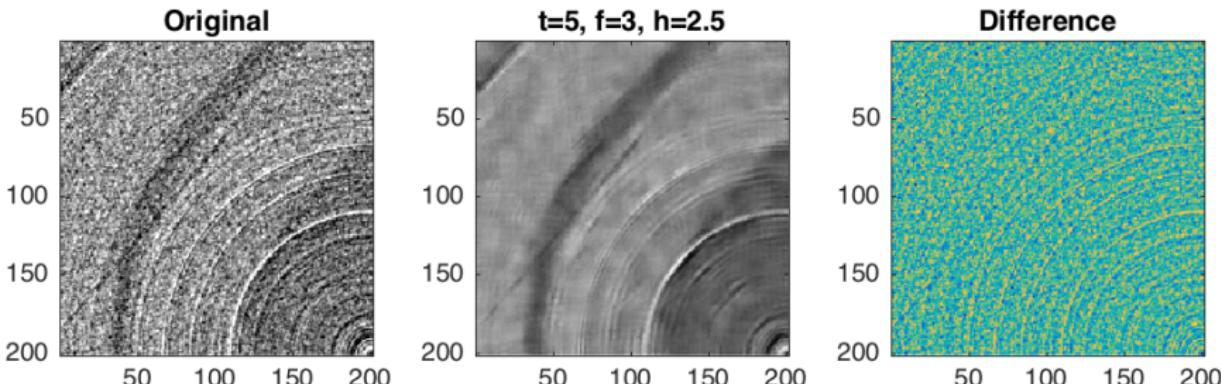
$C(p)$ is the sum of all pixel weights as

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

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

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

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



Observations

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

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.

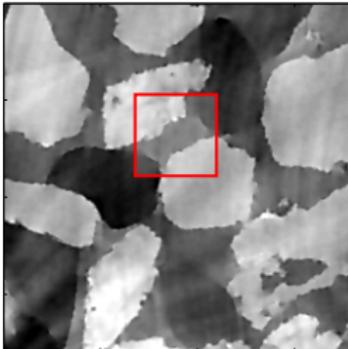
τ Time increment.

Comparing different filters

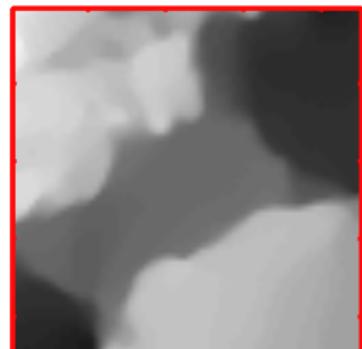
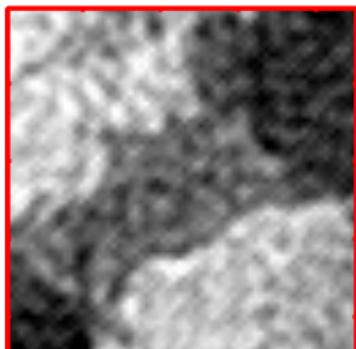
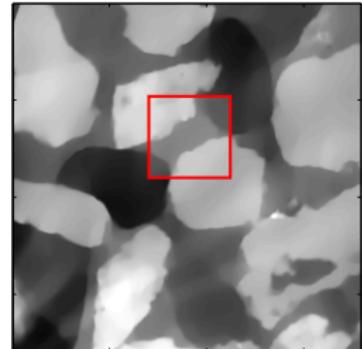
Original



Diffusion filter

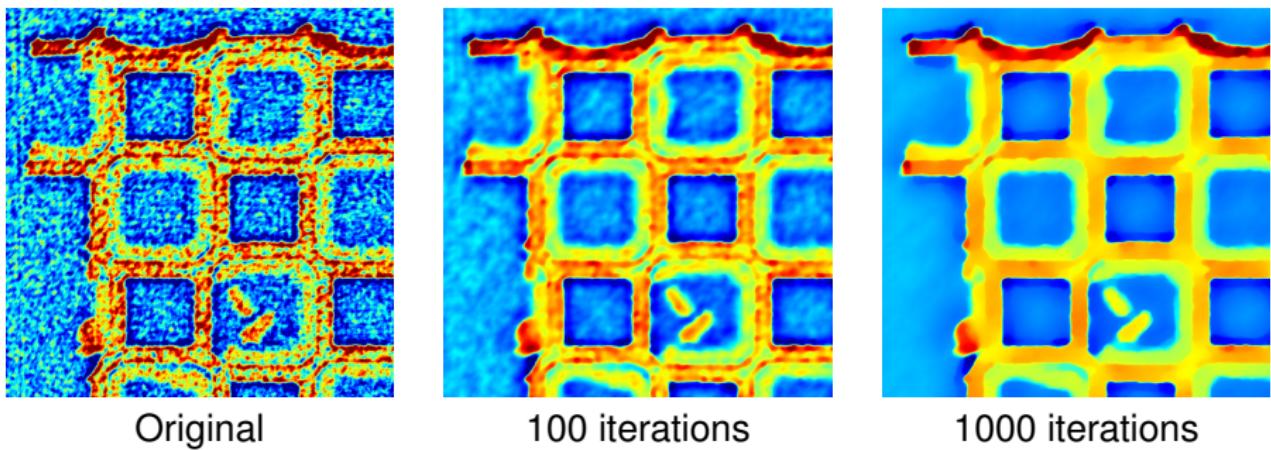


ISS filter



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

Neutron CT of a diesel particulate filter

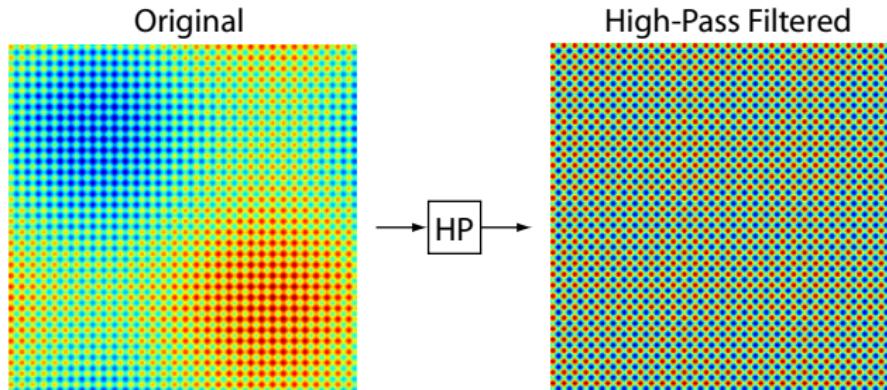


Grünzweig *et al.*, MTZ, 2012, ?
Kaestner *et al.*, WCNDT-18, 2012, Kaestner et al. (2012)

High-pass filters and edge detection

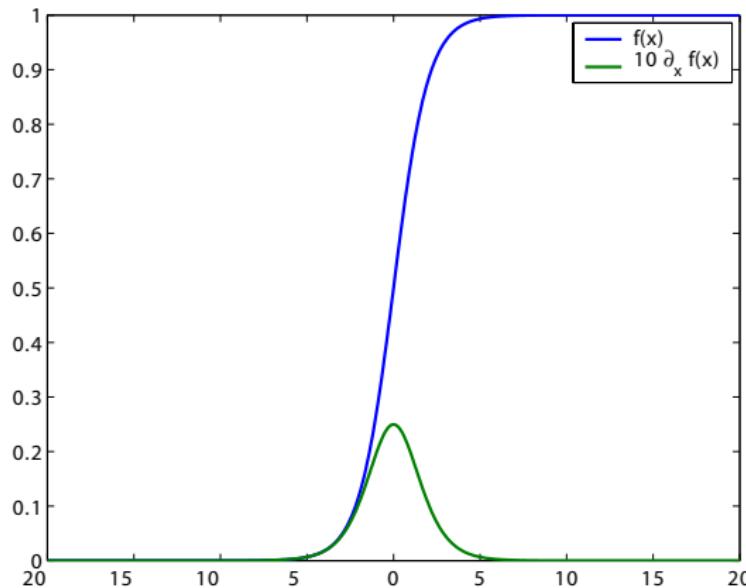
The effects of a *High-Pass* filter are in general

- rapid changes are enhanced
- slow changes are suppressed



Properties of the gradient:

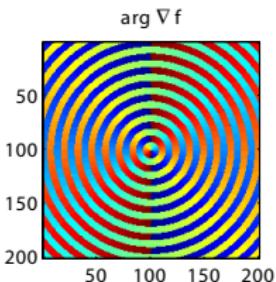
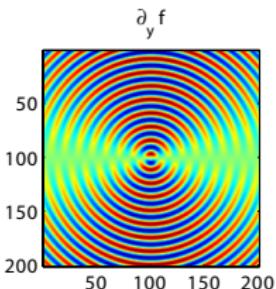
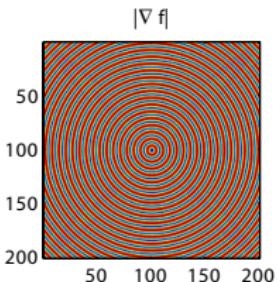
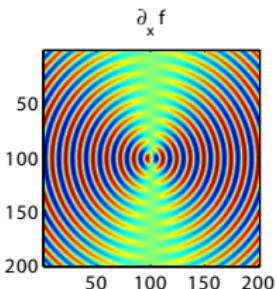
- Large response for step changes
- No response for constant levels



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

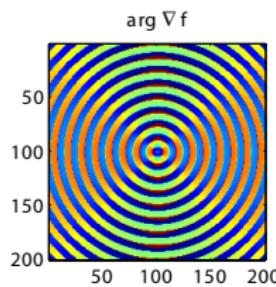
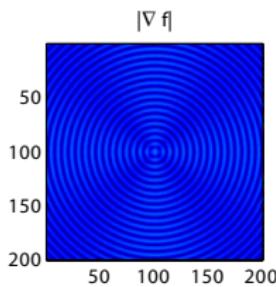
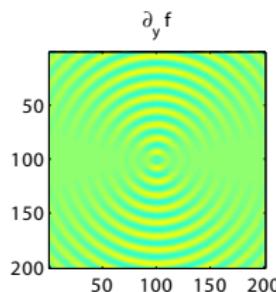
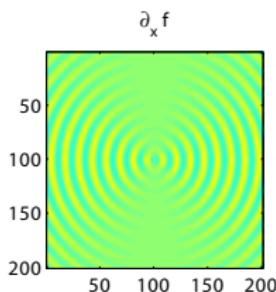
$$D_x = D_y^T = \frac{1}{2} \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

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



Jähne (2002)

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



Gradient of a Real Image

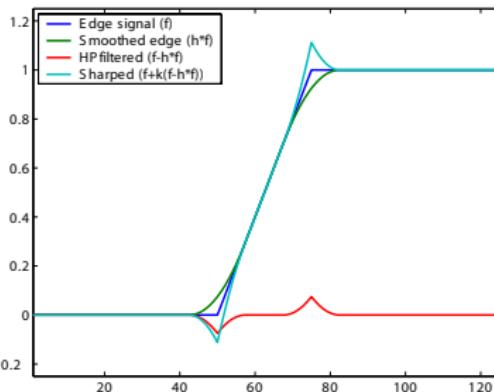
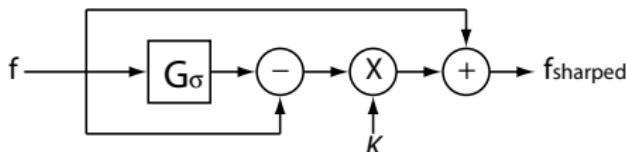
Original

 ∂_y  ∂_x 

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

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

Block diagram



Sharpening Images – 2D Example

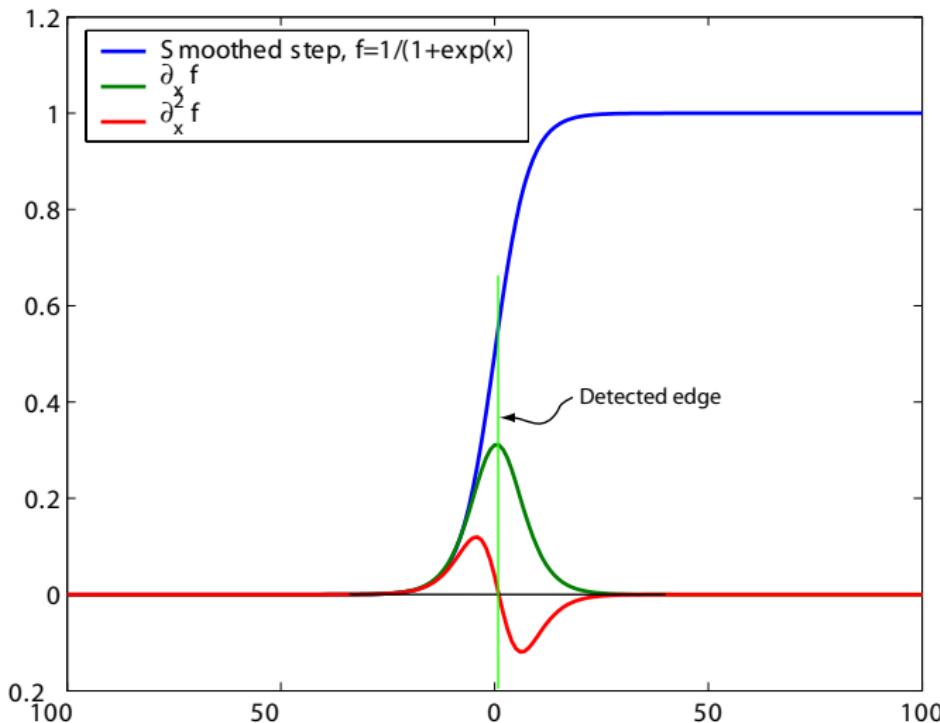
 $\sigma=1.5, K=15$ 

Original

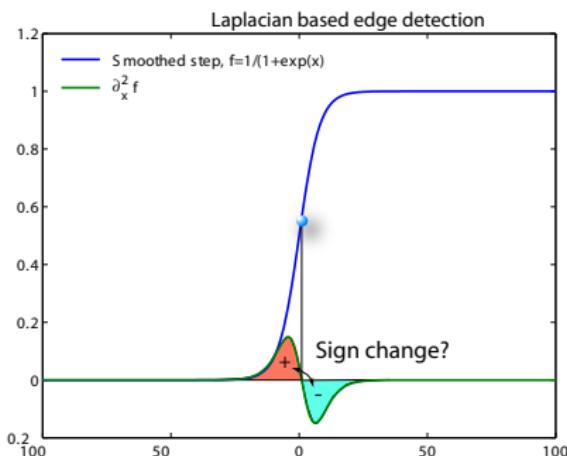
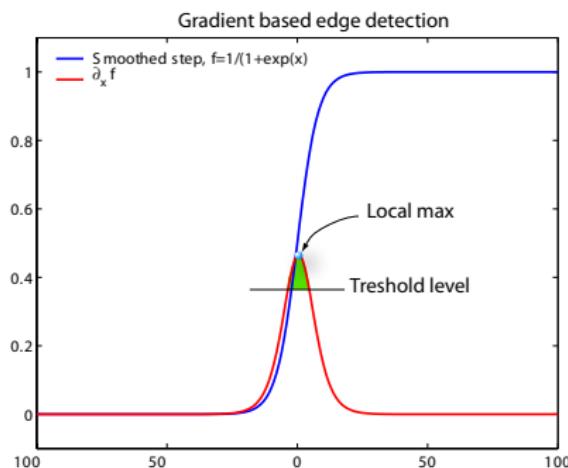
 $\sigma=1.5, K=10$  $\sigma=10, K=15$  $\sigma=10, K=10$ 

The interesting features of an edge are.

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



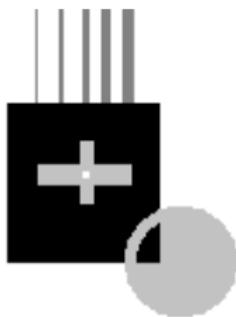
Edges can be detected with first and second derivatives



A simple Laplacian-based edge detector:

- Compute $g = L_x * f + L_y * f$, with $L_x = L_y^T = [1 \ -2 \ 1]$
- Assign $g \stackrel{\text{Background}}{\underset{\text{Edge}}{\gtrless}} 0$

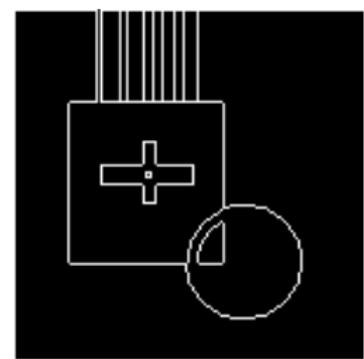
Original



$\nabla^2 f$



$\nabla^2 f < 0$



Problem

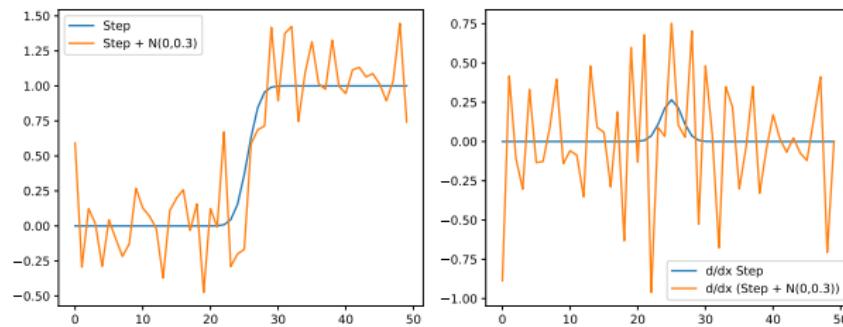
The large problem for an edge detector is noise, it can

- 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 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

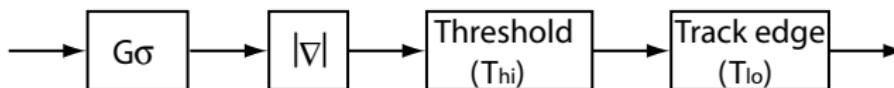
-1		1
-1		1
-1		1

See e.g. Haralik and Shapiro (1992)

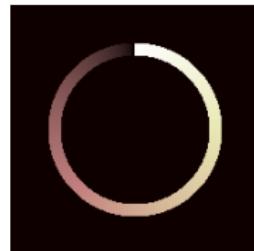
The edge detection performance can be improved by combining

- Smoothing
- Thresholding
- Tracking

Block diagram



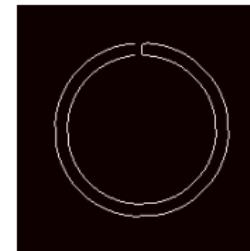
Example



Original



Sobel



Canny

Canny (1986)

Filters in the frequency domain

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

$$af(x) + bg(x) \Leftrightarrow aF(\omega) + bG(\omega)$$

- Convolution

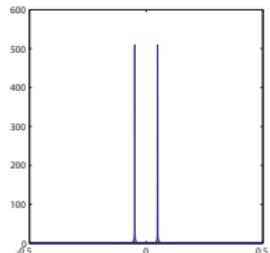
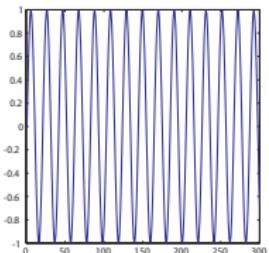
$$f * g(t) \Leftrightarrow F(\omega) \cdot G(\omega)$$

- Symmetry

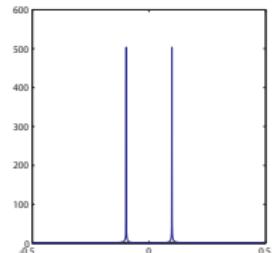
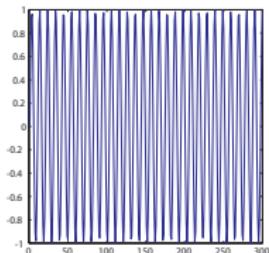
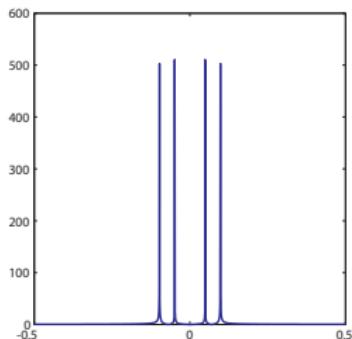
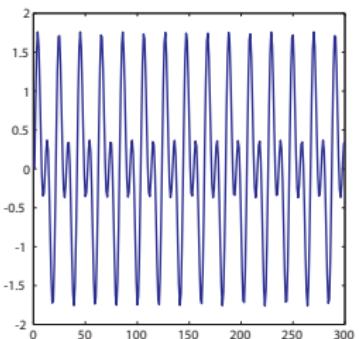
$$x \in \mathbf{R} \Rightarrow F(\omega) = F(-\omega)$$

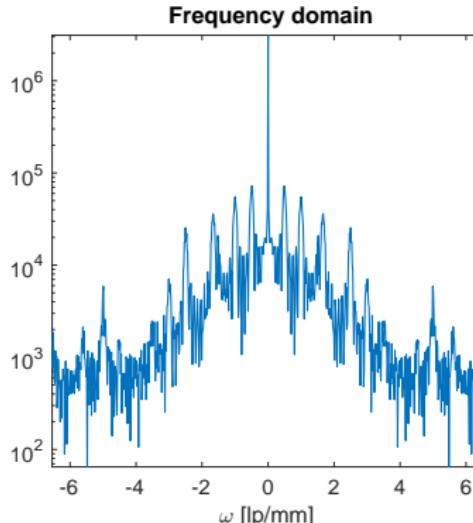
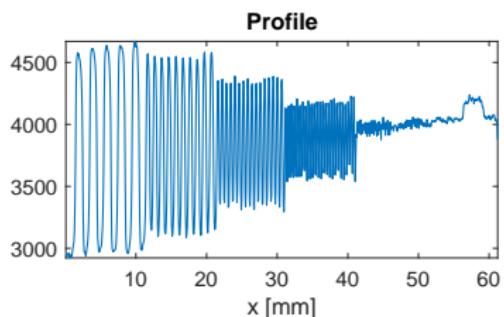
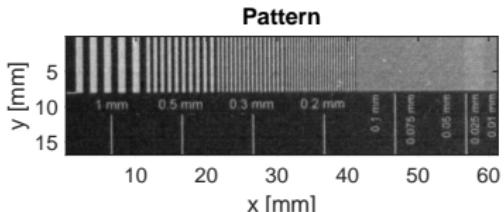
General Jähne Jähne (2002) or Detailed Granlund Granlund and Knutsson (1995)

Addition of 1D signals

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

+

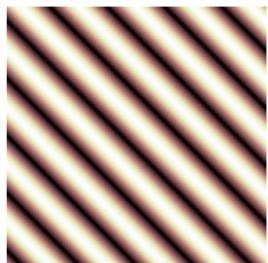
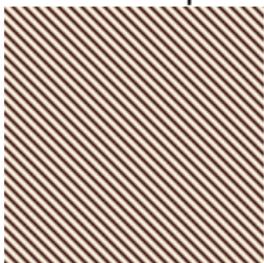
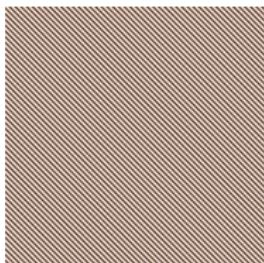
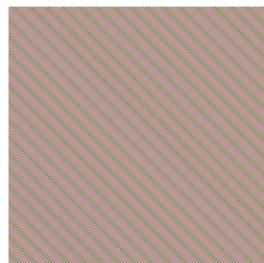
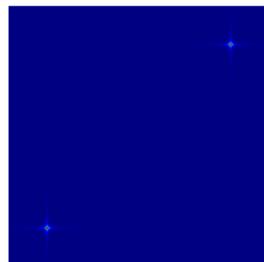
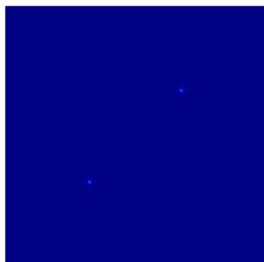
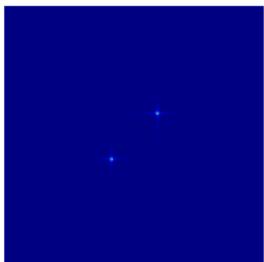
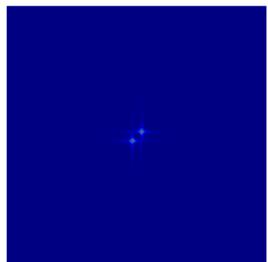
 $g(t)$ and $\mathcal{F}\{g\}$  $f(t) + g(t)$ and $\mathcal{F}\{f\} + \mathcal{F}\{g\}$ 



What we can see:

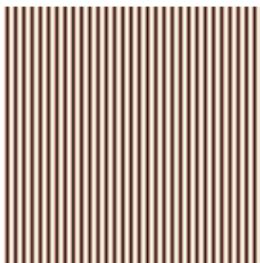
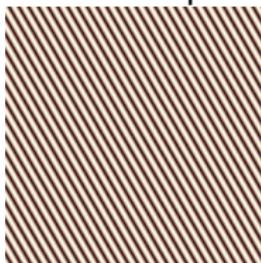
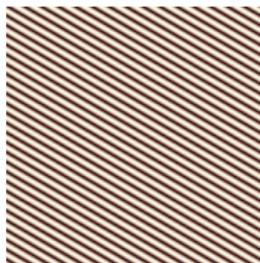
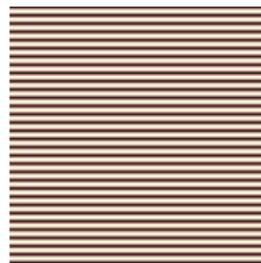
- Peaks for each line frequency
- Smoothing window
- Noise level

Spatial domain

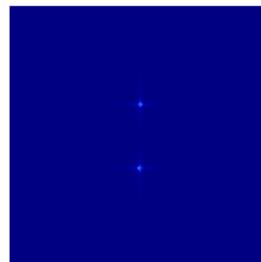
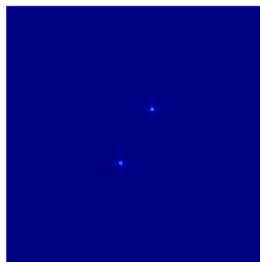
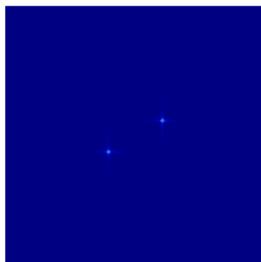
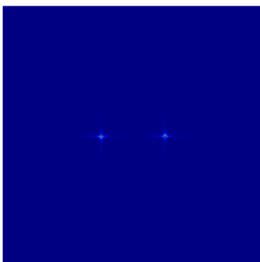
 $\omega = 0.05$  $\omega = 0.25$  $\omega = 0.5$  $\omega = 1.0$ 

Frequency domain

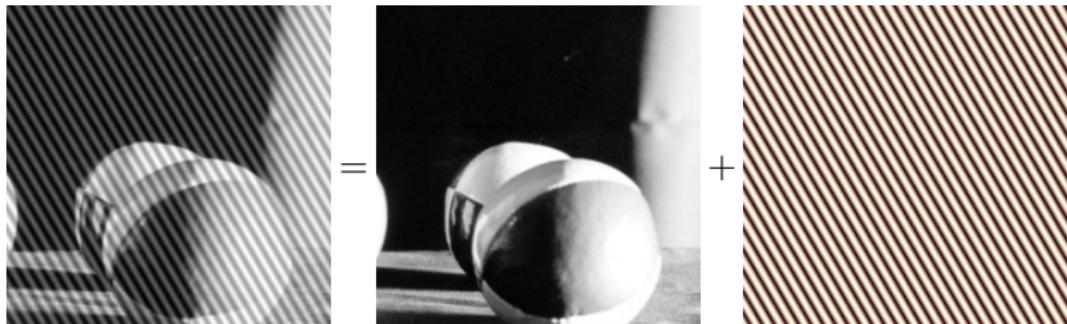
Spatial domain

 $\theta = 0$  $\theta = 30^\circ$  $\theta = 60^\circ$  $\theta = 90^\circ$

Frequency domain



You have



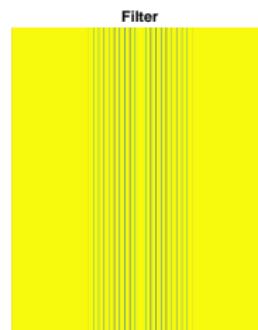
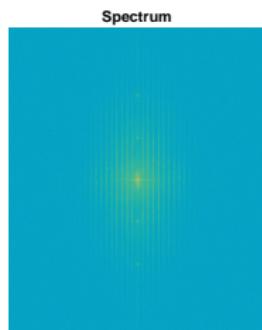
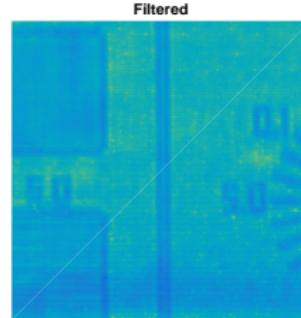
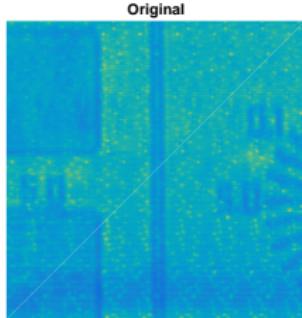
Filter with a band stop filter defined as

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

The image is processed by

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

Filter cyclic patterns



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

- 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.