Laplacian filter

*Commonly used for sharpening.

•Based on the 2nd derivative

•*Isotropical*, i.e. yields the same result independent of the rotation of the image (rotation invariance)

•The Laplace operator:

$$\nabla^2 f(x,y) = \frac{\partial^2}{\partial x^2} f(x,y) + \frac{\partial^2}{\partial y^2} f(x,y)$$

Laplacian filter

- Discrete Laplace:

$$\nabla^2 f(i,j) = f(i,j-1) + f(i-1,j) + f(i+1,j) + f(i,j+1) - 4f(i,j)$$

Laplace-kernels

0	1	0
1	−4	1
0	1	0

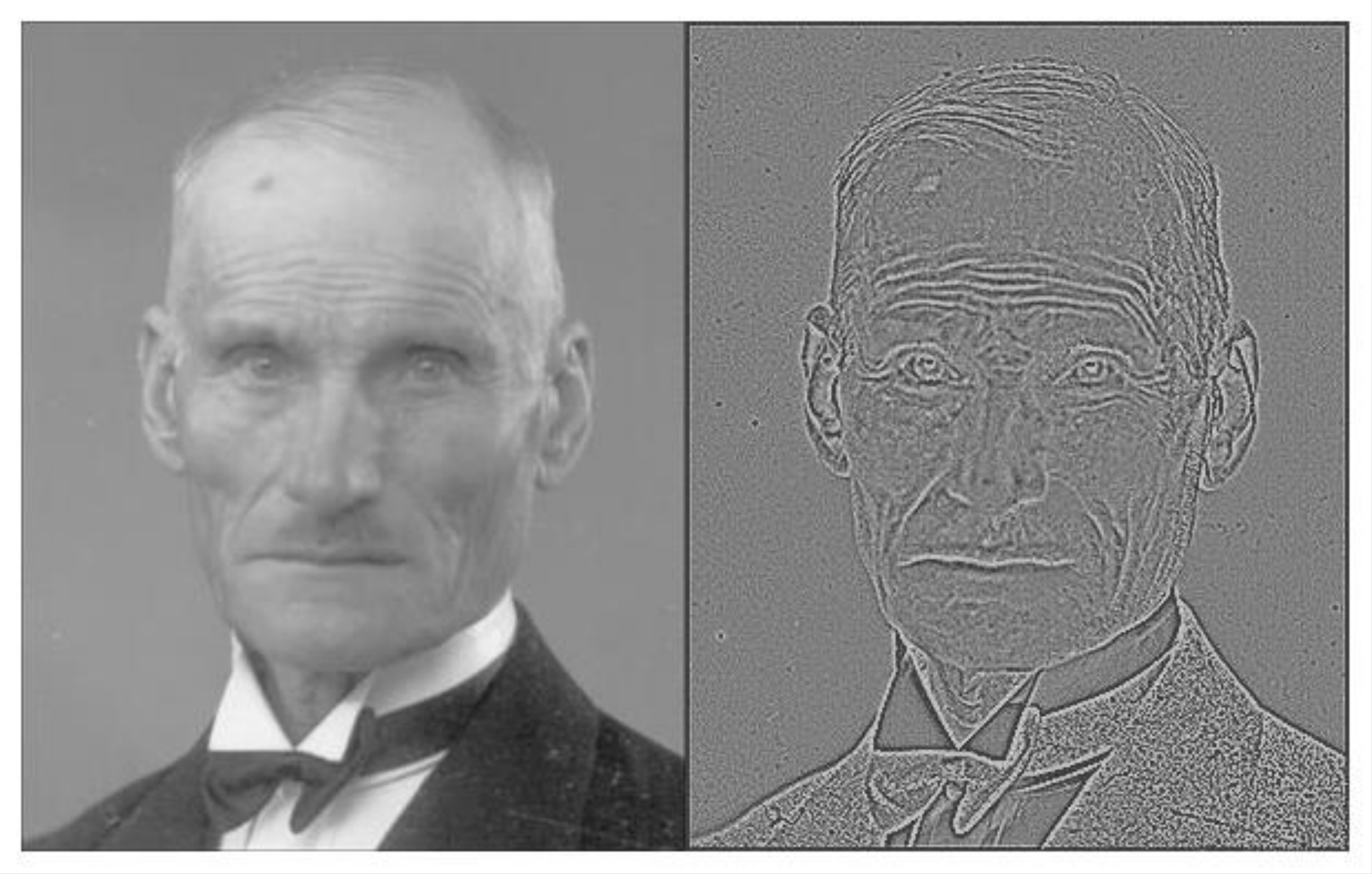
1	1	1
1	−8	1
1	1	1

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

Laplacian filter

Example:



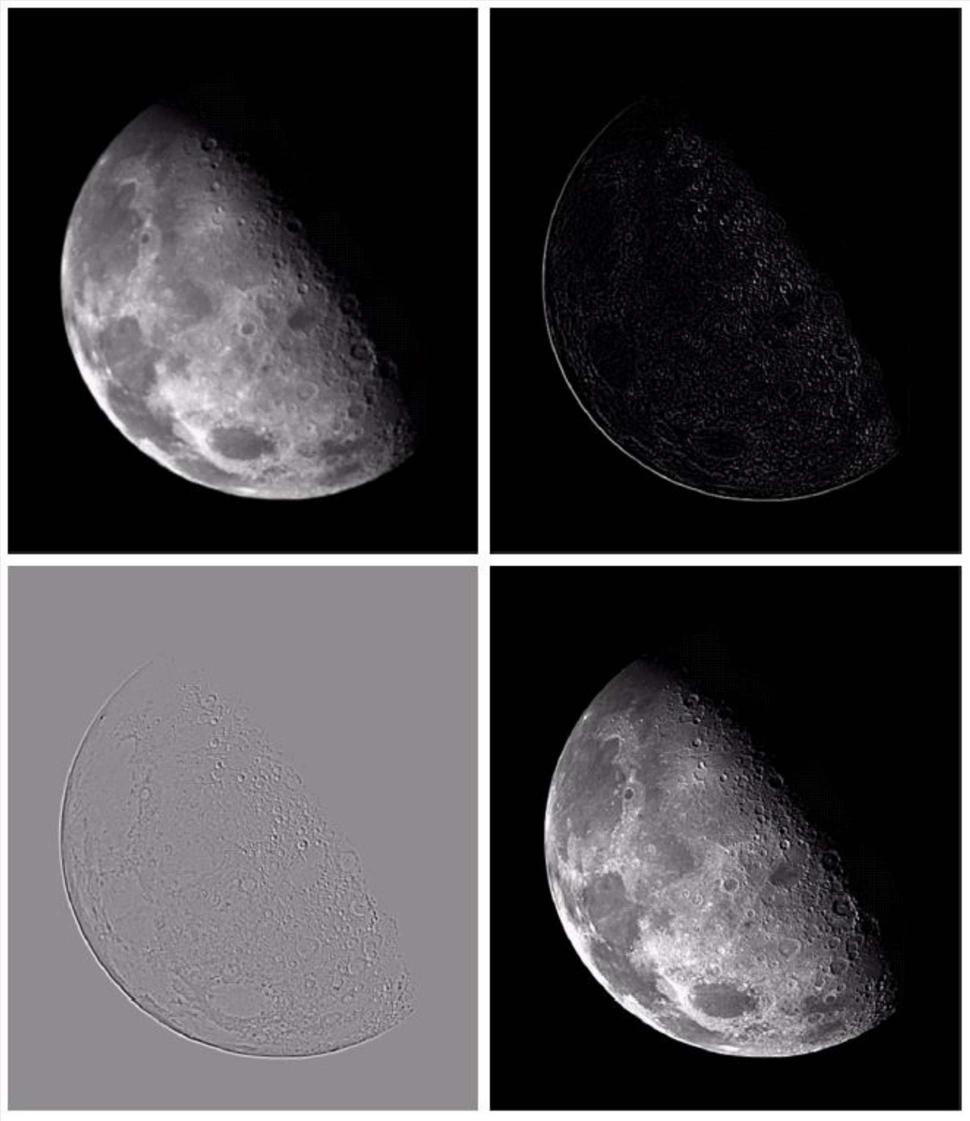
The Laplace image has an offset; 0-level is set to 0.5

Laplacian filter

Sharpening results if the Laplace image is added to the original:

$$g(x,y) =$$

$$= \begin{cases} f(x,y) - \nabla^2 f(x,y) & \text{if } centerkoeff. < 0 \\ f(x,y) + \nabla^2 f(x,y) & \text{if } centerkoeff. > 0 \end{cases}$$



Laplacian filter

The sharpening

$$g(x,y) = \begin{cases} f(x,y) - \nabla^2 f(x,y) & \text{if } \mathit{centerkoeff.} < 0 \\ f(x,y) + \nabla^2 f(x,y) & \text{if } \mathit{centerkoeff.} > 0 \end{cases}$$

can be implemented by <u>one</u> filter kernel.
If Laplace is implemented by

0	1	0
1	−4	1
0	1	0

the total sharpening is implemented by

0	−1	0
−1	5	−1
0	−1	0

Derivating filters

- The 1st derivative is used in *gradient filters*.
- The gradient: $\nabla \mathbf{f} = (\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}) = (g_x, g_y)$

The filter responses g_x och g_y result from these kernels:

−1	−2	−1
0	0	0
1	2	1

=> g_y

−1	0	1
−2	0	2
−1	0	1

=> g_x

Sobel-filter

Derivating filters

- Another common gradient filter : Prewitts

−1	−1	−1
0	0	0
1	1	1

−1	0	1
−1	0	1
−1	0	1

Prewitt

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

Sobel

Derivating filters

- The filter responses from the Sobel-filters can be used as follows:

$$g = \sqrt{g_x^2 + g_y^2}$$ Gradient magnitude

$$\varphi = \tan^{-1}(g_y / g_x)$$ Gradient direction

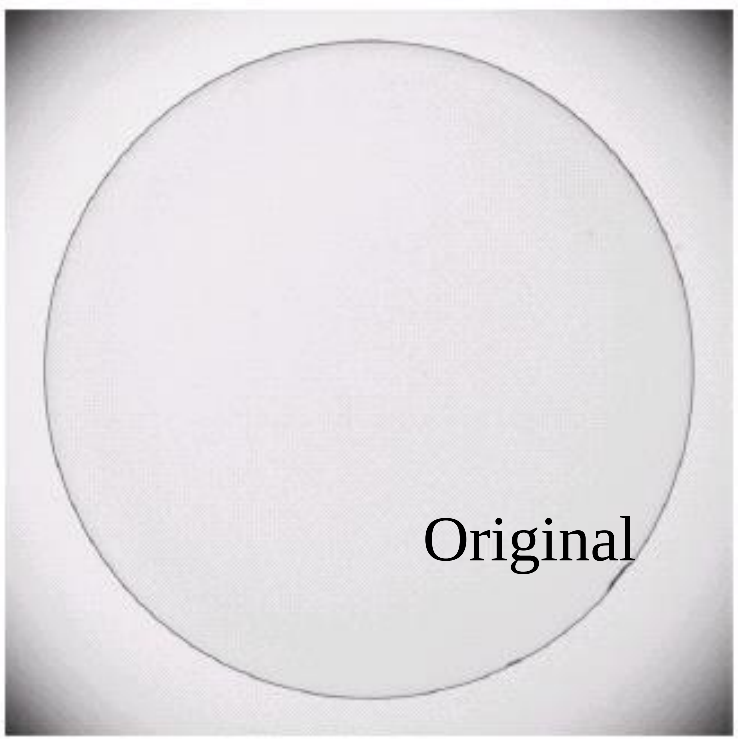
Laplacian and Median

..\Matlab\html\MatLabFilterDemo.html

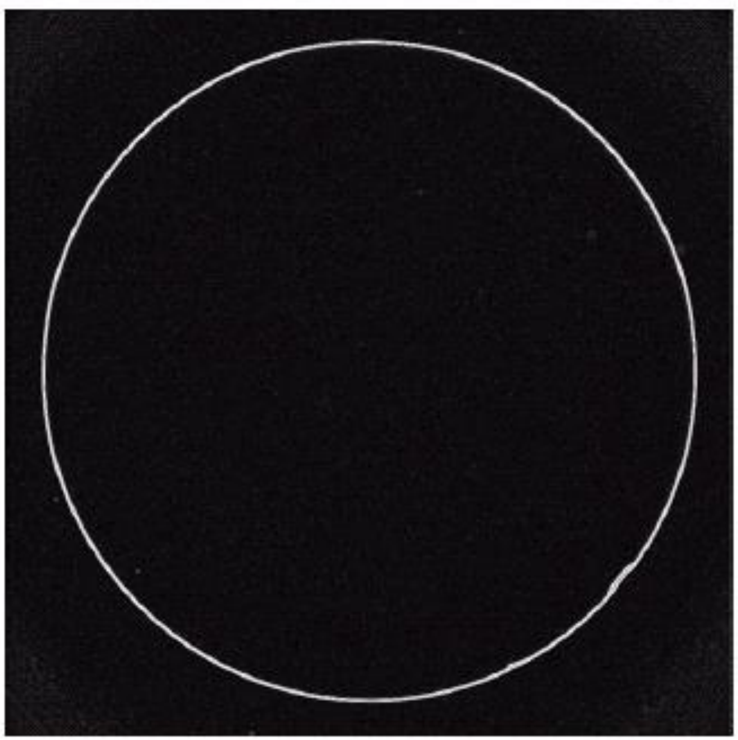
Derivative Filters

Example:

Gradient magnitude

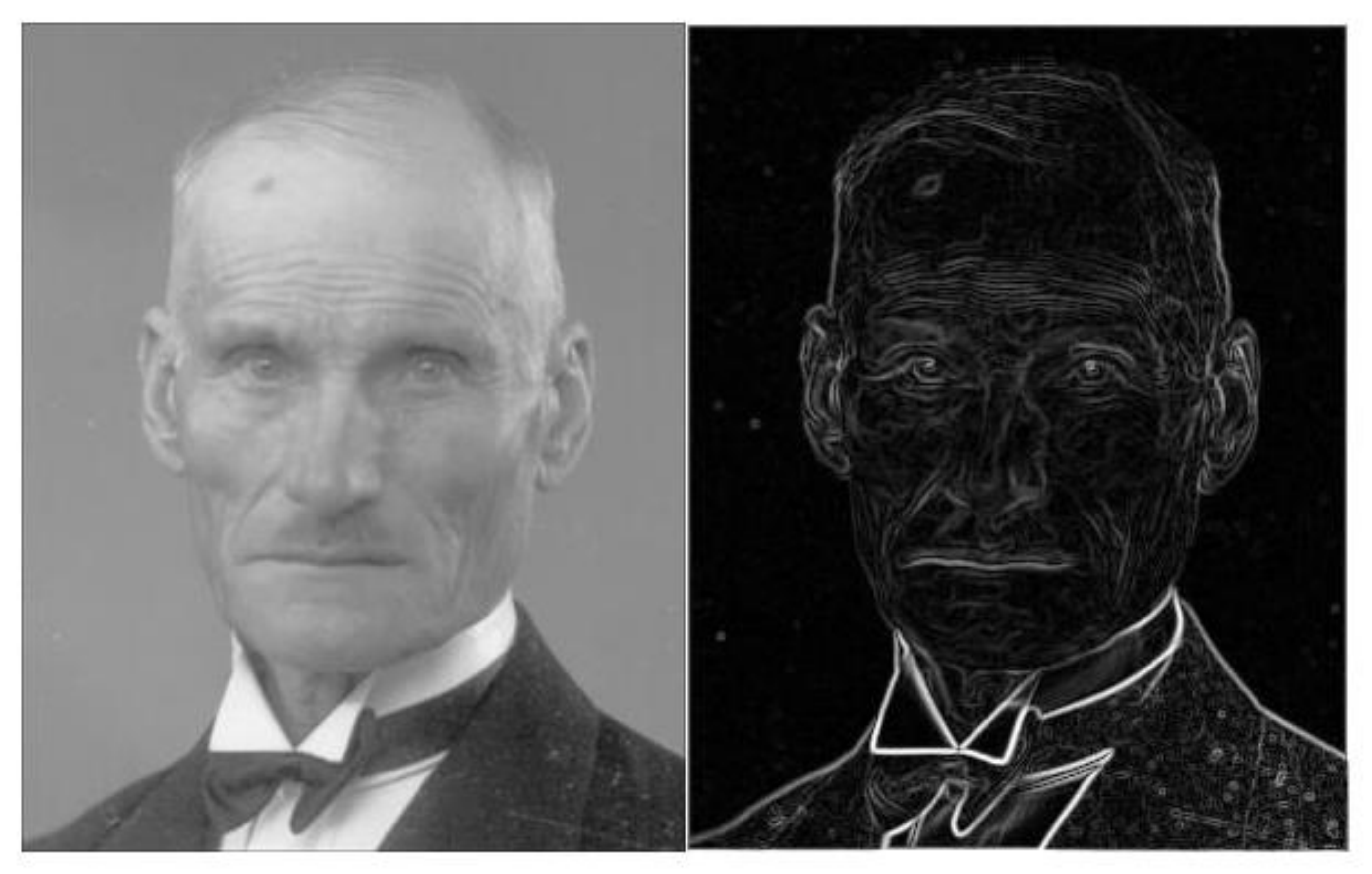


Original



Derivating filters

Example:

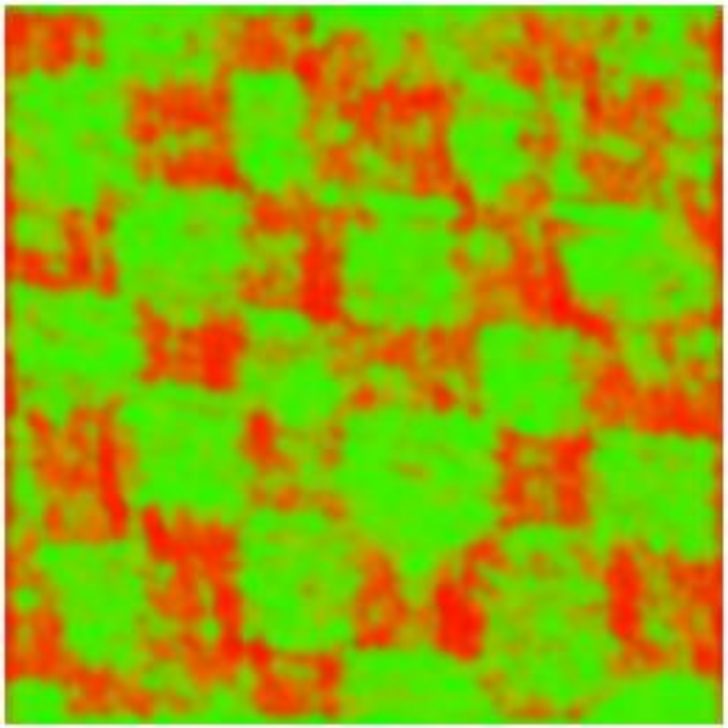


Original Gradient magnitude

Derivating filters

Using the gradient direction to find main orientation

Original



Orientation

Adaptive filters

Regular filters work the same way over the entire image regardless of the image content. *Adaptive filters* change their behaviour based on image content.

Example:

Edge preserving smoothing with an averaging kernel. Adaptivity: only pixels with values close to the value of the centerpixel contribute to the average.

Averaging and Median Filtering

Consider the noisy image in Figure 3.18a. In order to remove most of the noise, the Gaussian filter is forced to smooth away high-frequency detail, which is most noticeable near strong edges. Median filtering does better but, as mentioned before, does not do as good a job at smoothing away from discontinuities.

(a)

(b)

(c)

Bilateral Filters

What if instead of rejecting a fixed percentage , we simply reject (in a soft way) pixels whose values differ too much from the central pixel value? This is the essential idea in bilateral filtering,

In the bilateral filter, the output pixel value depends on a weighted combination of neighboring pixel values

$$g(i,j) = \frac{\sum_{k,l} f(k,l) w(i,j,k,l)}{\sum_{k,l} w(i,j,k,l)}. \tag{3.34}$$

The weighting coefficient $w(i,j,k,l)$ depends on the product of a *domain kernel* (Figure 3.19c),

$$d(i,j,k,l) = \exp\left(-\frac{(i-k)^2 + (j-l)^2}{2\sigma_d^2}\right), \tag{3.35}$$

and a data-dependent *range kernel* (Figure 3.19d),

$$r(i,j,k,l) = \exp\left(-\frac{\|f(i,j) - f(k,l)\|^2}{2\sigma_r^2}\right). \tag{3.36}$$

When multiplied together, these yield the data-dependent *bilateral weight function*

$$w(i,j,k,l) = \exp\left(-\frac{(i-k)^2 + (j-l)^2}{2\sigma_d^2} - \frac{\|f(i,j) - f(k,l)\|^2}{2\sigma_r^2}\right). \tag{3.37}$$

Bilateral Kernels

1	2	1	2	4
2	1	3	5	8
1	3	7	6	9
3	4	8	6	7
4	5	7	8	9

(a) median = 4

1	2	1	2	4
2	1	3	5	8
1	3	7	6	9
3	4	8	6	7
4	5	7	8	9

(b) α-mean= 4.6

	2	1	0	1	2
2	0.1	0.3	0.4	0.3	0.1
1	0.3	0.6	0.8	0.6	0.3
0	0.4	0.8	1.0	0.8	0.4
1	0.3	0.6	0.8	0.6	0.3
2	0.1	0.3	0.4	0.3	0.1

(c) domain filter

0.0	0.0	0.0	0.0	0.2
0.0	0.0	0.0	0.4	0.8
0.0	0.0	1.0	0.8	0.4
0.0	0.2	0.8	0.8	1.0
0.2	0.4	1.0	0.8	0.4

(d) range filter

Figure 3.19 Median and bilateral filtering: (a) median pixel (green); (b) selected α-trimmed mean pixels; (c) domain filter (numbers along edge are pixel distances); (d) range filter.

Bilateral Filter @ Edge

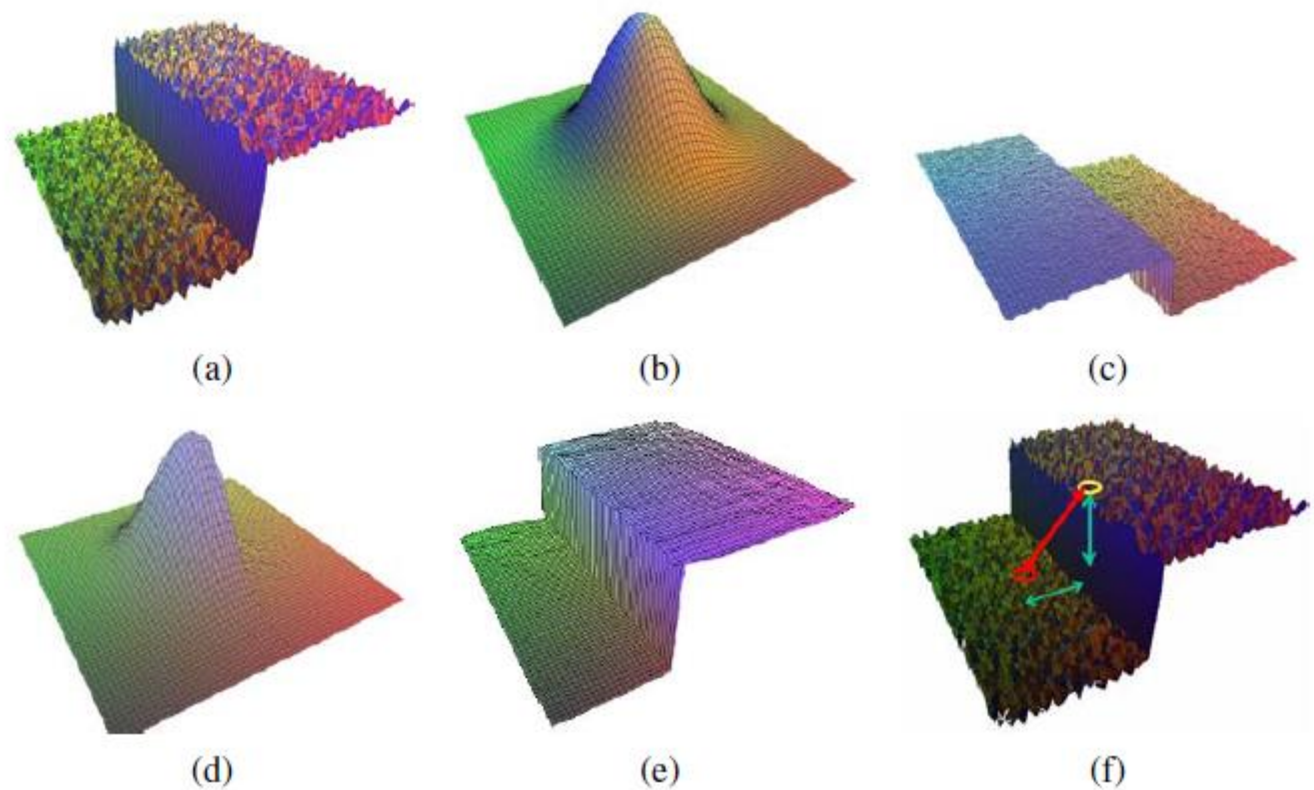


Figure 3.20 Bilateral filtering (Durand and Dorsey 2002) © 2002 ACM: (a) noisy step edge input; (b) domain filter (Gaussian); (c) range filter (similarity to center pixel value); (d) bilateral filter; (e) filtered step edge output; (f) 3D distance between pixels.

Summary

1) Know how linear filters work, what a filter kernel is
2) Do filtering with paper and pen
3) Know what a convolution and a correlation is
4) Different types of 1-dimensional changes, edges, lines, …
5) Differential filters
6) Know most popular filters: Prewitt, Sobel, Laplacian
7) Laplacian and sharpening
8) Non-linear filters: median
9) Similarity/difference/application of median and mean
10) Edge magnitude and direction
11) Basic idea behind bilateral filtering

Fourier Filtering

Reiner Lenz

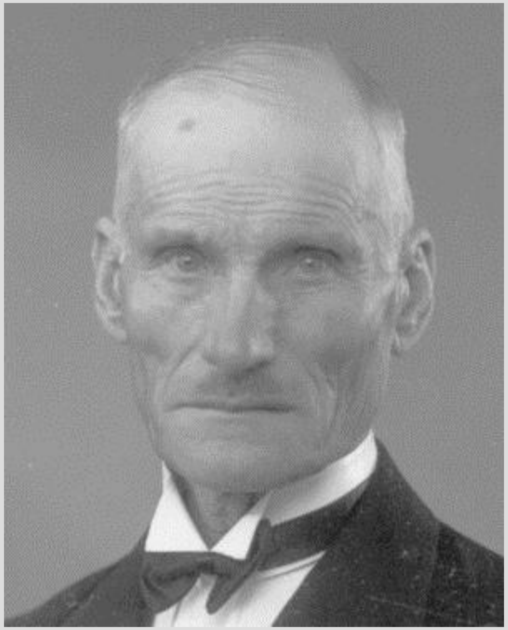
2015

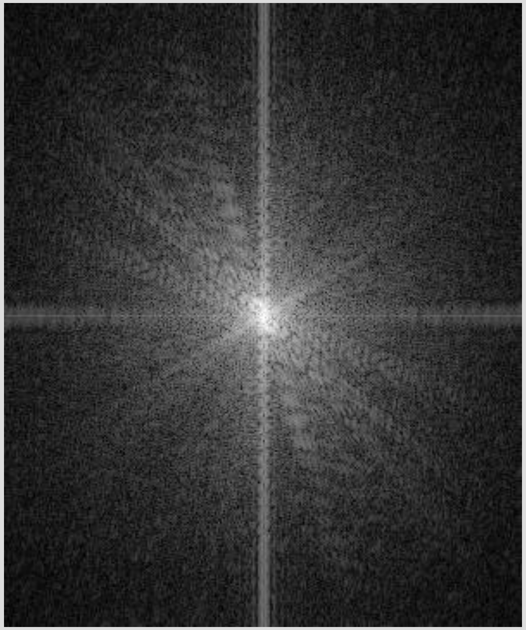
Filtering in the frequency domain

f(x,y)

$$\Im\left[f(x,y)\right]$$

$|F(u,v)|$





$$\Im^{-1}\left[F(u,v)\right]$$

Spatial domain

Frequency domain

$\Im$: Fourier transform

The Fourier transform

$$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y) e^{-j2\pi(ux+vy)} dxdy$$

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y) e^{-j2\pi\left(ux/M + vy/N\right)}$$

(DFT)

$$f(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v) e^{j2\pi\left(ux/M + vy/N\right)}$$

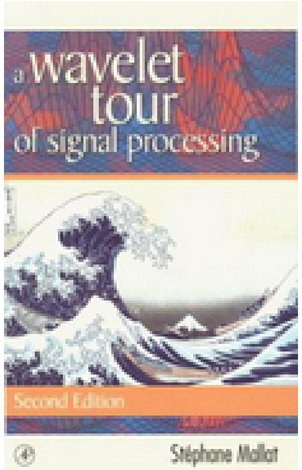
(Inverse DFT)

Recommended Reading

Översikt

☐ Spara

☐ Lägg till i min lista



Mer av samma författare

Exemplarinformation	Mer information Detaljer
Författare	**Mallat, S. G. (Stéphane G.)**
Titel	A wavelet tour of signal processing [Elektronisk resurs]
Utgivare:	Academic Press,
Utgivningsdatum:	1999.
Sidor:	1 online resource (xxiv, 637 p.) :
ISBN:	9780124666061

▸ LÄNK

INTERNET

Hylla	**Material**		**Här finns hyllan**
INTERNET	E-böcker och e-examensarbeten	Fulltext	

The Fourier transform

F(u,v) is a complex number: F(u,v)=R(u,v)+iI(u,v)

The Fourier spectrum: |F(u,v)|

Phase angle: $\Phi(u,v) = \tan^{-1}\left[I(u,v) \middle/ R(u,v) \right]$

Power spectrum: $|F(u,v)|^2$

f(x,y) real (which all our images are) => $F(u,v)=F^*(-u,-v)$
=> |F(u,v)|=|F(-u,-v)| (the spectrum is symmetric)

Linear – Shift-Invariant Systems

Consider a system, black box, operator T

T maps input signals s(t) to output signals o(t) = T(s(t))
in the simplest case the signals are functions of the time variable t like sound waves

The system T is linear if

$T(as_1(t) + bs_2(t))$ = a $T(s_1(t))$+ b$T(s_2(t))$

Consider a shifted new coordinate system (select a new starting point for your signal)
In the new coordinate system the signal s(t) becomes s(t-τ)

For most systems T the output T(s(t-τ)), generated by s(t-τ),
will have essentially the same form is the output T(s(t)), generated by s(t)

For the exponential function we have: $e^{i(t-\tau)} = e^{-i\tau}e^{it}$

The original signal e^{it} is multiplied by the factor $e^{-i\tau}$
and the effect of the coordinate shift is isolated from the form of the signal

General Case

Develop a signal in a Fourier series

since the system is linear it is sufficient to consider the Fourier terms e^{int}

Since the system preserves signals of the form e^{int}
(they are multiplied by factors $e^{in\tau}$)

We can track the changes of a shift of coordinate system
by considering each Fourier terms e^{int} independent of the others

Typical shift operations

Time dependent systems:

Start an experiment at different points in time

Use different clocks

Space dependent systems:

Do an experiment at different locations

Move an object in front of a camera

2D rotations

Change the orientation of a 2D coordinate system

Choose different (shifted) angles for hue color descriptors

Systems of a different type:

Most important are 3D rotations/orientations which are more complicated than the 2D case since for 3D rotations you can, in general, not interchange the order in which they are applied