Outline

Fourier Transform

Convolution

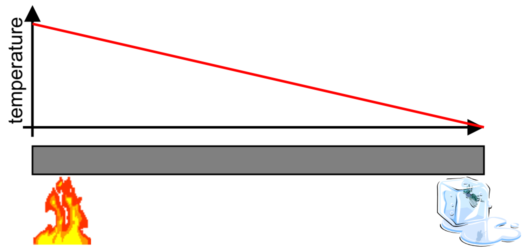
Image Restoration: Linear Filtering

Diffusion Processes for Noise Filtering

- *linear scale space theory*
- Gauss-Laplace pyramid for image representation
- nonlinear diffusion in the Malik Perona sense

Linear Diffusion and Image Processing

Heat equation and diffusion processes: We observe experimentally that *heat diffuses* from the heated part of a metal beam to the cooled end. The temperature decay is linear under stationary conditions.



Likewise, *chemicals diffuse* from regions of hight concentration to regions with low concentrations.

Idea: Utilize the physical process of *diffusion* to smooth noisy images. *Image intensities* follow a diffusive dynamics which terminates with a homogeneous image of average intensity.

Mathematics of Diffusion: concentration differences $\nabla_{\mathbf{x}} f(\mathbf{x}, t)$ of a quantity $f(\mathbf{x}, t)$ (here pixel intensities) cause a flux

$$\begin{aligned} j \; : \; \mathbb{R}^2 \times \mathbb{R} &\rightarrow \mathbb{R}^2, \\ (\mathbf{x}, t) &\mapsto (j_x(\mathbf{x}, t), j_y(\mathbf{x}, t)) =: j(\mathbf{x}, t) \end{aligned}$$

which transports $f(\mathbf{x}, t)$ from high concentration regions to low ones.

Fick's Law: The flux $j(\mathbf{x})$ is proportional to the concentration differences (linearity!) and $\mathbf{D}$ denotes the diffusion tensor:

$$j(\mathbf{x}) = -\mathbf{D} \nabla f(\mathbf{x}, t)$$

Continuity equation: Changes in f can only be achieved by transport, not by "destroying" f, i.e.,

$$\partial_t f = -\nabla \cdot j = -\text{div } j$$

Diffusion equation: Insert Fick's law into the continuity equation yields

$$\boxed{\partial_t f(\mathbf{x}, t) = \nabla \cdot \big(\mathbf{D} \nabla f(\mathbf{x}, t)\big)}$$

Variations of the diffusion process

- *homogeneous* diffusion: $\mathbf{D}$ is space independent.
- *inhomogeneous* diffusion: $\mathbf{D}$ is a function of $\mathbf{x}$, i.e., depends on the space.
- *isotropic* diffusion: $\nabla f || j$, i.e., the gradient is parallel to the flux.
- *nonlinear* diffusion: the diffusion tensor $\mathbf{D}$ depends on f.
- *scalar* diffusion: $\mathbf{D} = D \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

Solution of the Diffusion Equation

- consider a scalar diffusion process

$$\begin{aligned} \frac{\partial}{\partial t} f &= D \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \right) f = D \Delta f \\ f(x, y, 0) &:= f_0(x, y) \quad \text{boundary condition} \end{aligned}$$

Decomposition of the function in spatial Fourier components.

$$\begin{aligned} f(x, y, t) &= \int_\Omega \hat{f}(u, v, t) \exp(\imath 2\pi (ux + vy)) du dv \\ \Delta f(x, y, t) &= \int_\Omega \hat{f}(u, v, t) (2\pi\imath)^2 (u^2 + v^2) \exp(\imath 2\pi (ux + vy)) du dv \\ &= \mathcal{F}^{-1} \left[-4\pi^2 (u^2 + v^2) \hat{f}(u, v, t) \right] \end{aligned}$$

Fourier transformed diffusion equation

$$\frac{\partial}{\partial t}\hat{f}(u,v,t) = -4\pi^2 D(u^2+v^2)\hat{f}(u,v,t)$$

Integrate w.r.t. time: Ordinary differential equation in time.

$$\begin{aligned}
\frac{d\hat{f}(u,v,t)}{\hat{f}(u,v,t)} &= -4\pi^2 D(u^2+v^2)dt \\
\ln \hat{f}(u,v,t) &= -4\pi^2 D(u^2+v^2)t + \text{const} \\
\hat{f}(u,v,t) &= \hat{f}(u,v,0)\exp\Big(-4\pi^2 D(u^2+v^2)t\Big)
\end{aligned}$$

The constant has been identified as the function value at time $t = 0$.

Boundary constraints: let $f_0(x,y) = \delta(x)\delta(y)$ (δ-peak at the origin)

$$\hat{f}(u,v,0) = \int_\Omega \delta(x)\delta(y)\exp(-\imath 2\pi(ux+vy))dxdy = 1$$

Solution by inverse Fourier transformation:

$$\begin{aligned}
f(x,y,t) &= \int_\Omega \hat{f}(u,v,t)\exp(\imath 2\pi(ux+vy))du\,dv \\
&= \int_\Omega \exp\Big(-4\pi^2 D(u^2+v^2)t\Big)\exp(\imath 2\pi(ux+vy))du\,dv \\
&= \dots \quad \text{quadratic expansion, Gaussian integration} \\
&= \frac{1}{4\pi Dt}\exp\left(-\frac{x^2+y^2}{4Dt}\right) \\
&= \frac{1}{2\pi\sigma^2}\exp\left(-\frac{|\mathbf{x}|^2}{2\sigma^2}\right) \quad \text{for} \quad \sigma^2 = 2Dt
\end{aligned}$$

Time Evolution of Image Diffusion

Remark: the time evolution of a δ-function under diffusion, also called Green's function, is described by a Gaussian with variance proportional to $2Dt$.

Time evolution of an image: decompose the original image

$$f(x,y,0) = \int_\Omega f(\alpha,\beta,0)\delta(x-\alpha)\delta(y-\beta)d\alpha\,d\beta$$

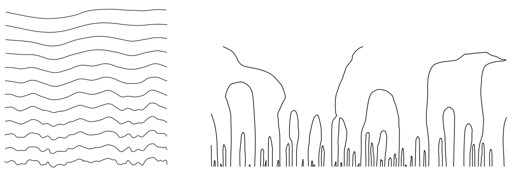
Linearity: since the diffusion equation is linear we can superpose time evolutions of δ-functions at positions $(x-\alpha, y-\beta)$.

Maximum-Minimum principle: $\inf_{\mathbb{R}^2} f(x,y,0) \le f \le \sup_{\mathbb{R}^2} f(x,y,0)$

Note: the linearity guarantees that the diffused image can be written as a convolution of the image with the time evolution of δ-functions.

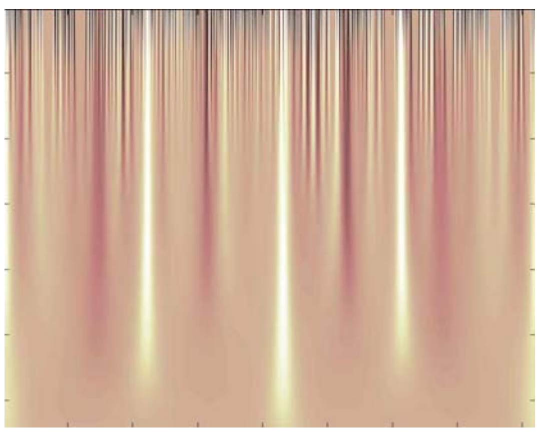
$$\begin{aligned}
f(x,y,t) &= \int_\Omega \frac{f(\alpha,\beta,0)}{4\pi Dt}\exp\left(-\frac{(x-\alpha)^2+(y-\beta)^2}{4Dt}\right)d\alpha\,d\beta \\
&= (G_{\sqrt{2Dt}} * f)(x,y,0)
\end{aligned}$$

where G_σ is a Gaussian with standard deviation σ.

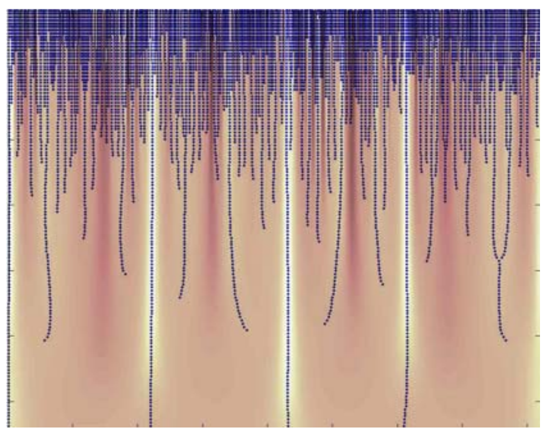


A.P. Witkin, *Scale Space Filtering*, in Proc. IJCAI 1983, p1019-1022

The Scale Space



Edge Tracking in Scale Space



Gaussian Smoothing of an Image

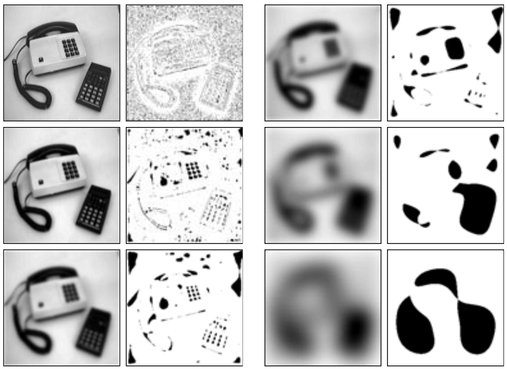


Figure 3: Different levels in the scale-space representation of a two-dimensional image at scale levels $t = 0, 2, 8, 32, 128$ and 512 together with grey-level blobs indicating local minima at each scale.

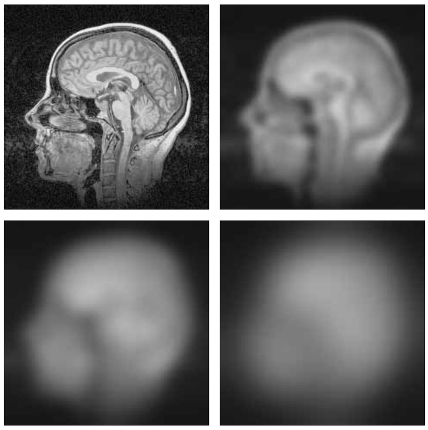


Figure 1: Scale-space behaviour of linear diffusion filtering. (a) Top Left: Original image, $\Omega = (0, 236)^2$. (b) Top Right: $t = 12.5$. (c) Bottom Left: $t = 50$. (d) Bottom Right: $t = 200$. Author: Joachim Weickert

Finite Difference Approximation

Diffusion equation: $\partial_t f = \partial_{xx} f + \partial_{yy} f$

Discretization: grid with size h_1, h_2, time step size τ (in image processing: often $h_1 = h_2 = 1$)

$$\begin{aligned} x_i &:= (i - \frac{1}{2})h_1 \\ y_j &:= (j - \frac{1}{2})h_2 \\ t_k &:= k\tau \\ f_{ij}^k &: \text{ approximates } f(x_i, y_j, t_k) \end{aligned}$$

Finite difference approximation in (x_i, y_j, t_k):

$$\begin{aligned} \frac{\partial}{\partial t} f &= \frac{f_{ij}^{k+1} - f_{ij}^k}{\tau} + \mathcal{O}(\tau) \\ \frac{\partial^2}{\partial x^2} f &= \frac{f_{i+1,j}^k - 2f_{ij}^k + f_{i-1,j}^k}{h_1^2} + \mathcal{O}(h_1^2) \end{aligned}$$

leads to the scheme ($\mathcal{O}(\tau), \mathcal{O}(h_1^2)$ neglected)

$$\boxed{\frac{f_{ij}^{k+1} - f_{ij}^k}{\tau} = \frac{f_{i+1,j}^k - 2f_{ij}^k + f_{i-1,j}^k}{h_1^2} + \frac{f_{i,j+1}^k - 2f_{ij}^k + f_{i,j-1}^k}{h_2^2}}$$

Unknown f_{ij}^{k+1} follows explicitly from unknown values at level k.

Inhomogeneous Linear Diffusion

Idea: control the diffusivity in a time independent (fixed) but space dependent way, i.e.,

$$D(|\nabla f_0(x,y)|^2) := \frac{1}{\sqrt{1 + |\nabla f_0(x,y)|^2/\lambda}} \qquad (\lambda > 0).$$

Diffusivity is nonlinear but PDE remains linear

$$\partial_t f = \nabla \cdot \left(D\left(|\nabla f_0(x,y)|^2\right) \nabla f \right)$$

Results: blurring is reduced but the edges are still smoothed after long evolution times.

Gauss-Laplace Pyramid

Efficient image representation by frequency space decomposition: Object information in images is often contained already in the low frequency bands of Fourier space $\Rightarrow$ no need to store all high frequency information!

Burt & Adelson (1983): Decompose an image by successive Gaussian filtering with kernel width spaced in octaves ($\times 2$).

"The Laplacian Pyramid as a compact image code", IEEE-TCOM 31, 532–540

Applications of image pyramids:

1. **Image quantization**: the Laplace pyramid coefficients are strongly decorrelated
2. **Progressive image transmission**: send first low pass image content and fill in the high frequency information when needed.
3. **Smart sensing**: control selective attention to avoid information overflow

Gauss-Laplace Pyramid Algorithm

Image decomposition in operator formalism:

original image:	G^0	reduction operator:	R
Gauss pyramid:	$G^i\big\|_{i=1}^{n}$	expansion operator:	E
Laplace pyramid:	$L^i\big\|_{i=1}^{n-1}$	smoothing filter:	B^0
		identity operator:	I

Reduction-expansion scheme: The *reduction operator subsamples* the image by a factor of two in each dimension. The *expansion operator replicates* each pixel in each dimension.

Efficiency gain: Often in computer vision most of the computation is invested in for the high resolution scales. If possible subsampling by a *pyramid scheme* and processing on a low resultion scale yields substantial gains in time efficiency.

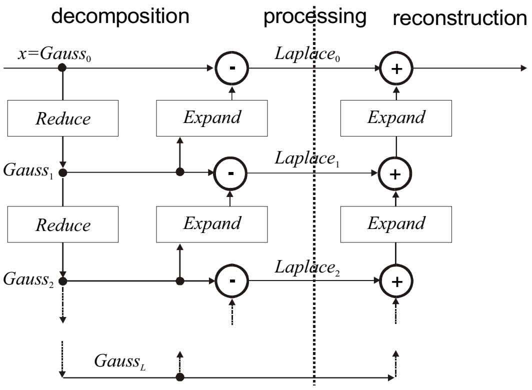
Recursive Filtering with Laplace Filter

Construction of a Laplace pyramid $L^0, L^1, \ldots, L^{n-1}$

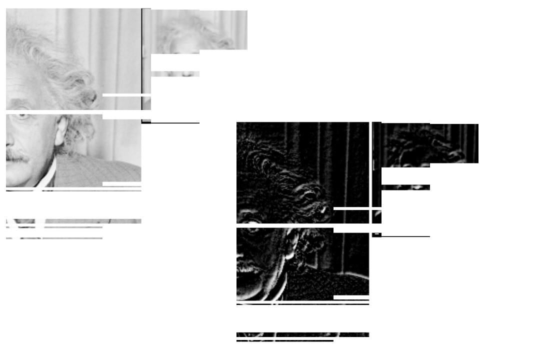
$$\begin{aligned}
\textbf{Initialization}: \quad L^0 &= G^0 - EG^1 \\
&= (I - E(RB)^0)G^0 \\
G^1 &= (RB)^0 G^0 \\
\textbf{Iteration of level } i: \quad G^{i+1} &= (RB)^i G^i \\
L^i &= G^i - EG^{i+1} \\
&= (I - E(RB)^i)G^i \\
\textbf{Reconstruction}: \quad G^{k-1} &= L^{k-1} + E\,G^k
\end{aligned}$$

Choose smoothing filter B as binomial mask $\frac{1}{16}(1,4,6,4,1)$

Schematic View of Gauss-Laplace Pyramid Algorithm



Example of a Gauss-Laplace Pyramid



Remarks on the Gauss-Laplace Pyramid

Redundancy in 1 dimension: generate $L^0, \ldots, L^{\log_2 N-1}, G^{\log_2 N}$ with approximately $2N$ coefficients for a 1-dim. image with N pixels since

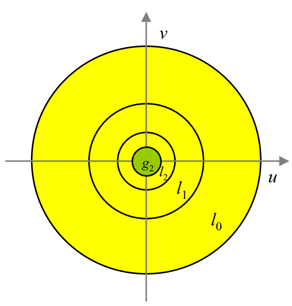
$$N + \frac{N}{2} + \frac{N}{4} + \cdots + 2 + 1 = N \sum_{i=0}^{\log_2 N} \left(\frac{1}{2}\right)^i = 2N(1 - 2^{-\log_2 N - 1}).$$

Gauss/Laplace pyramid redundancy in 2 diminesions: $\leq 33,\overline{3}\%$

$$N + \frac{N}{4} + \frac{N}{16} + \cdots + 4 + 1 = N \sum_{i=0}^{\log_2 \sqrt{N}} \left(\frac{1}{4}\right)^i = \frac{4}{3} N(1 - 2^{-\log_2 \sqrt{N} - 1}).$$

Fourier Space Decomposition by GL-Pyramid

A constant bit budget per Laplace level provides a compact and efficient image code (fixed # of bits per average power).



Load balancing code: The Gauss pyramid defines circles in Fourier space with radii $\rho_i = \frac{\rho_0}{2^i}$.

Area of the rings:

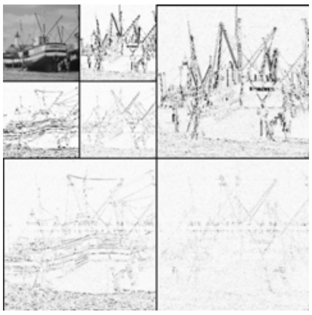
$$A_i = \pi\rho_0^2\left(\frac{1}{2^{2i}} - \frac{1}{2^{2i+2}}\right) = 3\pi\frac{\rho_0^2}{2^{2i+2}}$$

Average power Φ_i **per Laplace level** i:

$$\Phi_i A_i \approx \underbrace{\rho_0^{-2} 2^{2i}}_{\rho_i^{-2}} 3\pi \frac{\rho_0^2}{2^{2i+2}} = \frac{3\pi}{4}$$

Wavelet Bases for Image Coding

Wavelets are an orthonormal basis with selfsimilar basis functions.



Wavelets have been suggested with different regularity properties and finite support (Daubechies , Lemarie wavelets).

Self-similarity is very well adapted to power distribution in images (see GL pyramid).

Quadrature mirror filters enable efficient computation (subsampling scheme).

Nonlinear Isotropic Diffusion Filtering

Idea: we introduce a diffusivity which depends on the gradient of the time dependent intensity function, i.e., $D = D(|\nabla f|^2)$.

Diffusion across edges is reduced or suppressed.

Nonlinear diffusion equation:

$$\partial_t f = \text{div}\Big(D(|\nabla f|^2)\nabla f\Big) \quad \text{on} \quad \Omega \times (0, \infty)$$

with the original image as initial condition $f(\mathbf{x}, 0)$ on Ω and

$$\partial_n f = 0 \quad \text{on} \quad \partial\Omega \times (0, \infty)$$

$\partial_n f$ is the gradient in normal direction.

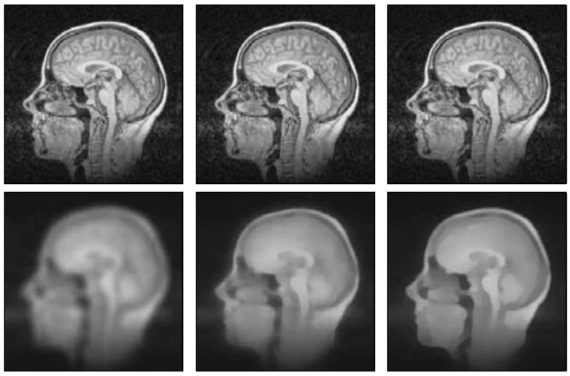
Well-posedness and scale-space properties: if the flux function

$$\Phi(s) := D(|s|^2)s$$

is monotonously increasing in s then classical mathematical theories such as monotone operators and differential inequalities ensure well-posedness.

Extremum principle: it is equivalent to the noncreation of new level-crossings under certain conditions. ⇒ *causality* property guarantees that features can be traced back from coarse to fine scales.

Axiomatization of nonlinear diffusion has been proposed.



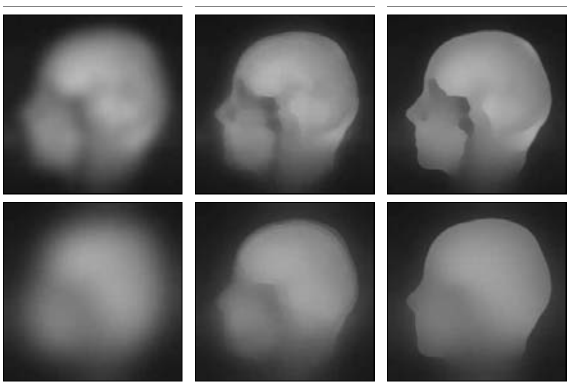


Fig. 1. Diffusion scale-spaces with a convex potential function. TOP: Original image, $\Omega = (0, 236)^2$. (A) LEFT COLUMN: Linear diffusion, top to bottom: $t = 0$, 12.5, 50, 200. (B) MIDDLE COLUMN: Inhomogeneous linear diffusion ($\lambda = 8$), $t = 0$, 70, 200, 600. (C) RIGHT COLUMN: Nonlinear isotropic diffusion with the Charbonnier diffusivity ($\lambda = 3$), $t = 0$, 70, 150, 400.

Diffusion Filtering and Energy Minimization

Consider a potential function $\Psi(|\nabla f|)$ with the property

$$\nabla\Psi(|\nabla f|) = \Phi(\nabla f) = D(|\nabla f|^2)\nabla f,$$

that is, the gradient of the potential is given by the mathematical flux $\Phi(\nabla f)$.

The energy functional

$$E(f) := \int_\Omega \Psi(|\nabla f|)dx$$

is minimized by the gradient descent method (variational calculus yields the extremality condition div $\nabla\Psi(|\nabla f|) = \Delta\Psi(|\nabla f|) = 0$)

$$\partial_t f = \text{div}\Big(D(|\nabla f|^2)\nabla f\Big)$$

Survey of methods:

method	diffusivity $D(s^2)$	potential $\Psi(s)$	$\Psi(s)$ convex for
linear diffusion	1	$s^2/2$	all s
Charbonnier	$1/\sqrt{1+s^2/\lambda^2}$	$\sqrt{\lambda^4+s^2\lambda^2}-\lambda^2$	all s
Perona-Malik	$1/\left(1+s^2/\lambda^2\right)$	$\lambda^2 \log\left(1+s^2/\lambda^2\right)/2$	$\lvert s\rvert \le \lambda$

The MATLAB program by F. D'Almeida uses the diffusivity

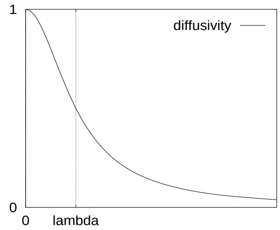
$$D(s) = \begin{cases} 1-\exp\left(-\frac{c_m}{(|s|/\lambda)^m}\right), & s>0 \\ 1, & s\le 0 \end{cases}$$

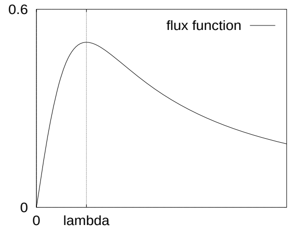
where the constant c_m is chosen such that the flux is monotonously increasing for $s \le \lambda$ ($c_m = 3.31488$ for $m = 8$). In general choose $8 \le m \le 16$.

http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=3710&objectType=FILE

Perona-Malik Nonlinear Diffusion

1D Perona-Malik Diffusion: $\partial_t f = \partial_x \big(\underbrace{D(f_x^2) f_x}_{\Phi(f_x)}\big) = \Phi'(f_x) f_{xx}$



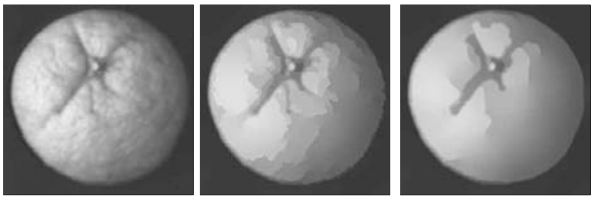


diffusivity $D(f_x^2) = \frac{1}{1+f_x^2/\lambda^2}$, flux function $\Phi(f_x) = \frac{f_x}{1+f_x^2/\lambda^2}$

Edge enhancement: Backward diffusion for $f_x > \lambda$ since $\Phi'(f_x) < 0$ (this case defines a classical ill-posed process!)

Contrast parameter: λ is denoted as the contrast parameter since it switches between forward and backward diffusion. $\Rightarrow$ edge smoothing and edge enhancement.

Problems with PM filtering: stair-casing of images



a) original image, b) PM diffusion filtering, c) regularized isotropic nonlinear diffusion

Solution: Convolve the image with a Gaussian of width σ s.t. ∇f is replaced by $\nabla(G_\sigma * f) =: \nabla f_\sigma$; $\Rightarrow$ well-posedness.

Discrete Nonlinear Diffusion

Discretization of $\partial_t f = \partial_x(D(|\nabla f_\sigma|^2)\partial_x f) + \partial_y(D(|\nabla f_\sigma|^2)\partial_y f)$:

$$\begin{aligned} \frac{df_{ij}}{dt} &= \frac{1}{h_1}\left(\frac{D_{i+1,j}+D_{ij}}{2}\frac{f_{i+1,j}-f_{ij}}{h_1} - \frac{D_{ij}+D_{i-1,j}}{2}\frac{f_{ij}-f_{i-1,j}}{h_1}\right) \\ &+ \frac{1}{h_2}\left(\frac{D_{i,j+1}+D_{ij}}{2}\frac{f_{i,j+1}-f_{ij}}{h_2} - \frac{D_{ij}+D_{i,j-1}}{2}\frac{f_{ij}-f_{i,j-1}}{h_2}\right) \end{aligned}$$

Compact notation: (use single index $k(i,j)$ for pixel (i,j))

$$\frac{df_k}{dt} = \sum_{n=1}^{2} \sum_{l \in \mathcal{N}_n(k)} \frac{D_l + D_k}{2h_n^2}(f_l - f_k)$$

where $\mathcal{N}_n(k)$ is the set of neighbors in the direction of n.

Vector matrix notation:

$$\frac{df}{dt} = A(f)\, f$$

with the matrix elements

$$a_{kl} := \begin{cases} \frac{D_k+D_l}{2h_n^2} & l \in \mathcal{N}_n(k), \\ -\sum_{n=1}^{2}\sum_{l\in\mathcal{N}_n(k)} \frac{D_l+D_k}{2h_n^2} & l = k, \\ 0 & \text{else.} \end{cases}$$

Vector-valued Nonlinear Diffusion

Naive idea: run diffusion separately in all (color) channels $\Rightarrow$ problem: edges locations might differ between channels which causes unpleasant color effects.

NL diffusion of color images with common diffusivity: (Gerig et al. 1992)
diffusivity is $D(\sum_{j=1}^{3} |\nabla f_j|^2)$

Medical imaging: vector valued nonlinear diffusion can also be used to smoothen Magnetic Resonance Images.

Separate diffusion in each color channel

Common diffusivity in all color channels