

236861 Numerical Geometry of Images

Tutorial 9

**PDEs based methods in image processing.
Scale-space.**

Anastasia Dubrovina

©2009

Overview

- ▶ Partial differential equations are increasingly being used in the image processing and computer vision community.
- ▶ Linear and non-linear scale-space.
- ▶ Discrete scale space.
- ▶ Semi-implicit numerical schemes for discretizing PDEs.

Connecting PDEs to image processing

- ▶ A *scale space* is an image representation at a continuum of scales, embedding the image f into a family $\{T_t f, t \geq 0\}$ of gradually simplified versions of it.
- ▶ Alvarez *et al.* established the connection between the scale space analysis and PDEs.
- ▶ Imposing a set of natural filtering axioms (based on desired image properties), it is proven that the resulting filtered image must necessarily be the viscosity solution of a second order parabolic PDE.

Linear scale-space

When T_t is linear, one obtains the Gaussian (linear) scale space.

The linear scale space is represented by the linear diffusion eq.

$$u_t = \Delta u; \quad u(x, 0) = f(x) \text{ ("original image")}$$

We add a scale dimension to the original image-by using the single scale-parameter t .

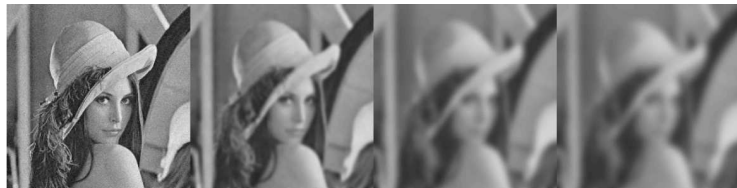


Figure: Linear scale-space.

Non-linear diffusion processes, Perona-Malik model

- ▶ When moving from coarse to fine, the linear diffusion dislocates the edges.
- ▶ Perona and Malik proposed a nonlinear adaptive process where the diffusion coefficient is spatially variable.

$$u_t = \operatorname{div}(g(|\nabla u|^2)\nabla u), \quad (1)$$

$$u(x, 0) = f(x). \quad (2)$$

The diffusivity function $g(s)$

$g(s)$ is a decreasing function $g : [0, \infty) \rightarrow [0, 1]$, with $g(0) = 1$ and $\lim_{s \rightarrow \infty} g(s) = 0$.

- ▶ Inside the regions with small gradient magnitude Eq. (1) (Perona-Malik) represents isotropic smoothing.
- ▶ At the region boundaries, where the magnitude of the gradient is large, the edges are preserved.

The proposed diffusion coefficients are:

$$g_1(s) = \frac{1}{1 + s/K_{PM}}, \text{ and } g_2(s) = e^{-\frac{s}{K_{PM}}} \quad (3)$$

where the term K_{PM} is a fixed gradient threshold.

Linear and nonlinear diffusion filtering



Figure: Left: Original cameraman image. Middle: Noisy image(random noise). Right: Denoising by linear diffusion. Bottom right: Denoising by Perona-Malik nonlinear diffusion. Parameters: 25 iterations, $t = 0.15$.

Regularized Perona-Malik model

- ▶ Motivated by the ill-posedness of the Perona-Malik model, Catte *et al.* (92) proposed a regularization of the Perona-Malik model.

They included in the diffusion coefficient $g(|\nabla u|^2)$ the gradient ∇u by a smooth version of it, namely $G_\sigma * \nabla u$, where G_σ is a smoothing Gaussian kernel of variance σ .

$$u_t = \operatorname{div}(g(|\nabla(G_\sigma * u)|^2)\nabla u), \quad (4)$$

$$u(x, 0) = f(x). \quad (5)$$

- ▶ In the case of very noisy initial data, a main drawback of the Perona-Malik is the impossibility of separating between true and false edges (created by the noise). The Perona-Malik regularized model avoids this, because Eq. 4 diffuses only if the gradient is estimated to be small.

Discrete diffusion filtering - the general model

A discrete image is a vector $f \in \mathbb{R}^N$, $N \geq 2$, $J = \{1, 2, \dots, N\}$.
 $(u_k)_{k \in \mathbb{N}_0}$ - sequence of processed versions of f , using:

$$\begin{cases} u^0 = f, \\ u^{k+1} = Q(u^k)u^k, \quad k \in \mathbb{N}_0. \end{cases}$$

Discrete diffusion requirements

The MATRIX $Q = (q_{ij})$ satisfies:

$$(D_1): \quad Q \in C(\mathbb{R}^N, \mathbb{R}^{N \times N})$$

$$(D_2): \quad q_{ij} = q_{ji}, \quad \forall i, j \in J$$

$$(D_3): \quad \sum_{j \in J} q_{ij} = 1, \quad \forall i \in J$$

$$(D_4): \quad q_{ij} \geq 0, \quad \forall i, j \in J$$

$$(D_5): \quad \text{irreducibility}$$

$$(D_6): \quad q_{ii} > 0, \quad \forall i \in J$$

(D_5) : $\forall i, j \in J$ there exist $k_0, \dots, k_r \in J$ with $k_0 = i$ and $k_r = j$ such that $q_{k_p k_{p+1}} \neq 0, \forall p = 0, \dots, r - 1$. Or: we can connect any 2 pixels with non-vanishing diffusivities.

Discrete Scale-space Properties

Under $D_1 - D_6$ the filtering process is well-posed and satisfies the following properties:

- ▶ Extremum Principle
- ▶ Average Grey-Level invariance
- ▶ Smoothing transformation (Lyapunov functions)
- ▶ Convergence to a constant steady-state

Extremum Principle and Conservation of average grey-value

Theorem 1 Let $f \in \mathbb{R}^N$ and let $(u_k)_{k \in \mathbb{N}_0}$ be the sequence of filtered image according according to $D_1 - D_6$. Then

$$a \leq u_i^k \leq b, \quad \forall i \in J, \quad \forall k \in \mathbb{N}_0$$

where $a = \min_{j \in J} f_j$, $b = \max_{j \in J} f_j$.

Theorem 2 The average gray level $\mu = \frac{1}{N} \sum_{j \in J} f_j$ is not affected by the discrete diffusion filter:

$$\frac{1}{N} \sum_{j \in J} u_j^k = \mu, \quad \forall k \in \mathbb{N}_0$$

Smoothing transformation (Lyapunov functions)

- ▶ The p -norms are decreasing in k for all $p \geq 1$

$$\|u_k\|_p = \left(\sum_{j \in J} |u_j^k|^p \right)^{1/p}$$

- ▶ All even central moments are decreasing in k

$$M_{2n}[u^k] = \frac{1}{N} \sum_{j \in J} (u_j^k - \mu)^{2n} \quad (n \in \mathbb{N})$$

- ▶ The entropy (measure of uncertainty and missing information) is increasing in k

$$S[u^k] = - \sum_{j \in J} u_j^k \ln u_j^k$$

Convergence to a steady state

$$\lim_{k \rightarrow \infty} u_i^k = \mu, \quad \forall i \in J$$

The discrete scale -space evolution tends to the most global image representation that is possible: constant image with the same average grey-level as f .

Discretization of PDEs

Let us be given a general PDE of the form

$$\partial_t u = \mathcal{A}(u)u; \quad u : [0, \infty) \times \Omega \subset \mathbb{R}^m \mapsto \mathbb{R}, \quad u(0, x) = u_0(x),$$

where $\mathcal{A}(u)$ is some differential operator. When discretized, the operator $\mathcal{A}(u)$ is replaced by a matrix $A(u)$. The PDE is solved numerically, evaluating the solution u^k at iteration k . The way in which the k -th iteration is computed is called a *numerical scheme*.

We distinguish between the following schemes:

- ▶ *Explicit*: $u^{k+1} = Q(u^k)u^k$
- ▶ *Semi-implicit*: $Q(u^k)u^{k+1} = u^k$ (linear)
- ▶ *Fully implicit*: $Q(u^{k+1})u^{k+1} = u^k$ (non-linear)

where $Q(u)$ is some other matrix given in terms of $A(u)$.

Example: 1D nonlinear diffusion

Consider the 1D nonlinear diffusion equation

$$\partial_t u = \partial_x (g(u_x) u_x)$$

where $g \in [0, 1]$ is some *edge indicator*.

Let us discretize the PDE on a uniform grid of size N with step h , such that u is represented as a vector $(u_1; \dots; u_N)$. Let the time step be τ . Ignoring boundary conditions,

$$\partial_x (g u_x) \approx \sum_{j \in \mathcal{N}(i)} \frac{g_j + g_i}{2h^2} (u_j - u_i)$$

where $\mathcal{N}(i) = \{i - 1, i + 1\}$ and $g_i = g_i(u_{j \in \mathcal{N}(i)})$.

Example: 1D nonlinear diffusion (cont.)

We can represent the r.h.s. of the discretized equation as $A(u)u$, where the matrix A is given by

$$a_{ij}(u) = \begin{cases} \frac{g_i + g_j}{2h^2} & j \in \mathcal{N}(i) \\ -\sum_{k \in \mathcal{N}(i)} \frac{g_i + g_k}{2h^2} & i = j \\ 0 & \text{else} \end{cases} = \begin{cases} \frac{g_i + g_j}{2h^2} & j \in \mathcal{N}(i) \\ -\sum_{k \neq i} a_{ik} & i = j \\ 0 & \text{else} \end{cases}$$

(remember that g_i is a function of u). The matrix A has a tridiagonal form, is symmetric and the sum of its rows or columns vanishes.

Since $g_i \geq 0$, it follows that the off-diagonal elements of A are non-negative, i.e. $a_{ij} \geq 0$ for $j \neq i$.

Explicit scheme for 1D nonlinear diffusion

The explicit scheme is obtained from the simplest discretization of the equation:

$$\frac{u^{k+1} - u^k}{\tau} = A(u^k)u^k$$

which can be written as

$$u^{k+1} = (I + \tau A(u^k))u^k \equiv Q(u^k)u^k.$$

Stability of the 1D explicit scheme

Question: for which values of τ the explicit scheme is stable ?

The stability analysis can be performed by definition, requiring that $\|Q\| = \max|\lambda^Q| \leq 1$ (i.e. the iteration matrix acts as a contraction). Yet, this gives a bound on τ expressed dependently of $A(u^k)$.

Weickert presented the following condition based on Lyapunov stability analysis:

$$\tau < \frac{1}{\max_i \sum_{j \neq i} a_{ij}}$$

In the common case $h = 1$, since $g \leq 1$, we have $\tau < \frac{1}{2}$.

Semi-implicit scheme for 1D nonlinear diffusion

The semi-implicit scheme is obtained from the following discretization:

$$\frac{u^{k+1} - u^k}{\tau} = A(u^k)u^{k+1},$$

which leads to $(I - \tau A(u^k))u^{k+1} = u^k$ and can be written as

$$u^{k+1} = (I - \tau A(u^k))^{-1}u^k \equiv B(u^k)^{-1}u^k,$$

The semi-implicit scheme is unconditionally stable (i.e. stable for every $\tau > 0$).

The matrix $B = I - \tau A$ is tridiagonal and positive definite. On each iteration, we need to invert B . This can be performed efficiently in $\mathcal{O}(N)$ operations using Thomas algorithm.

Thomas algorithm for tridiagonal matrix inversion

Goal: solve the system $Bu = d$

Step 1: LR-decomposition - decompose the matrix

$$B = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ \gamma_1 & \alpha_2 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & & \gamma_{N-1} & \alpha_N \end{bmatrix}$$

into lower and upper bidiagonal matrices L and R s.t. $B = LR$, where

$$L = \begin{bmatrix} 1 & & & & \\ l_1 & 1 & & & \\ & \ddots & \ddots & & \\ & & & l_{N-1} & 1 \end{bmatrix} \quad R = \begin{bmatrix} m_1 & r_1 & & & \\ & \ddots & \ddots & & \\ & & & m_{N-1} & r_{N-1} \\ & & & & m_N \end{bmatrix}$$

Thomas algorithm for tridiagonal matrix inversion (cont.)

Step 1 (cont.) The coefficients are defined in the following manner:

$$r_i = \beta_i$$

whereas m_i and l_i are computed recursively:

- ▶ $m_1 = \alpha_1$
- ▶ for $i = 1, 2, \dots, N - 1$:
- ▶ $l_i := \gamma_i / m_i$
- ▶ $m_{i+1} := \alpha_{i+1} - l_i \beta_i$

Thomas algorithm for tridiagonal matrix inversion (cont.)

Solving $LRu = d$ is done in two steps:

Step 2: Forward substitution - solve $Ly = d$ for y iteratively:

- ▶ $y_1 = d_1$
- ▶ for $i = 2, \dots, N$:
- ▶ $y_i := d_i - l_{i-1}y_{i-1}$

Step 3: Backward substitution - solve $Ru = y$ for u iteratively:

- ▶ $u_N = y_N/m_N$
- ▶ for $i = N - 1, N - 2, \dots, 1$:
- ▶ $u_i := (y_i - \beta_i u_{i+1})/m_i$

Generalization to m dimensions

The m -dimensional version of the diffusion equation is

$$u_t = \operatorname{div} (g(\|\nabla u\|)\nabla u) = \sum_{i=1}^m \partial_{x_i} (g(\|\nabla u\|)\partial_{x_i} u) = \sum_{i=1}^m \mathcal{A}_i(u)u;$$
$$u : [0, \infty) \times \Omega \subset \mathbb{R}^m \mapsto \mathbb{R}; \quad u(0, x) = u_0(x).$$

The equation is discretized on an $N_1 \times \dots \times N_m$ grid, with pixel sizes h_1, \dots, h_m .

Having u represented as an $N_1 \cdot \dots \cdot N_m$ -long vector, the r.h.s. is written as $\sum_{i=1}^m \mathcal{A}_i(u)$. The matrix \mathcal{A}_i corresponds to discrete derivative along the i -th axis.

Explicit m -dimensional scheme

Explicit m -dimensional scheme now has the following form:

$$u^{k+1} = \left(I + \tau \sum_{l=1}^m A_l(u^k) \right) u^k$$

The scheme is stable for

$$\tau < \frac{1}{\max_i \sum_{j \neq i} a_{ij}},$$

where

$$\max_i \sum_{j \neq i} a_{ij} \leq 2 \left(\frac{1}{h_1^2} + \dots + \frac{1}{h_m^2} \right) \sup g \leq 2 \left(\frac{1}{h_1^2} + \dots + \frac{1}{h_m^2} \right).$$

In the common case $h_1 = \dots = h_m = 1$, we have $\tau < \frac{1}{2m}$, i.e. the step size becomes smaller as the dimension grows.

Semi-implicit m -dimensional scheme

Explicit m -dimensional scheme now has the following form:

$$u^{k+1} = \left(I - \tau \sum_{i=1}^m A_i(u^k) \right)^{-1} u^k$$

Main caveat: the matrix is now not tridiagonal and cannot be inverted efficiently.

AOS scheme

As an alternative to semi-implicit scheme, Weickert et al. proposed the *additive operator splitting* (AOS) scheme:

$$u^{k+1} = \frac{1}{m} \sum_{i=1}^m \left(I - m\tau A_i(u^k) \right)^{-1} u^k$$

Here, unlike the previous case, all the matrices to be inverted are tridiagonal and the Thomas algorithm can be employed again.

LOD scheme

Fully implicit m -dimensional scheme can be written in the following form:

$$u^{k+1} = \prod_{i=1}^m \left(I - \tau A_i(u^k) \right)^{-1} u^k$$

It is an *multiplicative operator splitting*, called *locally one dimensional* (LOD) scheme, as it consists of subsequently applying the 1D operators $(I - \tau A_i(u^k))^{-1}$.

Main caveat: since the operators do not commute, the order of their application is important. As the result, e.g., rotating the image by 90 degrees changes the result.