

Image Evolutions

Images as functions

Last time we considered images as functions

$$I : \Omega \rightarrow \mathbb{R}^n$$

Image evolutions

Now we consider image *evolutions over time*

$$I : \Omega \times [0, T] \rightarrow \mathbb{R}^n.$$

Assuming a 2D-domain, the image has now three parameters: $I(x, y, t)$.

Discretized view

In practice, this means starting with an initial image I_0 , and generating a *sequence* of images $I_k : \Omega \rightarrow \mathbb{R}^n$:

$$I_0, I_1, I_2, I_3, \dots$$

by some specific algorithm. One is interested in the result image I_k for some $k \geq 1$.

Image Evolutions

Image evolution

$$I : \Omega \times [0, T] \rightarrow \mathbb{R}^n.$$

Usual form

The evolution is usually specified in the form

$$(\partial_t I)(x, y, t) = f(x, y, t).$$

The right hand side is some function $f : \Omega \times [0, T] \rightarrow \mathbb{R}^n$ which may depend on the image I it self (at time t), and its *derivatives*.

Incremental update

At each time step t , this gives an *incremental update* of the value $I(x, y, t)$ of I at each point $(x, y) \in \Omega$:

$$I(x, y, t + dt) = I(x, y, t) + \tau \cdot f(x, y, t)$$

with a small time step $\tau > 0$, or, in the discretized view,

$$I_{k+1}(x, y) := I_k(x, y) + \tau \cdot f_k(x, y)$$

Diffusion

For simplicity we will work with grayscale images $I : \Omega \times [0, T] \rightarrow \mathbb{R}$.

Diffusion

A general diffusion is given by the update equation

$$\partial_t I = \operatorname{div}(D \nabla I)$$

$D : \Omega \times [0, T] \rightarrow \mathbb{R}^{2 \times 2}$ is a 2×2 matrix called the **diffusion tensor**. It may vary depending on the position in Ω and the time t , and may depend on the image I itself.

Differential operators ∇ and div are only w.r.t. spatial variables x, y .

Gradient of a scalar image $I : \Omega \times [0, T] \rightarrow \mathbb{R}$

$$\nabla I : \Omega \times [0, T] \rightarrow \mathbb{R}^2, \quad (\nabla I)(x, y, t) = \begin{pmatrix} (\partial_x I)(x, y, t) \\ (\partial_y I)(x, y, t) \end{pmatrix}$$

Divergence of a 2D-vector field $v : \Omega \times [0, T] \rightarrow \mathbb{R}^2$

$$\operatorname{div} v : \Omega \times [0, T] \rightarrow \mathbb{R}, \quad (\operatorname{div} v)(x, y, t) = (\partial_x v_1)(x, y, t) + (\partial_y v_2)(x, y, t)$$

Diffusion: Computation of the right hand side

Diffusion

$$(\partial_t I)(x, y, t) = (\operatorname{div}(D\nabla I))(x, y, t)$$

1. Start with image $I : \Omega \times [0, T] \rightarrow \mathbb{R}$, values $I(x, y, t) \in \mathbb{R}$
2. Compute the gradient

$$g(x, y, t) := (\nabla I)(x, y, t) = \begin{pmatrix} (\partial_x I)(x, y, t) \\ (\partial_y I)(x, y, t) \end{pmatrix} \in \mathbb{R}^2$$

3. Multiply the diffusion tensor $D(x, y, t) \in \mathbb{R}^{2 \times 2}$ with the gradient $g(x, y, t) \in \mathbb{R}^2$:

$$v(x, y, t) := D(x, y, t)g(x, y, t) \in \mathbb{R}^2$$

4. Take divergence of v :

$$d(x, y, t) := (\operatorname{div} v)(x, y, t) = (\partial_x v_1)(x, y, t) + (\partial_y v_2)(x, y, t) \in \mathbb{R}$$

Types of Diffusion: Isotropic/Nonisotropic

► **Isotropic diffusion:**

$D(x, y, t) \in \mathbb{R}^{2 \times 2}$ is a diagonal matrix with two equal entries

$$D(x, y, t) = \varphi(x, y, t) \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \varphi(x, y, t) & 0 \\ 0 & \varphi(x, y, t) \end{pmatrix} \quad (1)$$

with a scalar $\varphi : \Omega \times [0, T] \rightarrow \mathbb{R}$. φ is called **diffusivity**.

Then $\operatorname{div}(D\nabla I) = \operatorname{div}(\varphi\nabla I)$. Diffusion equation becomes:

$$\partial_t I = \operatorname{div}(\varphi\nabla I)$$

► **Anisotropic Diffusion:**

$D(x, y, t) \in \mathbb{R}^{2 \times 2}$ is not isotropic (a general positive definite, symmetric matrix).

Types of Diffusion: Linear/Nonlinear

- ▶ **Linear Diffusion:**

$D(x, y, t) \in \mathbb{R}^{2 \times 2}$ does not depend on the image I at time t .

- ▶ **Nonlinear Diffusion:**

$D(x, y, t) \in \mathbb{R}^{2 \times 2}$ depends on the image I at time t .

$$D = \mathbb{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \Rightarrow \operatorname{div} (D \nabla I) = \Delta I \quad (2)$$

Special case: Dissipation (linear isotropic diffusion)

Constant diffusion tensor at each point (x, y, t) :

$$D(x, y, t) := \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

Then $D\nabla I = \nabla I$, so that we get the Laplacian

$$\operatorname{div}(D\nabla I) = \operatorname{div}(\nabla I) = \Delta I$$

Diffusion equation simplifies to

$$(\partial_t I)(x, y, t) = (\Delta I)(x, y, t)$$

Special case: Perona-Malik (nonlinear isotropic diffusion)

Diffusion tensor depends on the image I :

$$D(x, y, t) = \begin{pmatrix} \varphi(x, y, t) & 0 \\ 0 & \varphi(x, y, t) \end{pmatrix}$$

$$\varphi(x, y, t) = \frac{1}{\sqrt{|\nabla I(x, y, t)|^2 + \varepsilon}}$$

Discretization of isotropic diffusion

Diffusion:

$$\partial_t I = \operatorname{div}(\varphi \nabla I) = \operatorname{div} \begin{pmatrix} \varphi \partial_x I \\ \varphi \partial_y I \end{pmatrix} = \partial_x(\varphi \partial_x I) + \partial_y(\varphi \partial_y I).$$

Temporal derivative

Discretization of $(\partial_t I)(x, y, t)$ by forward differences with time step τ :

$$\frac{I(x, y, t + \tau) - I(x, y, t)}{\tau}$$

Discretization of isotropic diffusion

Diffusion:

$$\partial_t I = \operatorname{div}(\varphi \nabla I) = \operatorname{div} \begin{pmatrix} \varphi \partial_x I \\ \varphi \partial_y I \end{pmatrix} = \partial_x(\varphi \partial_x I) + \partial_y(\varphi \partial_y I).$$

Spatial derivatives

Discretization of ∂_x, ∂_y by central differences with step $\frac{1}{2}$:

$$\partial_x(\varphi \partial I)(x, y, t) : \quad (\varphi \partial_x I)(x + \frac{1}{2}, y, t) - (\varphi \partial_x I)(x - \frac{1}{2}, y, t)$$

$$\partial_y(\varphi \partial I)(x, y, t) : \quad (\varphi \partial_y I)(x, y + \frac{1}{2}, t) - (\varphi \partial_y I)(x, y - \frac{1}{2}, t)$$

$$(\varphi \partial_x I)(x + \frac{1}{2}, y, t) : \quad \varphi(x + \frac{1}{2}, y, t) (I(x + 1, y, t) - I(x, y, t))$$

$$(\varphi \partial_x I)(x - \frac{1}{2}, y, t) : \quad \varphi(x - \frac{1}{2}, y, t) (I(x, y, t) - I(x - 1, y, t))$$

$$(\varphi \partial_y I)(x, y + \frac{1}{2}, t) : \quad \varphi(x, y + \frac{1}{2}, t) (I(x, y + 1, t) - I(x, y, t))$$

$$(\varphi \partial_y I)(x, y - \frac{1}{2}, t) : \quad \varphi(x, y - \frac{1}{2}, t) (I(x, y, t) - I(x, y - 1, t))$$

Discretization of isotropic diffusion

Diffusion:

$$\partial_t I = \operatorname{div}(\varphi \nabla I) = \operatorname{div} \begin{pmatrix} \varphi \partial_x I \\ \varphi \partial_y I \end{pmatrix} = \partial_x(\varphi \partial_x I) + \partial_y(\varphi \partial_y I).$$

Diffusivity

Approximate the diffusivity φ at half-pixel locations by averaging:

$$\varphi(x + \frac{1}{2}, y, t) : \frac{1}{2} \left(\varphi(x + 1, y, t) + \varphi(x, y, t) \right) =: \varphi_r$$

$$\varphi(x - \frac{1}{2}, y, t) : \frac{1}{2} \left(\varphi(x - 1, y, t) + \varphi(x, y, t) \right) =: \varphi_l$$

$$\varphi(x, y + \frac{1}{2}, t) : \frac{1}{2} \left(\varphi(x, y + 1, t) + \varphi(x, y, t) \right) =: \varphi_u$$

$$\varphi(x, y - \frac{1}{2}, t) : \frac{1}{2} \left(\varphi(x, y - 1, t) + \varphi(x, y, t) \right) =: \varphi_d$$

Discretization: Final scheme

Diffusion:

$$\partial_t I = \operatorname{div}(\varphi \nabla I) = \operatorname{div} \begin{pmatrix} \varphi \partial_x I \\ \varphi \partial_y I \end{pmatrix} = \partial_x(\varphi \partial_x I) + \partial_y(\varphi \partial_y I).$$

Discretized

$$\begin{aligned} \frac{I(x, y, t + \tau) - I(x, y, t)}{\tau} = & \varphi_r I(x + 1, y, t) + \varphi_l I(x - 1, y, t) \\ & + \varphi_u I(x, y + 1, t) + \varphi_d I(x, y - 1, t) \\ & - (\varphi_r + \varphi_l + \varphi_u + \varphi_d) I(x, y, t) \end{aligned}$$

Final scheme

$$\begin{aligned} I(x, y, t + \tau) = I(x, y, t) + \tau \left(& \varphi_r I(x + 1, y, t) + \varphi_l I(x - 1, y, t) \right. \\ & + \varphi_u I(x, y + 1, t) + \varphi_d I(x, y - 1, t) \\ & \left. - (\varphi_r + \varphi_l + \varphi_u + \varphi_d) I(x, y, t) \right) \end{aligned}$$

Discretization: Boundary conditions

A natural assumption is to have the gradient vanish at the image boundaries, meaning that $\frac{\partial}{\partial x} I = 0$ at the left and right boundary and $\frac{\partial}{\partial y} I = 0$ at the top and bottom boundary. This ensures, that the average grey value of the image is preserved. you implement this by setting

$$I(-1, y, t) := I(0, y, t)$$

$$I(w, y, t) := I(w - 1, y, t)$$

$$I(x, -1, t) := I(x, 0, t)$$

$$I(x, h, t) := I(x, h - 1, t)$$

This simply means clamping the pixel locations back to Ω .