Medical Image Analysis and Processing

Image Noise Filtering

Anisotropic Diffusion Filter

Emad Fatemizadeh

Distance/online Course: Session 12

Date: 06 April 2021, 17th Farvardin 1400

Contents

Motivation

- Edge Preserving Filtering
- Edge-Aware filtering

Mathematical Background

› Let $u(x_1, x_2)$ is 2D scaler function and $J \in \mathbb{R}^2$ a 2D vector function:

› Gradient of u, ∇u:

$$\nabla u = \left[\frac{\partial u}{\partial x_1} \quad \frac{\partial u}{\partial x_2}\right]^t, t: Transpose$$

› Divergence of vector $\vec{J}$, $\nabla.\vec{J}$:

$$\vec{J} = [J_1 \quad J_2]^t \Rightarrow \nabla.\vec{J} = \frac{\partial J_1}{\partial x_1} + \frac{\partial J_2}{\partial x_2}$$

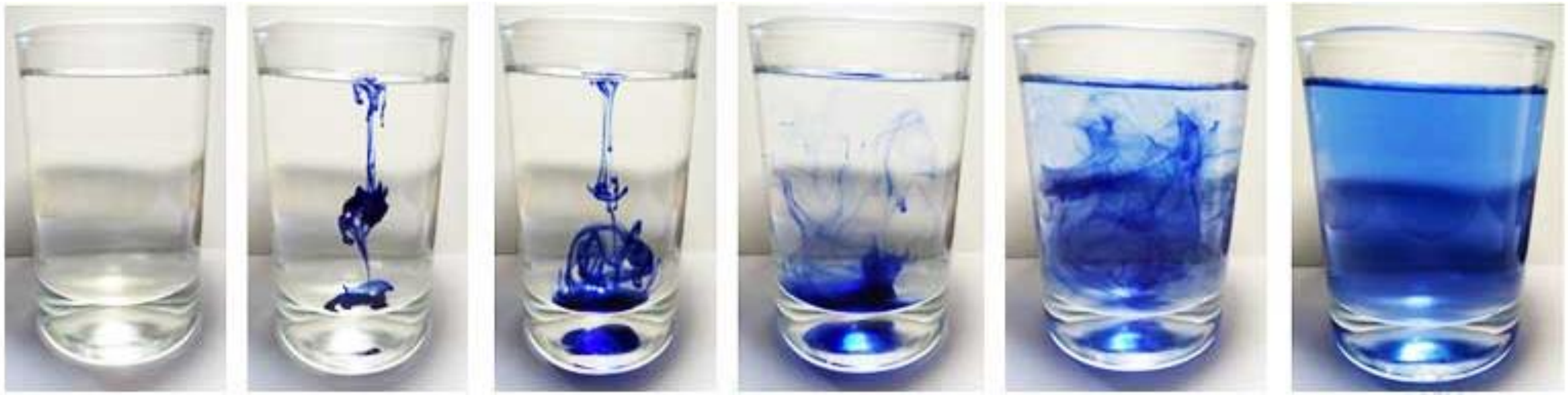
› Laplacian of u, $\nabla^2 u$:

$$\nabla^2 u = \nabla.(\nabla u) = \frac{\partial^2 u}{\partial x_1^2} + \frac{\partial^2 u}{\partial x_2^2}$$

Physical Background

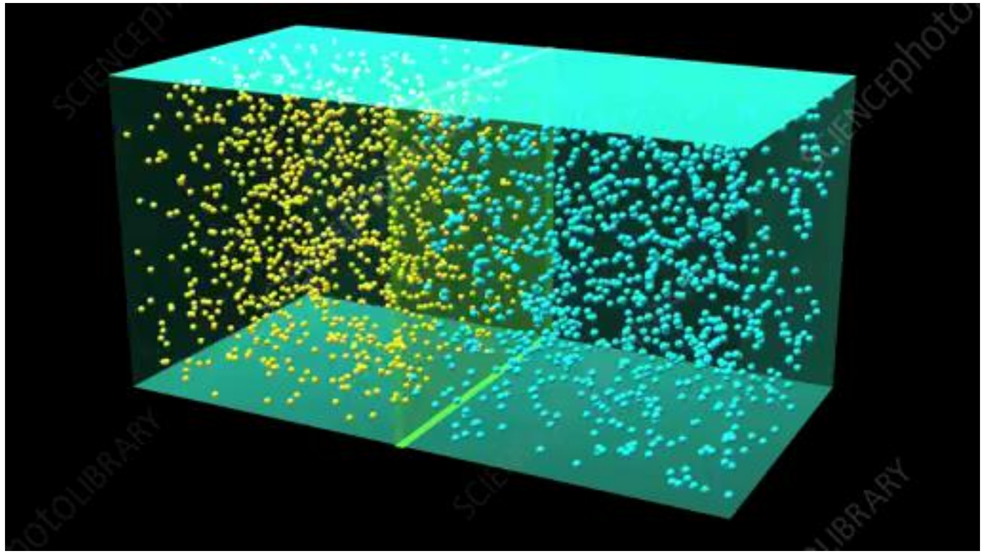
› Diffusion:

Diffusion is the net movement of anything (for example, atom, ions, molecules, and etc.) from a region of higher concentration to a region of lower concentration.



Physical Background

- Diffusion: A distinguishing feature of diffusion is that it depends on particle random walk (Brownian Motion), and results in mixing or mass transport without requiring directed bulk motion.



Physical Background

› Fick First Law:

$$\vec{J} = -D\nabla u$$

› u: Particle Concentration

› $\vec{J}$: Diffusion flux vector, amount of substance per unit area per unit time in each direction

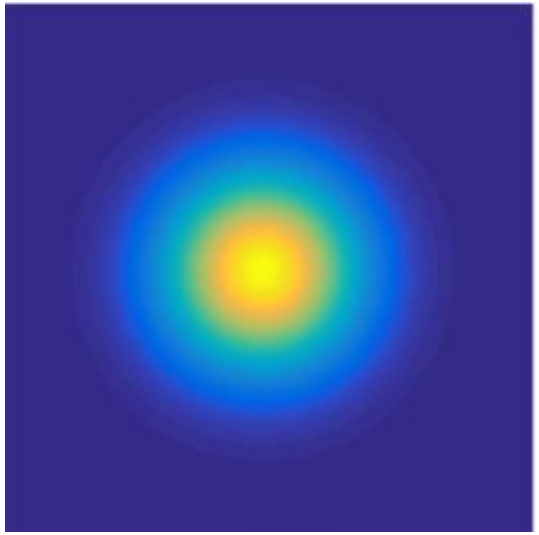
› D: is the diffusion coefficient (Tensor):

A Positive Definite Symmetric Matrix

Physical Background

› Isotropic vs Anisotropic Diffusion:

› Isotropic: Diffusion Coefficient is the same in every direction.



$$\vec{J} \parallel \nabla u$$

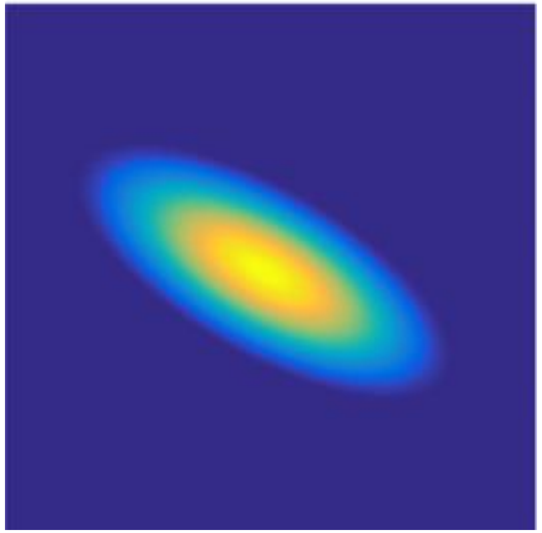
$$\mathcal{D} = \begin{bmatrix} D & 0 \\ 0 & D \end{bmatrix}$$

Physical Background

› Isotropic vs. Anisotropic Diffusion:

› Anisotropic: Different diffusion coefficients along different directions.

$$\vec{J} \nparallel \nabla u$$



$$\mathcal{D} = \begin{bmatrix} D_{x_1 x_1} & D_{x_1 x_2} \\ D_{x_2 x_1} & D_{x_2 x_2} \end{bmatrix}, D_{x_1 x_2} = D_{x_2 x_1}$$

Physical Background

- Inhomogeneous vs. homogeneous diffusion:
- Homogeneous Diffusion: $D = D_0$ is constant tensor
- Inhomogeneous Diffusion: $D = D(x_1, x_2)$ is spatial variant tensor.

Physical Background

- Mass Conservation (continuity equation):

$$\frac{\partial u}{\partial t} = -\nabla . \vec{J} = -\mathrm{div}(\vec{J})$$

- Using Fick's first law:

$$\frac{\partial u}{\partial t} = -\nabla . \vec{J} = -\nabla . (-D\nabla u) = \nabla . (D\nabla u)$$

- For homogeneous (Constant D) diffusion:

$$\frac{\partial u}{\partial t} = \nabla . (D\nabla u) = D\nabla . (\nabla u) = D\nabla^2(u)$$

- $\nabla^2(u)$: Laplacian

Physical Background

- Fick's second law:

$$\frac{\partial u}{\partial t} = \nabla.(D\nabla u)$$

- For homogeneous (Constant D) diffusion:

$$\frac{\partial u}{\partial t} = D\nabla^2(u) = D\left(\frac{\partial^2 u}{\partial x_1^2} + \frac{\partial^2 u}{\partial x_2^2}\right)$$

- Similar to heat equation

Homogeneous Linear Diffusion Filtering

- Consider the following isotropic diffusion problem:

$$\begin{cases} \dfrac{\partial u}{\partial t} = D\nabla^2 u & -\infty < x_1, x_2 < +\infty \\ u(x_1, x_2, t = 0) = u_0(x_1, x_2) \end{cases}$$

- A PDE with initial condition (not boundary condition)
- $u_0(x_1, x_2)$: Noisy image as initial condition
- It can be shown:

$$u(x_1, x_2) = u_0(x_1, x_2) ** h(x_1, x_2, t)$$

$$h(x_1, x_2, t) = \frac{1}{4\pi D t} exp\left(-\frac{x_1^2 + x_2^2}{4Dt}\right)$$

Homogeneous Linear Diffusion Filtering

- Solution:

$$u(x_1, x_2) = u_0(x_1, x_2) ** h(x_1, x_2, t)$$

$$h(x_1, x_2, t) = \frac{1}{4\pi Dt} exp\left(-\frac{x_1^2 + x_2^2}{4Dt}\right)$$

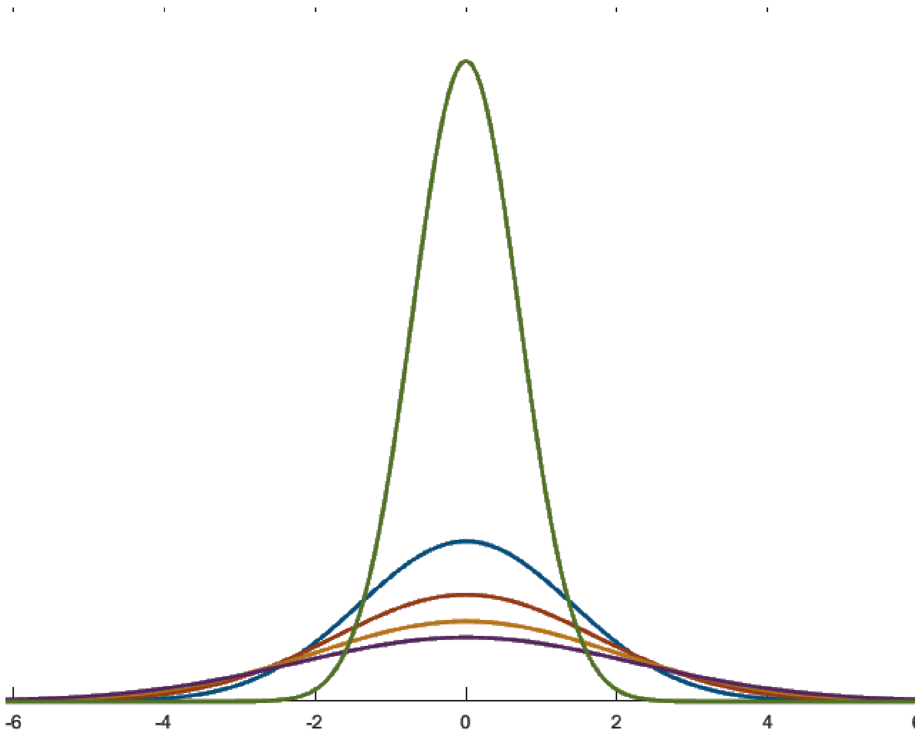
- Old fashion gaussian filtering ($2Dt = \sigma^2$)

$$h(x_1, x_2, t) = \frac{1}{2\pi\sigma^2} exp\left(-\frac{x_1^2 + x_2^2}{2\sigma^2}\right)$$

- A scale-space filtering!

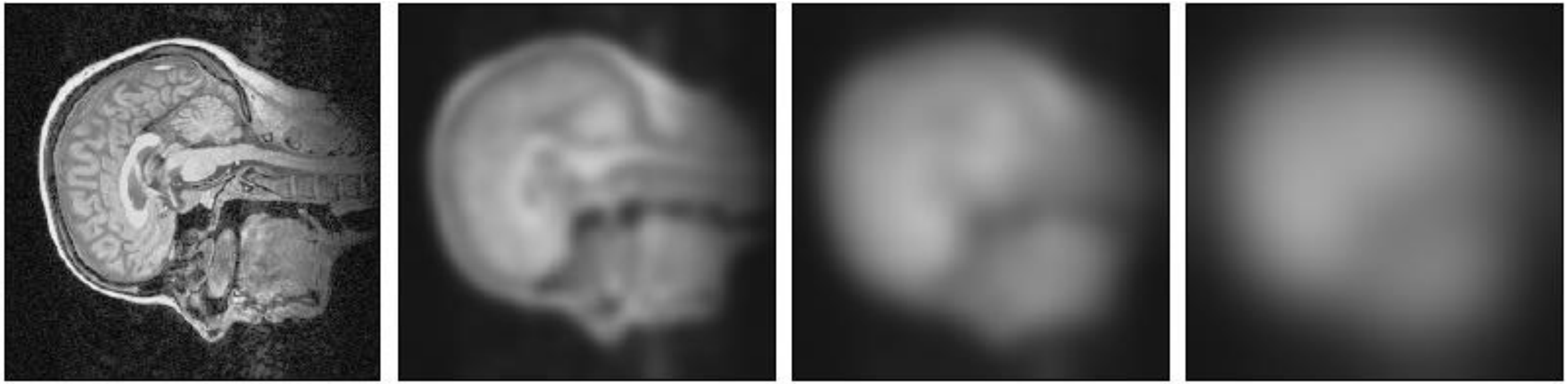
Homogeneous Linear Diffusion Filtering

› A scale-space filtering, $h(r, t)$, $\quad r = \sqrt{x_1^2 + x_2^2}$



Homogeneous Linear Diffusion Filtering

› Scale-space filtering example:



Homogeneous Linear Diffusion Filtering

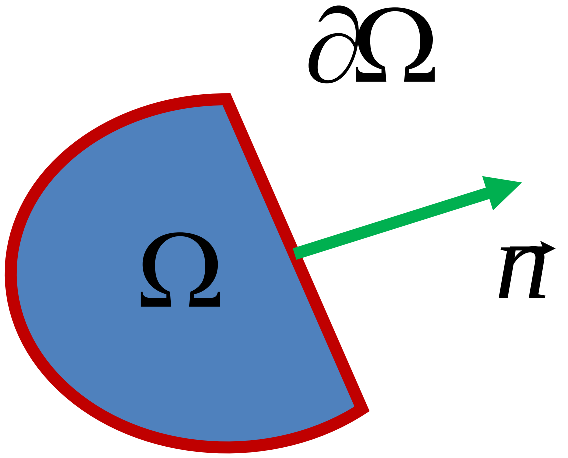
› A more realistic processing (Ω: image domain)

$$\begin{cases} \dfrac{\partial u}{\partial t} = D\nabla^2 u & (x_1, x_2) \in \Omega \\ u(x_1, x_2, t = 0) = u_0(x_1, x_2) & \\ \left.\dfrac{\partial u}{\partial \vec{n}}\right|_{\partial\Omega} = 0 & \end{cases}$$

› Ω: image domain

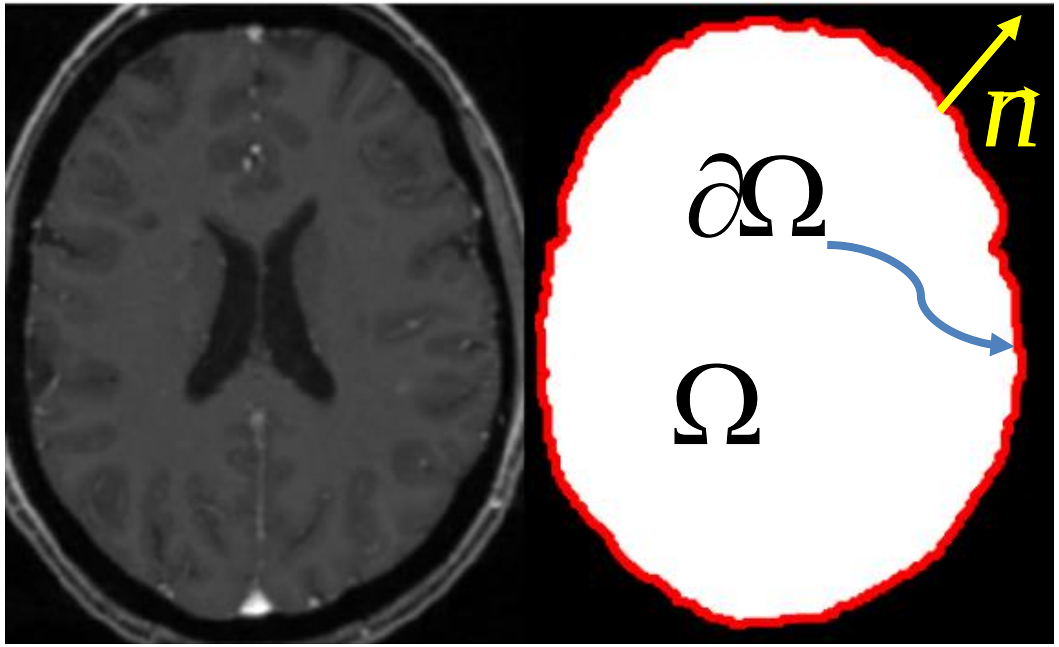
› ∂Ω: image boundary

› $\vec{n}$: image normal vector



Homogeneous Linear Diffusion Filtering

› Boundary extraction to avoid unnecessary image blurring



Inhomogeneous Linear Diffusion Filtering

- An Edge-Aware diffusion filter (version #1)
- $\frac{\partial u}{\partial t} = \nabla.\left(g(|\nabla f|^2)\nabla u\right), \quad (x_1, x_2) \in \Omega$
- $u(x_1, x_2, t = 0) = f(x_1, x_2)$
- $\left.\frac{\partial u}{\partial \vec{n}}\right|_{\partial\Omega} = 0$
- $f(x_1, x_2)$: Noisy image as initial condition
- $g(s)$: A monotonically decreasing function

Inhomogeneous Linear Diffusion Filtering

- Diffusion Coefficient function
- $g(s^2)$: A monotonically decreasing function
- $g(s^2) = \frac{g_0}{\sqrt{1+s^2/\lambda^2}}$, Charbonnier
- $g(s^2) = \frac{g_0}{1+s^2/\lambda^2}$, Perona-Malik
- $g(s^2) = g_0 exp(-s^2/\lambda^2)$, Gaussian

Inhomogeneous Linear Diffusion Filtering

- An Edge-Aware diffusion filter (version #2)
- f is too noisy for gradient estimation, try smoothing (regularization)
- $\frac{\partial u}{\partial t} = \nabla.(g(|\nabla f_\sigma|^2)\nabla u), (x_1, x_2) \in \Omega$
- $u(x_1, x_2, t=0) = f(x_1, x_2), \qquad \left.\frac{\partial u}{\partial \vec{n}}\right|_{\partial\Omega} = 0$
- $f(x_1, x_2)$: Noisy image as initial condition
- where, $f_\sigma = f * G_\sigma$(Gaussian Smoothing)

Inhomogeneous Nonlinear Diffusion Filtering

- An Edge-Aware diffusion filter (version #3)
- Use u for gradient smoothing,
- $\frac{\partial u}{\partial t} = \nabla.\left(g(|\nabla u|^2)\nabla u\right), \ (x_1, x_2) \in \Omega$
- $u(x_1, x_2, t = 0) = f(x_1, x_2), \qquad \left.\frac{\partial u}{\partial \vec{n}}\right|_{\partial\Omega} = 0$
- $f(x_1, x_2)$: Noisy image as initial condition
- This PDE in nonlinear, inhomogeneous, but isotropic

Inhomogeneous Nonlinear Diffusion Filtering

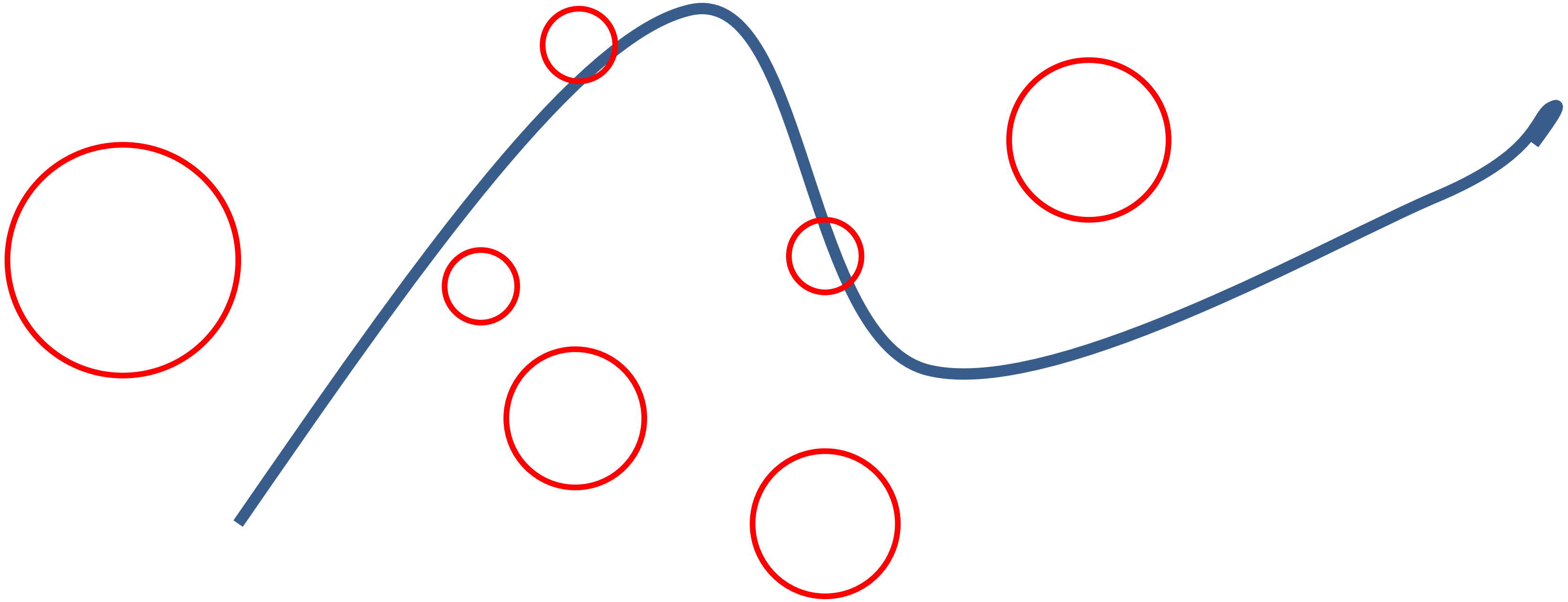
- An Edge-Aware diffusion filter (version #4)
- u is noisy in early steps, try smoothing (regularization):
- $\frac{\partial u}{\partial t} = \nabla.\left(g\left(\left|\nabla u_{\sigma(t)}\right|^2\right)\nabla u\right), \quad (x_1, x_2) \in \Omega$
- $u(x_1, x_2, t = 0) = f(x_1, x_2), \quad \left.\frac{\partial u}{\partial \vec{n}}\right|_{\partial\Omega} = 0$
- $f(x_1, x_2)$: Noisy image as initial condition
- where, $u_{\sigma(t)} = u * G_{\sigma(t)}$(Gaussian Smoothing), and $\sigma(t)$ is a decreasing function of time steps.

Inhomogeneous Nonlinear Diffusion Filtering

- An Edge-Aware diffusion filter (version #5)
- Add smoothing factor to equation
- $\frac{\partial u}{\partial t} = \nabla.\left(g\left(\left|\nabla u_{\sigma(t)}\right|^2\right)\nabla u\right) + \beta(f-u), \quad (x_1, x_2) \in \Omega$
- $u(x_1, x_2, t=0) = f(x_1, x_2), \quad \left.\frac{\partial u}{\partial \vec{n}}\right|_{\partial\Omega} = 0$
- $f(x_1, x_2)$: Noisy image as initial condition
- β : Adjust distance between initial, f, and final solution, u.

Inhomogeneous Nonlinear Diffusion Filtering

- How it works, weak smoothing near the edges



The End

› AnY QuEsTiOn?