Medical Image Analysis

CS 778 / 578

Computer Science and Electrical Engineering Dept.
West Virginia University

February 2, 2011

Outline

Outline

Anisotropic diffusion

The Weickert paper uses the notation

$$\partial_t u \equiv \frac{\partial u}{\partial t}$$

In this notation the anisotropic diffusion equation is written

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

where the diffusion coefficient, D, is a tensor (matrix).

Perona-Malik Implementation

$$
\begin{aligned}
\partial_t u &= \operatorname{div}\left(\begin{bmatrix} c_E u_x \\ c_N u_y \end{bmatrix} + \begin{bmatrix} c_W u_x \\ c_S u_y \end{bmatrix} \right) \\
&= \operatorname{div}\left(\begin{bmatrix} c_E & 0 \\ 0 & c_N \end{bmatrix} \nabla u + \begin{bmatrix} c_W & 0 \\ 0 & c_S \end{bmatrix} \nabla u \right)
\end{aligned}
$$

The implementation is anisotropic diffusion with a diagonal diffusion tensor.

The diffusion tensor D

From the statistical mechanics of diffusion, it is known that D is the covariance matrix of the molecular displacement probability density function: The probability that a molecule at position x_0 at time t will diffuse to $x_0 + r$ at time $t + \tau$

$$p_\tau(r) = N(0, 2\tau D)$$

So D must be

- Symmetric : $D = D^T$
- Non-negative-definite : $x^T Dx \geq 0$ for all x$\neq 0$

The diffusion tensor D

Physically, these constraints mean

- Symmetric : conservation of mass
- Non-negative-definite : no backward diffusion

$$\nabla u^T D \nabla u > 0$$

We can understand the geometry of anisotropic diffusion by looking at the eigenvalue decomposition of D.

$$\mathbf{D} = \mathbf{X} \Lambda \mathbf{X}^{-1}$$

The diffusion tensor D

$$\mathbf{D} = \mathbf{X}\Lambda\mathbf{X}^{-1}$$

- $X = [v1|v2]$, e are the eigenvectors of D

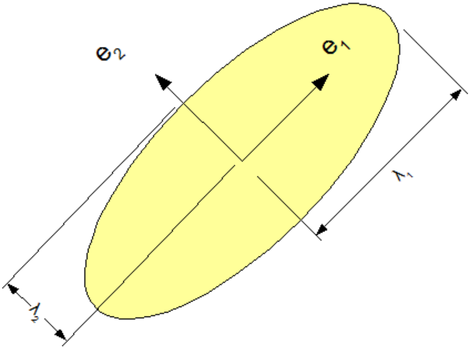
$$\Lambda = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

- λ_1, λ_2 are the eigenvalues of D
- Since D is real and symmetric, the eigenvalues are real
- Since D is symmetric, the eigenvectors are mutually orthogonal
- Since D is non-negative-definite, the eigenvalues ≥ 0

Eigenvalue Decomposition of D

- Since $\mathbf{X}$ is orthogonal $\mathbf{X}^{-1} = \mathbf{X}^T$
- $\mathbf{X}$ is a rotation matrix
- Λ is a nonuniform scaling matrix

$$\mathbf{D} = \mathbf{X}\Lambda\mathbf{X}^T$$



Eigenvalue Decomposition of D

- Since $\mathbf{X}$ is orthogonal $\mathbf{X}^{-1} = \mathbf{X}^T$
- $\mathbf{X}$ is a rotation matrix
- Λ is a nonuniform scaling matrix

$$\mathbf{D} = \mathbf{X}\Lambda\mathbf{X}^T$$

$$j = -\mathbf{D}\nabla u$$

- Rotate flux by multiplying with rotation matrix.
- Perform nonuniform scaling.
- Apply inverse rotation

Construct D

Construct $\mathbf{X}$ using regularized (smoothed) edge information.
Let $u_\sigma = K_\sigma * u$.

- $v1 \parallel \nabla u_\sigma$
- $v2 \perp v1$
- $||v1|| = ||v2|| = 1$
- $\mathbf{X} = [v1|v2]$

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

- $\lambda_1 = g(||\nabla u_\sigma||)$ the diffusivity from Perona-Malik
- $\lambda_2 = 1$

Construct D

When $g \approx 1$ (away from an edge), then $D \approx I$, and $j \approx -\nabla u$, (isotropic diffusion)

Near an edge $g \to 0$

$$D = \left[\begin{array}{c|c} v_1 & v_2 \end{array} \right] \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ \hline v_2 \end{bmatrix}$$

$$D\nabla u = \left[\begin{array}{c|c} v_1 & v_2 \end{array} \right] \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \left(\begin{bmatrix} v_1 \\ \hline v_2 \end{bmatrix} \nabla u \right)$$

Construct D

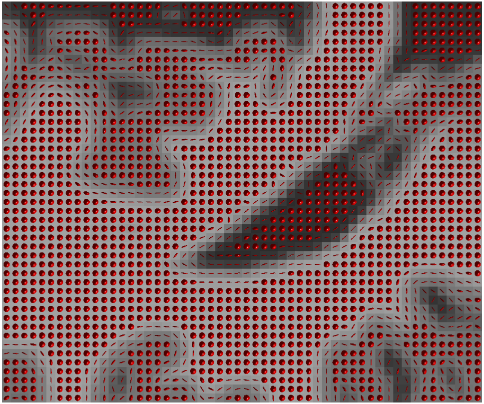
$$D\nabla u = \left[\begin{array}{c|c} v_1 & v_2 \end{array}\right] \left(\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_1 \cdot \nabla u \\ v_2 \cdot \nabla u \end{bmatrix} \right)$$

$$D\nabla u = \left[\begin{array}{c|c} v_1 & v_2 \end{array}\right] \begin{bmatrix} 0 \\ v_2 \cdot \nabla u \end{bmatrix}$$

$$D\nabla u = v_2 (v_2 \cdot \nabla u)$$

$D\nabla u$ is **parallel** to v_2, so it is **parallel** to the edge.

Tensor field for input image



Note that tensors are isotropic **within** homogeneous regions. Tensors are anisotropic and direct the flux **along** edges. The resulting diffusion process is edge enhancing.

Coherence enhancing diffusion : Structure matrix

Idea : Construct diffusion tensor, D, from image structure matrix, J.

$$J_\rho(\nabla u_\sigma) = K_\rho * (\nabla u_\sigma \nabla u_\sigma^T)$$

$$J_\rho(\nabla u_\sigma) = \begin{bmatrix} \overline{u_x^2} & \overline{u_x u_y} \\ \overline{u_x u_y} & \overline{u_y^2} \end{bmatrix}$$

Let v_1 and v_2 be the eigenvectors of J, and μ_1 and μ_2 be the corresponding eigenvalues.

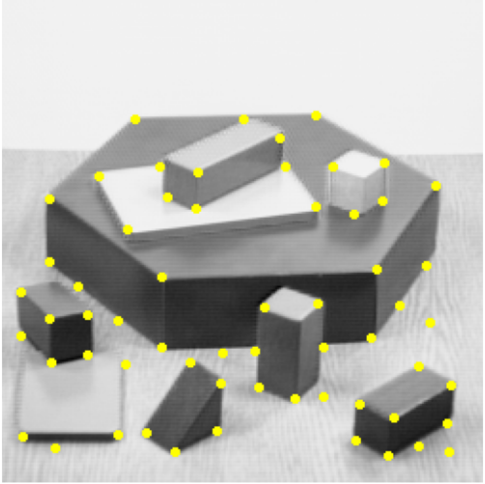
Structure matrix

Image structure can be deduced from the eigenvalue decomposition of J.

- Direction v_1 has the greatest intensity variation
- Direction v_2 has the least intensity variation

- μ_1, μ_2 give the degree of intensity variation (contrast) in the eigendirection.
- Constant-valued image $\rightarrow \mu_1 = \mu_2 = 0$
- Step edge $\rightarrow \mu_1 > 0, \mu_2 \approx 0$
- Corner $\rightarrow \mu_1 > 0, \mu_2 > 0$

In computer vision



D has the same eigenvectors as the structure matrix, but different eigenvalues (λ_1, λ_2)

$$\begin{aligned}\lambda_1 &= \alpha \\ \lambda_2 &= \begin{cases} \alpha & \text{if } \mu_1 = \mu_2 \\ \alpha + (1-\alpha)\exp\left(\frac{-C}{(\mu_1 - \mu_2)^{2m}}\right) & \text{otherwise} \end{cases}\end{aligned}$$

When μ_1, μ_2 are very different, smooth more in the coherence direction. (see paper for details on choosing *alpha*, *C*, *m*)

Outline

Variational Calculus

There is another way to arrive at the diffusion equation.

We wish to find u such that $E(u)$ is minimized, where $E(u)$ is a **functional** of the form

$$E(u) = \int_{\Omega} f(x, y, u, u_x, u_y)\, dx$$

A **functional** depends not just on variables (x,y), but also on functions and their derivatives u, u_x, u_y.

We will work with functionals by treating u, u_x, u_y as independent variables.

Variational Calculus

We will work with functionals by treating u, u_x, u_y as independent variables.
What this means:
When you see f_u : differentiate f with respect to u.
When you see f_{u_x} : differentiate f with respect to u_x.

Variational Calculus

When $E(u)$ is a **functional** of the form

$$E(u) = \int_{\Omega} f(x, y, u, u_x, u_y)\, dx$$

then, when $E(u)$ is minimized the following condition holds

$$\begin{aligned} \nabla E &= f_u - \frac{\partial}{\partial x} f_{u_x} - \frac{\partial}{\partial y} f_{u_y} = 0 \\ &= f_u - \operatorname{div}\begin{pmatrix} f_{u_x} \\ f_{u_y} \end{pmatrix} \end{aligned}$$

The minimum can be found using the descent method

$$\partial_t u = -f_u + \operatorname{div}\begin{pmatrix} f_{u_x} \\ f_{u_y} \end{pmatrix}$$

Variational Calculus

Membrane spline energy represents the stretching (arc-length) of a thin sheet. Minimizing this energy results in smooth surfaces.

The membrane spline energy has the form

$$E_{mem}(u) = \int_{\Omega} ||\nabla u||^2 \, dx = \int_{\Omega} u_x^2 + u_y^2 \, dx$$

Variational Calculus

Minimizing the membrane spline energy

$$E(u) = \int_{\Omega} ||\nabla u||^2 \, dx = \int_{\Omega} u_x^2 + u_y^2 \, dx$$

has Euler-Lagrange condition for minimization:

$$\begin{aligned} \nabla E &= f_u - \frac{\partial}{\partial x} f_{u_x} - \frac{\partial}{\partial y} f_{u_y} = 0 \\ &= 0 - \frac{\partial}{\partial x}(2u_x) - \frac{\partial}{\partial y}(2u_y) = 0 \end{aligned}$$

Leading to the evolution equation

$$\partial_t u = \text{div}(\nabla u)$$

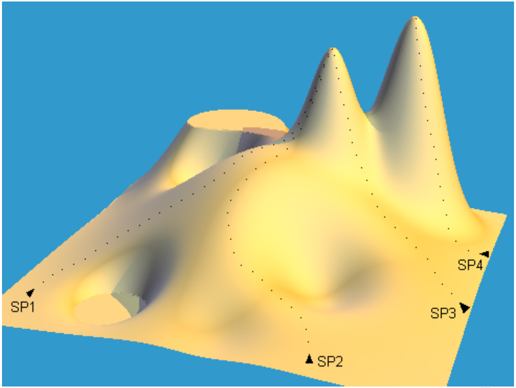
Gradient Descent

The evolution equation is a gradient descent equation.

The gradient is useful in this simple numerical optimization technique

$$\frac{dx}{dt} = \mp\nabla I \tag{1}$$

Will converge to a local minimum/maximum of I.



Data Constraints

Let u^0 be the original input image. We wish to penalize solutions which are far from the initial condition.
Find $\min_u E(u)$,

$$E(u) = \int_\Omega (u_x^2 + u_y^2 + \frac{\beta}{2}(u - u^o)^2)\, dx$$

results in the descent equation (a reaction-diffusion equation)

$$\partial_t u = \mathrm{div}(\nabla u) + \beta(u^0 - u)$$

The evolution reaches a nontrivial steady-state. (This is the reaction term proposed by Weickert in section 6.1)

With the data constraint we can directly solve for the steady-state solution. The steady-state equation

$$\partial_t u = \text{div}(\nabla u) + \beta(u^0 - u)$$

leads to a discretization of the form

$$0 = Au + \beta(u^0 - u)$$

which can be rearranged as

$$(A - \beta I)u = -\beta u^0$$

Note that when A is a function of time this method will still require iteration, but if A is constant we can get a solution in a single iteration.

Variational Calculus

Later we will look at a more general form of the variational calculus which will allow us to minimize other functionals, such as the thin-plate spline energy. This functional minimizes bending energy (curvature).

$$E_{TPS}(u) = \int_{\Omega} u_{xx}^2 + 2u_{xy}^2 + u_{yy}^2 \, dx$$