The Heat Equation

Imaging Lab 6 - May, 2017

Kamyar Nazeri
Student ID: 100633486

Anisotropic Diffusion: Color Images

According to heat equation, the heat dissipates with the following PDE:

$$u_t = \frac{\partial}{\partial x}(K u_x)$$

When $K(x)$ changes throughout the bar, then heat will flow unevenly and we call this anisotropic diffusion and the 2D PDE becomes:

$$u_t = \nabla \cdot (K \nabla u) = \frac{\partial}{\partial x}(K u_x) + \frac{\partial}{\partial y}(K u_y)$$

For conductivity function K we use Perona-Malik exponential edge-stopping function:

$$K = e^{-(\|\nabla u\|/a)^2}$$

Where

$$\|\nabla u\| = \sqrt{u_x^2 + u_y^2}$$

Temporal discretization of the PDE is:

$$u^{n+1} = u^n + \Delta t \, \left[\frac{\partial}{\partial x}(K u_x^n) + \frac{\partial}{\partial y}(K u_y^n)\right]$$

To apply anisotropic diffusion on a color image, we calculate the equation above on each of the 3 RGB channels and storing it in the corresponding channel of a new image. As the first step inside the for loop, we calculate the forward differences to approximate the first derivatives u_x and u_y:

$$\begin{aligned} u_x &\approx D_x^+ u = u(x+1, y) - u(x, y) \\ u_y &\approx D_y^+ u = u(x, y+1) - u(x, y) \end{aligned}$$

And to calculate $\frac{\partial}{\partial x}(K u_x)$ and $\frac{\partial}{\partial y}(K u_y)$ we perform forward then backward differences:

$$\begin{aligned} \frac{\partial}{\partial x}(K u_x) &\approx D_x^- (K D_x^+ u) \\ \frac{\partial}{\partial y}(K u_y) &\approx D_y^- (K D_y^+ u) \end{aligned}$$

Finally the PDE becomes:

$$u^{n+1} = u^n + \Delta t \left[D_x^- \left(e^{-(\|\nabla u^n\|/a)^2} D_x^+ u^n \right) + D_y^- \left(e^{-(\|\nabla u^n\|/a)^2} D_y^+ u^n \right) \right]$$

We use Forward Euler Method with Neumann boundary conditions to solve the above equation.

Coding Anisotropic Diffusion

Listing 1 shows the anisotropic diffusion applied on a color image in Matlab:

```
clear;
img = imread('peppers.png');

%Parameters
a = 25;                 % Edge-stopping
dt = 0.1;               % Time step
T = 20;                 % Stopping time
K = 0;                  % Conductivity
[m,n,c] = size(img);    % Image size

%Initialize: convert to double so we can do arithmetic.
res = double(img);

for t = 0:dt:T
    for c=1:3
        % read image channel (RGB)
        u = res(:,:,c);

        % u_x forward difference: u(x+1,y) - u
        u_x = u(:,[2:n,n]) - u;

        % u_y forward difference: u(x,y+1) - u
        u_y = u([2:m,m],:) - u;

        % norm of the gradient
        norm = sqrt(u_x.^2 + u_y.^2);

        % Perona-Malik exponential edge-stopping
        K = exp(-(norm./a).^2);

        % backward difference u_x(K.u_x)
        p_x = K .* u_x;
        p_x = p_x - p_x(:,[1,1:n-1]);

        % backward difference u_y(K.u_y)
        p_y = K .* u_y;
        p_y = p_y - p_y([1,1:m-1],:);

        % anisotropic forward euler
        res(:,:,c) = u + dt * (p_x + p_y);
    end
end

imshow(uint8(res));
```

Listing 1: Anisotropic Diffusion for Color Images in Matlab

Figure 1 shows a test image and two diffused images for different values of Edge-Stopping parameter. As shown below, anisotropic diffusion turns photos into cartoons and the value of a controls how diffusion will stop at the edges of the image:

Figure 1: Anisotropic Diffusion applied on a color image with different Edge-Stopping values.