ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Computer Graphics Lab
ML Group

Prof. M. Gross / Prof. J. Buhmann

Remo Ziegler, Christian Voegeli, Daniel Cotting, Christian Sigg, Jens Keuchel

Solve before: June 13, 2006

Visual Computing
(Non-)linear Diffusion
Solution

General Remarks

It is not necessary to hand in your results. You will find an exemplary solution on the lecture's web page.

Basics

For two dimensional diffusion processes, *Fick's law* states that a concentration gradient $\nabla u = (\partial_x u, \partial_y u)^\top$ (e.g. of mass/temperature) causes a flux $j = (j_x, j_y)^\top$ which compensates for this gradient ('equilibration of concentration differences'):

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

The relation between ∇u and j depends on some characteristics of the material and is specified by the positive diffusivity $\mathbf{D}$, which may vary depending on the local structure. In the following, we will only consider the so-called *isotropic* case by assuming that j and ∇u are parallel, which results in a scalar-valued diffusivity $\mathbf{D} \in \mathbb{R}^+$. (In the more general *anisotropic* case $\mathbf{D}$ is a matrix.)

The flux results in a temporal change (the *diffusion*) of the concentration u during which no mass is destroyed or created, which is expressed by the *continuity equation*:

$$\partial_t u = -\mathrm{div} j = -(\partial_x j_x + \partial_y j_y) \ ,$$

where t denotes time. Combining both equations gives the important *diffusion equation*:

$$\partial_t u = \mathrm{div}(\mathbf{D} \cdot \nabla u) \ .$$

In image processing, the concentration u corresponds to image gray values. Depending on whether the diffusivity $\mathbf{D}$ is adapted to local image structure, we distingish between *linear* and *nonlinear* isotropic diffusion filtering.

1) Linear isotropic diffusion

Consider the case where the diffusivity is constant and normalized to one, $\mathbf{D} = 1$:

$$\partial_t u = \mathrm{div}(\nabla u) = \Delta u = \partial_{xx} u + \partial_{yy} u \ .$$

a) As is shown in the lecture, the unique solution of this differential equation (under the boundary condition that u corresponds to the given image f for $t = 0$) is a convolution of f with a Gaussian. How is the standard deviation σ related to the diffusion time?

Solution: The image obtained after time t corresponds to convolving the original image f with a Gaussian with standard deviation $\sigma = \sqrt{2t}$.

b) In order to numerically implement the diffusion process, let u be discretized in space with step size h and in time with step size τ: $u_{ij}^{(n)} := u(hi, hj, n\tau)$. Approximate the derivatives in the diffusion equation by finite difference operators (Taylor's Formula), using 'forward differences' for the time derivatives $(\partial_t u)_{ij}^{(n)}$, and 'central differences' (involving the four nearest neighbors of $u_{ij}^{(n)}$) for the spatial derivatives $(\partial_{xx} u)_{ij}^{(n)}$ and $(\partial_{yy} u)_{ij}^{(n)}$.

Solution: Taylor's Formula:

$$u(t_0 + \tau) \approx u(t_0) + \partial_t u(t_0) \cdot \tau \ .$$

Finite difference approximation of first derivative in time (forward difference):

$$(\partial_t u)_{ij}^{(n)} \approx \frac{u_{ij}^{(n+1)} - u_{ij}^{(n)}}{\tau}$$

Finite difference approximation of second derivatives in space (twice central difference with step size $\frac{h}{2}$):

$$\begin{aligned}(\partial_{xx} u)_{ij}^{(n)} &\approx \frac{(\partial_x u)_{i+\frac{1}{2},j}^{(n)} - (\partial_x u)_{i-\frac{1}{2},j}^{(n)}}{(i+\frac{1}{2})h - (i-\frac{1}{2})h} \\ &\approx \frac{1}{h}\left(\frac{u_{i+1,j}^{(n)} - u_{ij}^{(n)}}{h} - \frac{u_{ij}^{(n)} - u_{i-1,j}^{(n)}}{h}\right) = \frac{u_{i+1,j}^{(n)} - 2u_{ij}^{(n)} + u_{i-1,j}^{(n)}}{h^2}\end{aligned}$$

and likewise

$$(\partial_{yy} u)_{ij}^{(n)} \approx \frac{u_{i,j+1}^{(n)} - 2u_{ij}^{(n)} + u_{i,j-1}^{(n)}}{h^2} \ .$$

c) Derive the recursive definition of the diffusion process by computing $u_{ij}^{(n+1)}$ from $u^{(n)}$.

Solution: Substituting derivatives into the diffusion equation yields:

$$\begin{aligned} & \partial_t u = \partial_{xx} u + \partial_{yy} u \\ \Rightarrow \quad & \frac{u_{ij}^{(n+1)} - u_{ij}^{(n)}}{\tau} = \frac{1}{h^2}(u_{i+1,j}^{(n)} + u_{i-1,j}^{(n)} + u_{i,j+1}^{(n)} + u_{i,j-1}^{(n)} - 4u_{ij}^{(n)}) \\ \Rightarrow \quad & u_{ij}^{(n+1)} = \left(1 - \frac{4\tau}{h^2}\right) u_{ij}^{(n)} + \frac{\tau}{h^2}\left(u_{i+1,j}^{(n)} + u_{i-1,j}^{(n)} + u_{i,j+1}^{(n)} + u_{i,j-1}^{(n)}\right) \ . \end{aligned}$$

d) Which time step sizes τ lead to a stable filtering process? What kind of filter is obtained in this case?

Solution: The filtering process is unstable if the first weight $1 - \frac{4\tau}{h^2}$ is negative, as this may cause the $u_{ij}^{(n)}$-values to become arbitrarily large. On the other hand, for nonnegative weights, i.e. $\tau \leq h^2/4$, we have a stable convex combination of the u-values:

$$\max_{i,j} \left| u_{ij}^{(n+1)} \right| = ||u^{(n+1)}||_\infty \leq \|u^{(n)}\|_\infty \ .$$

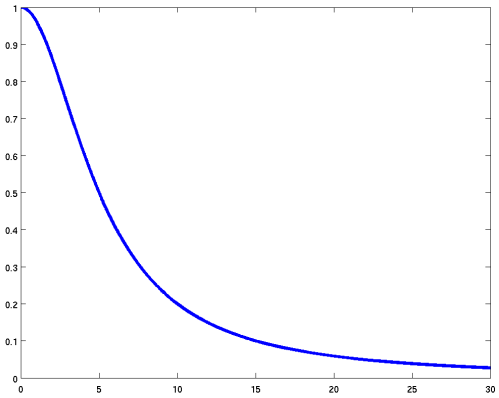
In this case the filtering process averages neighboring values, which results in a linear low pass filter. This approves that linear diffusion is equivalent to convolution with a Gaussian!

2) Nonlinear isotropic diffusion

Linear isotropic diffusion processes blur edges. In order to prevent this, the diffusivity can be adapted so that it decreases close to edges and is large only in homogeneous regions. For this purpose, let's assume that the diffusivity is a function of the magnitude of the gradient $|\nabla u|$. In the following, consider the one-dimensional case, where $\nabla u = \partial_x u$, which results in the diffusion equation

$$\partial_t u = \partial_x(\mathbf{D}(|\partial_x u|)\partial_x u).$$

a) Sketch the shape of the diffusivity function $\mathbf{D}(|\partial_x u|)$ that models the desired behavior.
Solution:



Diffusivity $\mathbf{D}(|\partial_x u|)$

b) Perona and Malik have suggested the diffusivity function

$$\mathbf{D}(|\nabla u|) = \frac{1}{1 + |\nabla u|^2/\lambda^2} \ ,$$

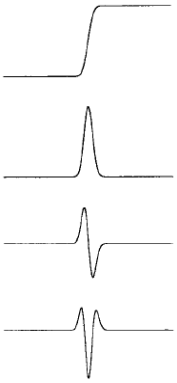
where λ is a contrast parameter. Derive a formula for this case (in 1D) that describes the variation in time of an edge, which is given by $\partial_t \, \partial_x u$.
Hint: Substitute the mathematical flux function $\Phi(\partial_x u) = -j = \mathbf{D}(|\partial_x u|) \cdot \partial_x u$, and differentiate using Φ.
Solution: The nonlinear diffusion equation simplifies to:

$$\partial_t u = \partial_x(\Phi(\partial_x u)) = \Phi'(\partial_x u)\partial_{xx} u \ .$$

As $\mathbf{D}(|\partial_x u|) > 0$*, we get*

$$\begin{aligned} \partial_t \, \partial_x u &= \partial_x \, \partial_t u \\ &= \partial_x \left[\Phi'(\partial_x u)\partial_{xx} u\right] \\ &= \Phi''(\partial_x u)\partial_{xx} u \cdot \partial_{xx} u + \Phi'(\partial_x u)\partial_{xxx} u \\ &= \Phi''(\partial_x u)(\partial_{xx} u)^2 + \Phi'(\partial_x u)\partial_{xxx} u. \end{aligned} \tag{1}$$

c) Suppose that a step edge is oriented in such a way that $\partial_x u > 0$. Sketch u and its first three derivatives. How do points close to the edge vary in time in comparison to the first derivative of the flux $\Phi'(\partial_x u)$?
Solution:



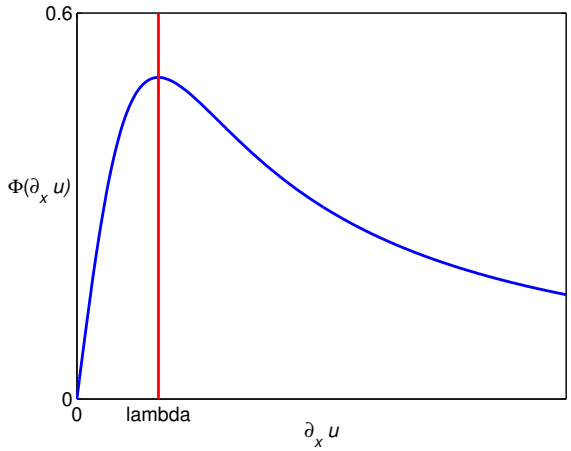
A step edge u and its 1st, 2nd, and 3rd derivatives

As the sketches show we have $\partial_{xx} u = 0$ and $\partial_{xxx} u \ll 0$ at the point of inflection of u since it corresponds to the point with maximum slope of u. Using equation (1) *we see that in a neighborhood of the point of inflection $\partial_t \, \partial_x u$ has sign opposite to $\Phi'(\partial_x u)$. Therefore, if $\Phi'(\partial_x u) > 0$ the slope of the edge will decrease with time; otherwise it will increase and the edge becomes sharper.*

d) Using the results of the previous questions, describe which edges are smoothed and which are enhanced by this nonlinear diffusion process.

Hint: Sketch the mathematical flux function Φ for the Perona-Malik diffusivity and observe the behavior of $\Phi'(\partial_x u)$.

Solution:



Plot of the flux function Φ

Now consider the plot of the flux function Φ in the special case of the given Perona-Malik diffusivity (Fig. 2). The plot shows that for $|\partial_x u| \leq \lambda$, we have $\Phi'(\partial_x u) \geq 0$, which results in smoothing 'weak' edges. For $|\partial_x u| > \lambda$ the derivative of the flux is negative, so that 'stronger' edges will be enhanced.

3) Diffusion in Matlab

To investigate the effects of diffusion for image filtering, we use functions from the nonlinear diffusion toolbox, which is available on the course web page.

a) The toolbox contains several demos for nonlinear diffusion that are based on a diffusivity function

$$\mathbf{D}(|\nabla u|) = \begin{cases} 1 - \exp\left(-\frac{c_m}{(|\nabla u|/\lambda)^m}\right), & \nabla u > 0 \\ 1, & \nabla u \leq 0 \end{cases}$$

where the constant c_m is chosen such that the flux is monotonously increasing for $\nabla u \leq \lambda$ ($c_m = 3.31488$ for $m = 8$). Basically, this diffusivity has the same properties as the Perona-Malik diffusivity discussed in the previous question.

Try out the demos nldif1demo.m, nldifdemo1.m, nldifdemo2.m and nldifdemo3.m to explore the capabilities of nonlinear diffusion.

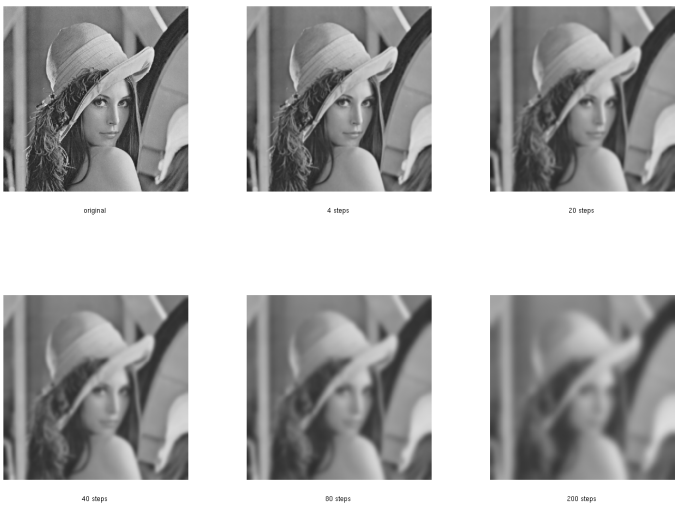
b) The command 'ldif' applies linear diffusion to an image with the spatial step size being fixed to 1. Fixing the temporal step size to 0.25, linear diffusion is applied for `nsteps` time steps in the following way:

```
t_stepsize = 0.25; nsteps = 10;
ima_ldif = ldif(ima,t_stepsize,nsteps);
```

Using the Lenna image from the previous exercise sheet, apply linear diffusion for 4,20,40,80,200 steps, and display the results. Can you obtain the same results with a Gaussian low pass filter?

Solution:

```
t_stepsize = 0.25;
nsteps = [4,20,40,80,200];
figure;
subplot(2,3,1); imshow(ima); xlabel('original');
for i=1:5
  ima_ldif = ldif(ima,t_stepsize,nsteps(i));
  subplot(2,3,i+1); imshow(ima_ldif); xlabel([int2str(nsteps(i)) ' steps']);
end
```



The same results are obtained with a Gaussian filter by setting $\sigma = \sqrt{(2T)}$*, where* $T =$`nsteps*t_stepsize`*. For example,*

```
h=fspecial('gaussian',50,sqrt(40));
imaf=imfilter(ima,h,'replicate');
```

produces the same result as linear diffusion for 80 steps with time step size 0.25. (In fact, linear diffusion is implemented here by applying a Gaussian filter multiple times!)

c) In order to apply nonlinear diffusion with the Perona-Malik diffusivity, you can use the command 'pmdif':

```
lambda = 0.05;
t_stepsize = 0.25; nsteps = 10;
ima_pmdif = pmdif(ima,lambda,0,t_stepsize,nsteps);
```

Additionally to the time step size and the number of steps, you need to specify the contrast parameter λ which specifies the minimum strength of an edge to be sharpened.

Using a fixed step size of 0.25, apply Perona-Malik nonlinear diffusion to the Lenna image for different contrast parameters $\lambda = 0.01, 0.02, 0.05, 0.1, 0.2$ for 20 steps, and display the results. How does an increasing λ-value influence the result?

Solution:

```
t_stepsize = 0.25; nsteps = 20;
lambda = [0.01,0.02,0.05,0.1,0.2];
figure;
subplot(2,3,1); imshow(ima); xlabel('original');
for i=1:5
  ima_pmdif = pmdif(ima,lambda(i),0,t_stepsize,nsteps);
  subplot(2,3,i+1); imshow(ima_pmdif); xlabel(['lambda = ' num2str(lambda(i))]);
end
```

For small λ-values, the image is only smoothed in homogeneous regions, while edges are enhanced. For increasing λ, only strong edges are enhanced, and the image is smoothed more.

d) Now keep $\lambda = 0.05$ fixed, and apply nonlinear diffusion for 5,10,20,50,100 steps. How does the image change?

Solution:

```
t_stepsize = 0.25;
nsteps = [5,10,20,50,100];
lambda = 0.05;
```

```
figure;
subplot(2,3,1); imshow(ima); xlabel('original');
for i=1:5
  ima_pmdif = pmdif(ima,lambda,0,t_stepsize,nsteps(i));
  subplot(2,3,i+1); imshow(ima_pmdif); xlabel([num2str(nsteps(i)) ' steps']);
end
```

With increasing number of steps, the image is smoothed more, but strong edges are maintained.

e) Add Gaussian noise with $\sigma = 0.02$ to the image, and apply nonlinear diffusion with $\lambda = 0.05$ for 10 steps. How do you explain the result?

Solution:

```
sigma = 0.02;
ima_noise = imnoise(ima,'gaussian',0,sigma);
figure;
subplot(1,2,1); imshow(ima_noise);
t_stepsize = 0.25; nsteps = 10;
lambda = 0.05;
ima_pmdif = pmdif(ima_noise,lambda,0,t_stepsize,nsteps);
subplot(1,2,2); imshow(ima_pmdif);
```

The Perona-Malik diffusion filter misinterprets some of the noise as edges and tries to preserve it.

f) Apply nonlinear diffusion for step sizes 0.5 and 1. How do you explain the results?

Solution:

```
t_stepsize = [0.5,1]; nsteps = 10;
lambda = 0.05;
figure;
for i=1:2
  ima_pmdif = pmdif(ima,lambda,0,t_stepsize(i),nsteps);
  subplot(1,2,i); imshow(ima_pmdif);
  xlabel(['step size ' num2str(t_stepsize(i))]);
end
```

step size 0.5

step size 1

For step sizes larger than 0.25, the diffusion process becomes numerically instable (cf. exercise 1), which results in the noise like artefacts.