Denoise By Diffusion Equation

HSU-HSIEN CHU

January 22, 2008

Introduction of this Project

This semester, we had learned some numerical scheme for ODE and PDE equations. In this project, i will choice some 1-Dim scheme and extand them to 2-D for solving diffusion equation on images.
After several times of smoothing, the edge of graph will also be smoothed. The image becomes foggy. So, i use some skills to maintain the edge.

A general 2-D diffusion equation

$$u_t = a\Delta u = a(u_{xx} + u_{yy})$$

For different a, there will be different level of diffuse.

Forward Euler Scheme

$$u_t = a(u_{xx} + u_{yy})$$

and

$$u_{xx} = \frac{U^n_{i+1,j} + U^n_{i-1,j} - 2U^n_{i,j}}{h^2}$$

$$u_{yy} = \frac{U^n_{i,j+1} + U^n_{i,j-1} - 2U^n_{i,j}}{h^2}$$

$$u_t = \frac{U^{n+1}_{i,j} - U^n_{i,j}}{k}$$

we have:

$$U_{i,j}^{n+1} = U_{i,j}^{n} + a \times r \times (U_{i+1,j}^{n} + U_{i-1,j}^{n} + U_{i,j+1}^{n} + U_{i,j-1}^{n} - 4U_{i,j}^{n})$$

where $r = \frac{k}{h^2}$

ADI Scheme

$$U^*_{i,j} = U^n_{ij} + \frac{k}{2}(D_y^2 U^n_{i,j} + D_x^2 U^*_{i,j})$$

$$U^{n+1}_{i,j} = U^*_{ij} + \frac{k}{2}(D_y^2 U^{n+1}_{i,j} + D_x^2 U^*_{i,j})$$

Convert to (do one direction first then another):

$$(I - \frac{k}{2}D_x^2)U^* = (I + \frac{k}{2}D_y^2)U^n$$

$$(I - \frac{k}{2}D_y^2)U^{n+1} = (I + \frac{k}{2}D_x^2)U^*$$

Smooth the Whole Image:Uniform diffuse

Use repeated boundary condition and the origion image as initial value and apply the schemes, we get:

Origional Image

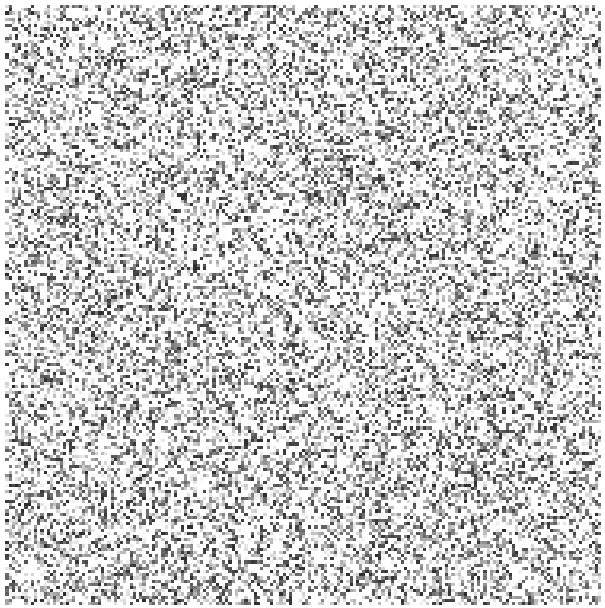
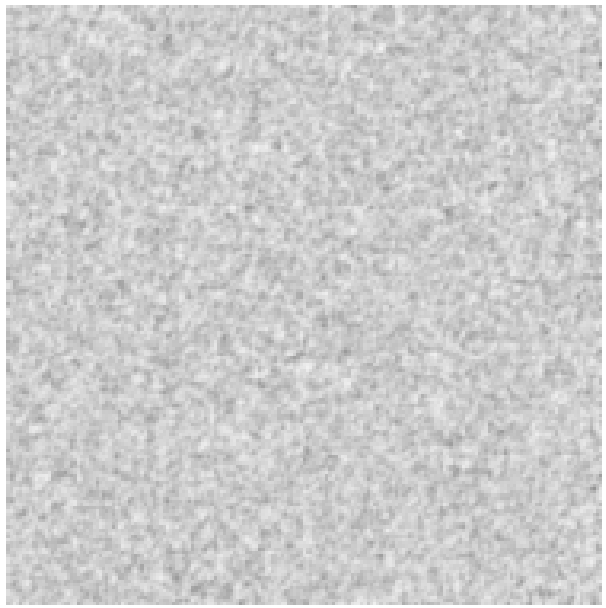
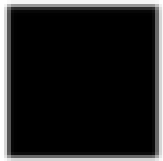


Image After Smooth
2.png mode=1 a=1.000 times=3 r=0.200 d=0



Origional Image

Image After Smooth
1.png mode=1 a=1.000 times=3 r=0.200 d=0



Origional Image

Image After Smooth
2.jpg mode=1 a=1.000 times=3 r=0.200 d=0

Skills for Maintain the Edge

Detect edge of graph by ∇u, and modify equation to:

$$\frac{\partial u}{\partial t} = a\nabla \cdot \frac{\nabla u}{|\nabla u|}$$

Diffuse less on edge.
To avoid the singular point of $\frac{1}{|\nabla u|}$, replece it by $\frac{1}{\sqrt{|\nabla u|^2+1}}$. So, i use this for so called 'denoise'. [1]

[1] Ref:The Digital TV Filter and Nonlinear Denoising

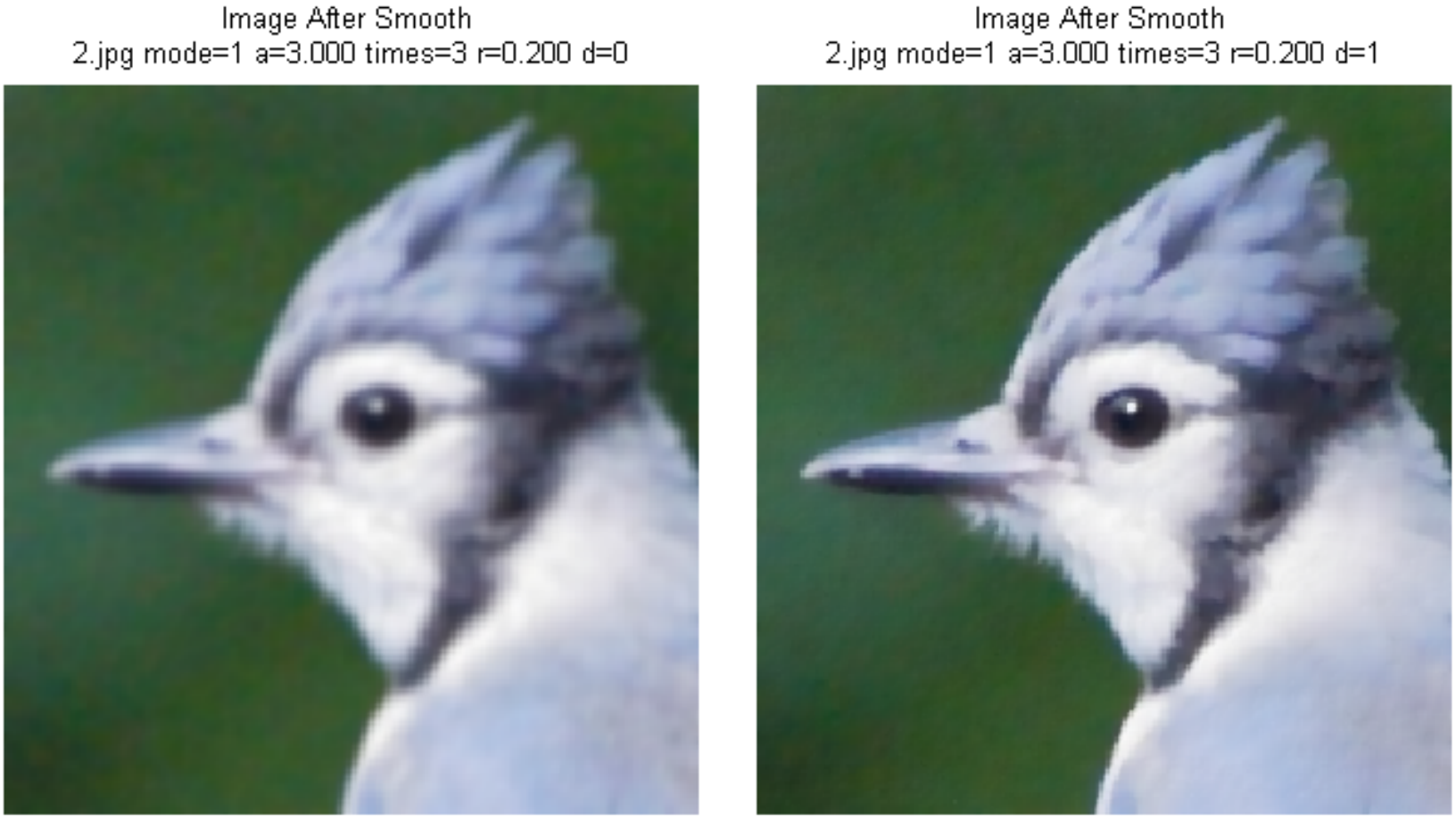


Figure: Comparison of uniform diffuse(left) and denoise(right).

MATLAB functions

- imload(filename): Load image file into matrix.
- imshow(IM): show matrix IM into image.
- imnoise(IM,TYPE,...): add noises into image matrix IM(I didn't used, but it's useful for you to noise you picture for denoising.
- imwrite(IM,filename): save image(I didn't used, but you can add it).

MATLAB functions

- imload(filename): Load image file into matrix.
- imshow(IM): show matrix IM into image.
- imnoise(IM,TYPE,...): add noises into image matrix IM(I didn't used, but it's useful for you to noise you picture for denoising.
- imwrite(IM,filename): save image(I didn't used, but you can add it).

MATLAB functions

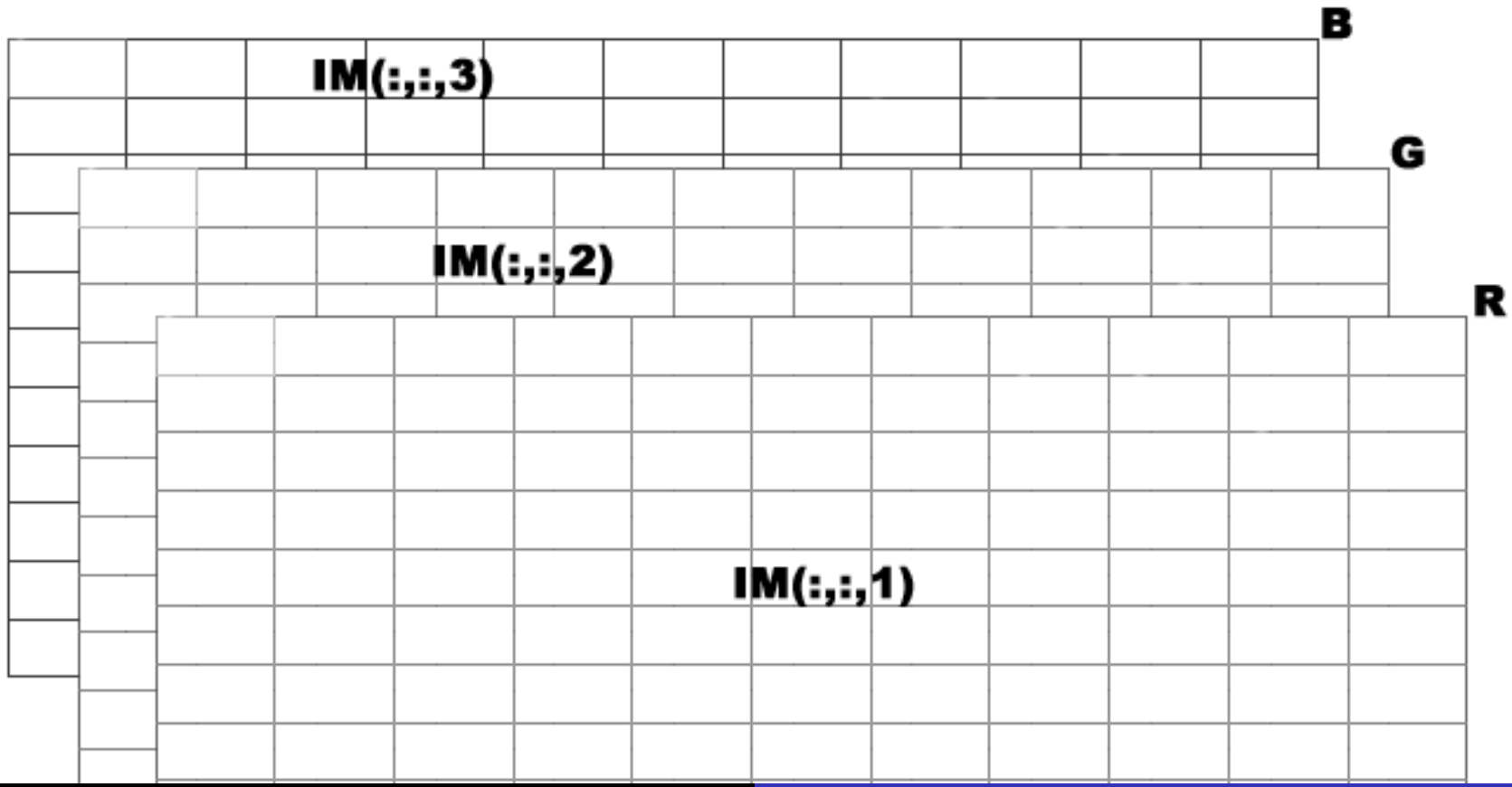
- imload(filename): Load image file into matrix.
- imshow(IM): show matrix IM into image.
- imnoise(IM,TYPE,...): add noises into image matrix IM(I didn't used, but it's useful for you to noise you picture for denoising.
- imwrite(IM,filename): save image(I didn't used, but you can add it).

MATLAB functions

- imload(filename): Load image file into matrix.
- imshow(IM): show matrix IM into image.
- imnoise(IM,TYPE,...): add noises into image matrix IM(I didn't used, but it's useful for you to noise you picture for denoising.
- imwrite(IM,filename): save image(I didn't used, but you can add it).

Image Data in Matrix

Three-dimensional (m-by-n-by-3) array of integers in the range [0, 255] (uint8).



Parameters

smooth(input_name, mode ,a ,times, d, r)

- input_name: filename of image.
- mode: whick scheme:
  - 0:up down left right 4-point average
  - 1:Forward Euler
  - 2:Forward Euler with midpoint on time
  - 3:ADI
- a: diffuse constant, recommend 1 to 3.
- times: denoise how many times.
- d: uniform diffuse(d=0) or denoise(d=1, only work for mode 1 and 2).
- r: $\frac{k}{h^2}$ for mode 1 and 2.

Parameters

smooth(input_name, mode ,a ,times, d, r)

- input_name: filename of image.
- mode: whick scheme:
  - 0:up down left right 4-point average
  - 1:Forward Euler
  - 2:Forward Euler with midpoint on time
  - 3:ADI
- a: diffuse constant, recommend 1 to 3.
- times: denoise how many times.
- d: uniform diffuse(d=0) or denoise(d=1, only work for mode 1 and 2).
- r: $\frac{k}{h^2}$ for mode 1 and 2.

Parameters

smooth(input_name, mode ,a ,times, d, r)

- input_name: filename of image.
- mode: whick scheme:
  - 0:up down left right 4-point average
  - 1:Forward Euler
  - 2:Forward Euler with midpoint on time
  - 3:ADI
- a: diffuse constant, recommend 1 to 3.
- times: denoise how many times.
- d: uniform diffuse(d=0) or denoise(d=1, only work for mode 1 and 2).
- r: $\frac{k}{h^2}$ for mode 1 and 2.

Parameters

smooth(input_name, mode ,a ,times, d, r)

- input_name: filename of image.
- mode: whick scheme:
  - 0:up down left right 4-point average
  - 1:Forward Euler
  - 2:Forward Euler with midpoint on time
  - 3:ADI
- a: diffuse constant, recommend 1 to 3.
- times: denoise how many times.
- d: uniform diffuse(d=0) or denoise(d=1, only work for mode 1 and 2).
- r: $\frac{k}{h^2}$ for mode 1 and 2.

Parameters

smooth(input_name, mode ,a ,times, d, r)

- input_name: filename of image.
- mode: whick scheme:
  - 0:up down left right 4-point average
  - 1:Forward Euler
  - 2:Forward Euler with midpoint on time
  - 3:ADI
- a: diffuse constant, recommend 1 to 3.
- times: denoise how many times.
- d: uniform diffuse(d=0) or denoise(d=1, only work for mode 1 and 2).
- r: $\frac{k}{h^2}$ for mode 1 and 2.

Parameters

smooth(input_name, mode ,a ,times, d, r)

- input_name: filename of image.
- mode: whick scheme:
  - 0:up down left right 4-point average
  - 1:Forward Euler
  - 2:Forward Euler with midpoint on time
  - 3:ADI
- a: diffuse constant, recommend 1 to 3.
- times: denoise how many times.
- d: uniform diffuse(d=0) or denoise(d=1, only work for mode 1 and 2).
- r: $\frac{k}{h^2}$ for mode 1 and 2.

Examples

Recommand parameters for example:

Example	mode	a	times	d	r
1.jpg	1	2	10	1	0.2
2.jpg	1	2	1	1	0.2
3.jpg	1	3	15	1	0.15
4.jpg	1	3	6	1	0.15

Origional Image

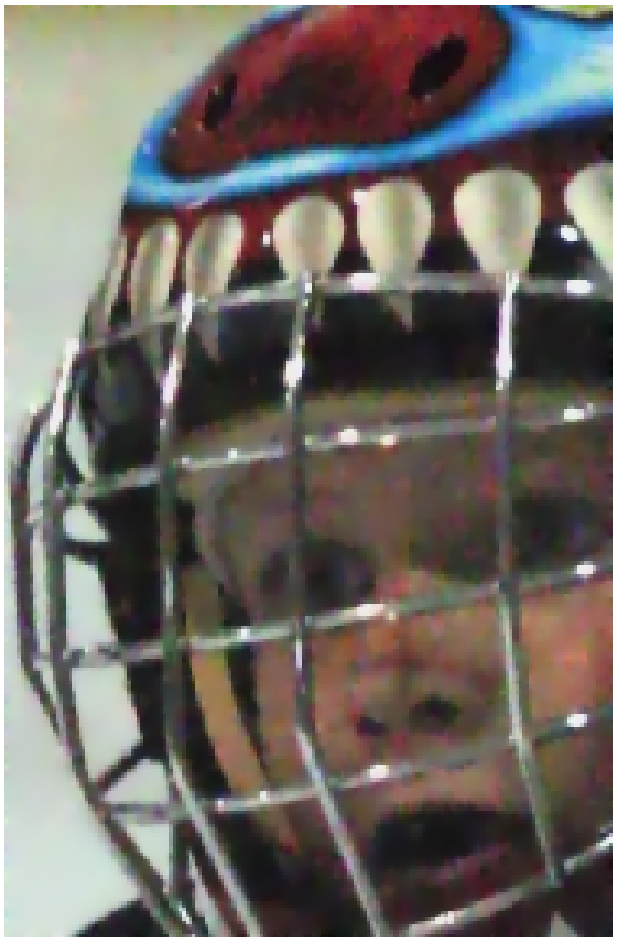
Image After Smooth
1.jpg mode=1 a=2.000 times=10 r=0.200 d=1

Origional Image

Image After Smooth
2.jpg mode=1 a=2.000 times=1 r=0.200 d=1

Origional Image

Image After Smooth
3.jpg mode=1 a=3.000 times=15 r=0.150 d=1



Origional Image

Image After Smooth
4.jpg mode=1 a=3.000 times=6 r=0.150 d=1

Refference


Rafael C. Gonzalez , Richard E. Woods , Steven L. Eddins
Digital Image Processing Using MATLAB.
ISBN:957-48-336-4.


Tony F. Chan, *Member,IEEE , Stanley Osher, and Jianhong Shen*
The Digital TV Filter and Nonlinear Denoising.
IEEE TRANSACTIONS ON IMAGE PROESSING, *VOL. 10,NO. 2,FEBRUARY 2001.*


Joachim Weickert
Anisotropic Diffusion in Image Processing.
Vollzg der Promotion*: 29. 01. 1996 D386*

Any Question?

Thanks for your attention!!