TP3: Image Smoothing

March 27, 2009

1 Introduction

Image smoothing as an important preprocessing step in image operations like feature extraction and image sampling since it suppresses high frequencies. In this TP, both linear and nonlinear methods to smooth an image are going to be studied.

2 Linear Smoothing

One of the first proposed methods for image smoothing is the convolution with a Gaussian Kernel.

$$u_{t+1} = G_\sigma * u_t. \tag{1}$$

Smoothing with the Gaussian is not really appropriate for the case of the feature extraction as it is not well localized and not contrast invariant. Though, it manages to suppress well the high frequencies. One interesting property of the Gaussian is that it is the fundamental solution of the heat equation.

$$\frac{\partial u}{\partial t} = \Delta u. \tag{2}$$

One important drawback of the Gaussian smoothing lies in the fact that it is an isotropic operator. Thus, it acts equally to all directions and while it smoothes homogeneous zones, it also smoothes the edges. In order to tackle this drawback a number of nonlinear smoothing methods that are more directional have been proposed.

3 Anisotropic Smoothing

The first nonlinear method to perform image smoothing was proposed by Perona and Malik [1]. The main idea of the proposed method is to try to perform smoothing in the areas that are homogeneous enough while, at the same time, try to enhance the boundaries. The performance of the algorithm along different areas is defined by the magnitude of the gradient $|Du|$ at the area. In areas where the $|Du|$ is small, a heat-like diffusion is performed while in areas where the $|Du|$ is large an inverse heat diffusion takes place. The divergence form of the Perona-Malik is the following:

$$\frac{\partial u}{\partial t} = div(g(|Du|)Du) \tag{3}$$

where $g(s) = \exp(-\frac{1}{K}s)$ and K is a parameter that influences the performance of the algorithm with respect to $|Du|$.

4 Total Variation Flow

Another important nonlinear method to denoise an image was proposed by Rudin, Osher and Fatemi [2]. In [2], the authors propose to minimize the BV norm of the image, $\int |Du(\mathbf{x})|dx$, in order to restore the image. The gradient descent for this specific objective function can be written as:

$$\frac{\partial u}{\partial t} = K * div(\frac{Du}{|Du|}). \tag{4}$$

This methods has a result a diffusion in a direction that it orthogonal to the gradient.

5 Work

5.1 Work to do

You should send as back a report containing the code you wrote, the results you obtained (stating explicitly what parameters you used) and your observations. You should perform your experiments in the corrupted by noise image that is provided to you. The report should be handed back by Monday 06/04. Afterwards, the solutions are going to be available and no report will be accepted.

5.2 Heat Diffusion

You should verify the equivalence between the heat diffusion equation and the convolution with a Gaussian. What you should do is convolve an image with a Gaussian Kernel of standard deviation equal to 2 and try to find the number of iterations that the heat diffusion equation should be applied to the same image to have an equivalent result. Be careful with the time step that you use, it should be small (i.e 0.2).

5.3 Anisotropic Diffusion

You should implement the anisotropic diffusion scheme as it is previously presented. Your function should take as input the original image, the number of the iterations as well as the parameter K. It is suggested that you follow the code outline provided.

All your experiments should be performed to the noise degraded image provided. You should experiment with different values of the parameter K, as well as different number of iterations. A function that will plot the profile of the intensity values along a line is given to you. It should be used in order to evaluate the results obtained and draw conclusions about the performance of the method. Eventually, the method should be compared to the simple Gaussian convolution.

5.4 Total Variation Flow

The Total Variation Flow should also be implemented. The created function should take as an input the original image, the number of the iterations as well as the parameter K. You should perform experiments in a similar way to the case of anisotropic diffusion.

5.5 Bonus

A degraded image whose parts are missing is given to you. Using the methods that you have implemented, you should try to complete the missing information (impainting).

6 Provided code

Two files will be provided. The first will describe the skeleton of the main function in seven steps:

- laod an image from file,
- choose a method,
- run this method,
- display image,
- display intensity(ies) profile(s), (*NB: close the window "Intensity profile" if you want to continue*)
- save an image in a file.

It is strongly recommended to use this scheme and the prototype of the 3 functions for the solution provided with the report.

```
1) GaussianFilter(_out, _in, _sigma, _radius)
2) AnisotropicDiffusion(_out, _in, _nbIt, _paramK)
3) TotalVariationFlow(_out, _in, _nbIt, _paramK)
```

The second file contain the class CIntensityProfile that will manage the display of intensity(ies) profile(s) in a standard way. Each profile (max 3) is displayed in a one image composed of graphics with different color (red, green, blue). The y coordinate will represent the intensity of the image at a abscissa along the specified line. This line can be a row or a column of the image. The display is performed by the function:

```
visualizeIntensityProfile()
```

This method has 3 different prototypes :

```
1) visualizeIntensityProfile(imgRed,indice,dim)
2) visualizeIntensityProfile(imgRed,imgGreen,indice,dim)
3) visualizeIntensityProfile(imgRed,imgGreen,imgBlue,indice,dim)
```

If you want display one profile in red use 1) and if you want display two profiles (first in red and second in green) use 2). The last method is able to display 3 profiles with RGB colors. Parameter *indice* must be used to indicate the line that you want display. Parameter *dim* indicate if this line is a row or a column.

References

[1] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990.

[2] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Phys. D*, 60(1-4):259–268, 1992.