UNIVERSITY of LIMERICK

OLLSCOIL LUIMNIGH

Perona-Malik Anisotropic Filtering

Machine Vision – RE4107
Dr. Colin Flanagan

Group Members:

Name	**ID**
Cian Conway	*10126767*
Patrick Stapleton	*10122834*
Ivan McCaffrey	*10098119*

Introduction

Perona-Malik diffusion or anisotropic diffusion is a computer vision filtering technique that aims at reducing image noise without reducing the image's quality in the process. This diffusion technique typically resembles the process that creates a scale space, where an image generates a parameterized family of successively more and more blurred images based on a diffusion process. Each of the resulting images in this filter is given as a convolution between the image and a 2D isotropic Gaussian filter. This diffusion is a linear and space invariant transformation of the original image. Anisotropic diffusion is a much generalized version of the diffusion process. It produces a family of parameterized images where each resulting image is a combination between a filter that depends on the content of the original image and the original image itself. This means that anisotropic diffusion is a linear and space invariant transformation of the original image.

Algorithm Properties

Let $\Omega \subset R^2$ denote a subset of the plane and $I(\cdot, t) \rightarrow R$ be a family of greyscale images. With this, anisotropic diffusion can be defined as:

$$\frac{\partial I}{\partial t} = \operatorname{div}\left(c(x,y,t)\nabla I\right) = \nabla c \cdot \nabla I + c(x,y,t)\Delta I$$

Where, Δ denotes Laplacian, ∇ denotes Gradient, *div(...)* is the divergence operator and *c(x,y,t)* is the diffusion coefficient. *c(x,y,t)* also controls the rate of the diffusion. When Perona and Malik pioneered the idea they proposed two functions for the diffusion coefficient:

$$c\left(\|\nabla I\|\right) = e^{-\left(\|\nabla I\|/K\right)^2}$$

and

$$c\left(\|\nabla I\|\right) = \frac{1}{1 + \left(\frac{\|\nabla I\|}{K}\right)^2}$$

The constant K controls the sensitivity to edges and is usually chosen experimentally or as a function of the noise in the image.

Comparison of "Stopping" Functions

To implement an alternate stopping function, the following code should be implemented in the python program:

```
def f(lam,b):
        func = 1/(1 + ((lam/b)**2))
        return func
```

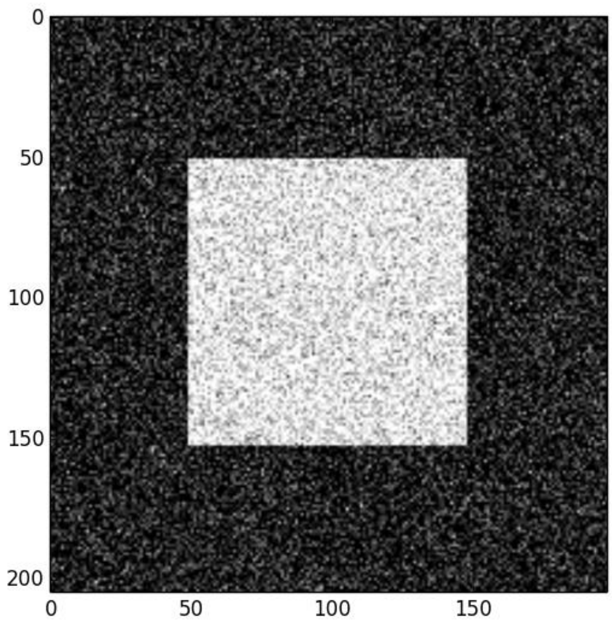
This code is commented into our program.

Procedure

This section contains the input and output from our python code. By viewing these images it is easy to see Perona-Malik anisotropic filtering in action. The images were taken from the sample images provided in the Sulis folder.

Image 1: noisy_rect.png

Before



After

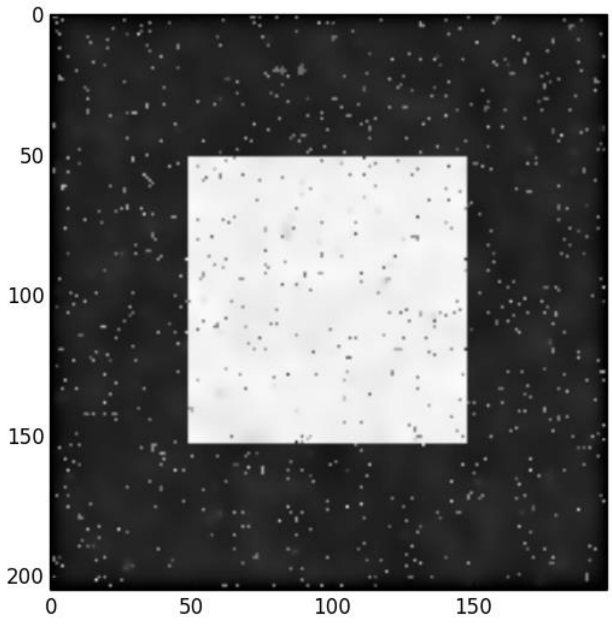
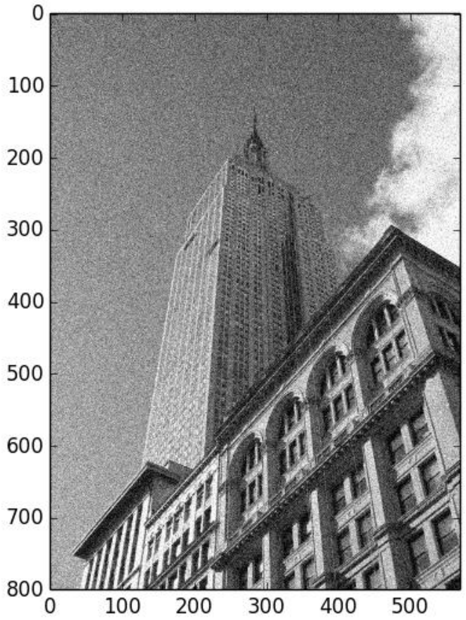
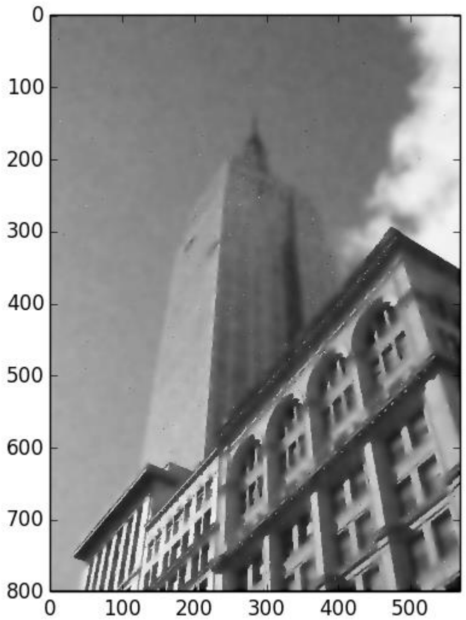


Image 2: noisy_empire.png

Before



After



Conclusions

It is obvious that the above images were successful in efficiently implementing Perona-Malik anisotropic diffusion. The above images clearly a show a significant noise reduction difference between the before and after images. Noise reduction was successfully achieved but it also important to note the edges and lines maintained their sharpness.