MATHEMATICAL APPROACHES TO DIGITAL COLOR IMAGE DENOISING

A Thesis
Presented to
The Academic Faculty

by

Hao Deng

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Mathematics

Georgia Institute of Technology
December 2009

MATHEMATICAL APPROACHES TO DIGITAL COLOR IMAGE DENOISING

Approved by:

Professor Haomin Zhou,
Committee Chair
School of Mathematics
Georgia Institute of Technology

Professor Haomin Zhou, Advisor
School of Mathematics
Georgia Institute of Technology

Professor Yang Wang
School of Mathematics
Michigan State University

Professor Haomin Zhou
School of Mathematics
Georgia Institute of Technology

Professor Ronghua Pan
School of Mathematics
Georgia Institute of Technology

Professor Luca Dieci
School of Mathematics
Georgia Institute of Technology

Professor Sung Ha Kang
School of Mathematics
Georgia Institute of Technology

Date Approved: August 2009

To my family,

and my friends

ACKNOWLEDGEMENTS

I first want to thank my two advisors, Dr. Yang Wang and Dr. Haomin Zhou. Not only did they teach me mathematics, engineering and inspire my research, they also helped me navigate both life and study issues from the very first day I arrived in Georgia Tech. I also feel grateful for all the patience that they has shown with every aspect of my work.

Secondly, let me thank Dr. Ronghua Pan, who introduced me to come to Georgia Tech and helped me a lot in my study and life.

Next, I will thank Dr. Yingjie Liu, Dr. Wilfrid Gangbo, Dr. Prasad Tetali, among others, for teaching me mathematics and for their encouraging discussions. I also want to extend my sincere gratitude to Dr. Luca Dieci, who extended every effort to help me to make everything smooth for me to get work done in these 5 years in Georgia Tech. I particularly want to thank Ms. Cathy Jacobson, who not only gave me the best English training to improve my speaking English and teaching skills, but also helped me to get used to the new environment in United States when I arrived at Atlanta in 2004. My special thanks goes to Ms. Rena Brakebill and Ms. Klara Grodzinsky, for their gracious support of my teaching, and to Ms. Sharon McDowell, Ms. Annette Rohrs, Ms. Genola Turner and IT group for their everyday support. Finally, I express my sincere thanks to all my thesis defense committee members for their careful reading my work.

There are many of my fellow graduate students whose friendship helped me so much over the 5 years. I would like to thank Hua Xu and his wife Jing Xu, who began to help me from the first day, first moment that I arrived at Atlanta. I will thank Luan Lin, for our over one year research discussions in the small but interesting

study group with Yang and Haomin. I have enjoyed numerous happiness with all my friends, notably Wen Jiang, Yongfeng Li, Jian Chen, Zixia Song, Kun Zhao, Nan Lu, Ramazan Tinaztepe, Lingyan Ruan, Jinyu Li, Cuizhen Shen, Liangda Huang, Yan Shu, Jie Ma, Jinyong Ma, Ying Wang and Weizhe Zhang.

Finally and most importantly, I owe so much thanks to my parent who have been supporting me all the time for any choices I made. All their support is the ultimate foundation of anything that I have accomplished.

TABLE OF CONTENTS

LIST OF FIGURES

SUMMARY

As digital photography rapidly replacing the traditional film photography as the photography of choice for all but a few devoted professionals, post image processing of natural color photos such as denoising becomes increasingly an integral part of digital photography.

Many mathematical models have been designed to remove noise from images. Most of them focus on grey value images with additive artificial noise. Only very few specifically target natural color photos taken by a digital camera with real noise. Noise in natural color photos have special characteristics that are substantially different from those that have been added artificially.

In this thesis previous denoising models are reviewed. We analyze the strengths and weakness of existing denoising models by showing where they perform well and where they don't. We put special focus on two models: The steering kernel regression model and the non-local model. For Kernel Regression model, an adaptive bilateral filter is introduced as complementary to enhance it. Also a non-local bilateral filter is proposed as an application of the idea of non-local means filter.

Then the idea of cross-channel denoising is proposed in this thesis. It is effective in denoising monochromatic images by understanding the characteristics of digital noise in natural color images. A non-traditional color space is also introduced specifically for this purpose. The cross-channel paradigm can be applied to most of the exisiting models to greatly improve their performance for denoising natural color images.

CHAPTER I

INTRODUCTION

1.1 Digital Images and Noise

Image processors could be categorized into different levels by the human vision standard. Lower-level ones are to clean and enhance observations, interpolate missing image data, or identify regions occupied by objects without telling what they are. Higher-level processors are to recognize object features and identify the associated hidden real-world contexts, such as face reconization for video surveillance and terrain reading for automatic piloting.

In this sense, the human vision system is a highly advanced and complex image processing senor. It automatically tells what people really want and discards the useless details. But for digital cameras, denoising becomes a hard task. No matter how good cameras are, an image improvement is desirable to extend their range of action.

There are a number of sources of image noise contamination.

Heat generated by cameras or external sources might free electrons from the image sensor itself, thus contaminating the *true* photoelectrons. These *thermal electrons* give rise to a form of noise called thermal noise or dark current.

Another type of noise is more akin to the *grain* obtained by using a high ISO setting (or high ISO film in a film camera). When we use a higher ISO, we are amplifying the signal we receive from the light photons. Unfortunately, as we amplify the signal, we also amplify the background electrical noise that is present in any electrical system.

In low light, there is not enough light for a proper exposure and the longer we

Figure 1: A color image taken by digital camera with real noise

Figure 2: A color image taken by digital camera with real noise(left); 100% crop of the hat(right).

Figure 3: clean image (left); image with additive gaussian noise, $\sigma = 50$(right).

Figure 4: clean image (left); image with additive salt and pepper noise (right).

allow the image sensor to collect the weak signal, the more background electrical noise it also collects. In this case the background electrical noise may be higher than the signal.

Practically, these noise roughly has a Gaussian distribution. This is the so-called amplifier noise, or Gaussian noise. Amplifier noise is a major part of the *read noise* of an image sensor, that is, of the constant noise level in dark areas of the image [37].

Another primary noise is Salt and Pepper noise (Figure 4). An image containing salt-and-pepper noise will have dark pixels in bright regions and bright pixels in dark regions. This type of noise can be caused by dead pixels, analog-to-digital converter errors, bit errors in transmission, etc [38, 39].

1.2 Signal and Noise Ratios

A digital image is generally expressed as a matrix of grey level (1D) or color values (n-D). In a movie, this matrix becomes 3D since the third one is corresponding to time. We use the pair $(i, u(i))$, the position and the value at this positioin, to express a digital image. For a grey value image, $u(i)$ is a scalar; and for a color image, $u(i)$ is a 3D or 4D vector.

Mathematically, one can write the observed image captured by devices as:

$$v(i) = u(i) + n(i), \tag{1.2.1}$$

where $v(i)$ is the observed value, $u(i)$ is the true value, which needs to be recovered from $v(i)$. $n(i)$ is the noise perturbation.

For a grey value image, the range of the pixel value is $(0, 255)$, where 0 represents black and 255 represents white. To measure the amount of noise of an image, one may use the signal noise ratio (SNR)

$$SNR = \frac{\sigma(u)}{\sigma(n)}, \tag{1.2.2}$$

where $\sigma(u)$ denotes the empirical standard deviation of $u(i)$,

$$\sigma(u) = \sqrt{\frac{\sum_i (u(i) - \bar{u})^2}{|I|}},$$

$$\sigma(n) = \sqrt{\frac{\sum_i (u(i) - v(i))^2}{|I|}},$$

where $\bar{u} = \frac{1}{|I|} \sum_i u(i)$ is the average of grey level values, computed from a clean image.

Because many signals have a very wide dynamic range, SNRs are usually expressed in terms of the logarithmic decibel scale. In decibels, the SNR is, by definition, 10 times the logarithm of the power ratio:

$$SNR(dB) = 10 \log_{10} \left(\frac{\sum_i (u(i) - \bar{u})^2}{\sum_i (u(i) - v(i))^2} \right). \tag{1.2.3}$$

Figure 5: clean image (left); image with Gaussian noise, with standard deviation $\sigma = 25$, SNR=5.64dB(right).

Although SNR is widely used in digital image processing, it can only be taken as one of the criteria to determine the quality of an image, otherwise, it might be misleading. See Figure 6.

1.3 Mathematical Model of Noise Removal

A denoising method can be defined as D_h working on an image u:

$$u = D_h u + n(D_h, u), \tag{1.3.4}$$

where h is the filtering parameter, $D_h u$ is the denoised image, and $n(D_h, u)$ is the noise guessed by the method.

Nowadays, it is not enough just to smooth u and get the denoised image. The more recent methods are actually not contented with a smoothing, but try to recover lost information in $n(D_h, u)$ as needed [8, 11], i.e. in a image captured by digital SLR cameras, we often need to keep the sharpness and the detailed information while the noise is being blurred.

In [16], the image method noise is defined as below:

Definition 1.4.1 (Method noise) Let u be an image and D_h a denoising operator depending on h. Then we define the method noise of u as the image difference

$$n(D_h, u) = u - D_h(u).$$

So in this thesis, three criteria will be taken into account in the comparision of denoising methods:

- a display of typical artifacts in denoised images.
- a formal computation of the method noise on smooth images, evaluation how small it is in accordance with image local smoothness.
- a classical comparison method based on noise simulation : it consists of taking a good quality image, add gaussian white noise to it with known σ and then compute the best image recovered from the noisy one by each method. A table of L^2 distances from the restored to the original can be established. The L^2 distance does not provide a good quality assessment. However, it reflects well the relative performances of algorithms.

We have to make the comment that, the method noise introduced in [16] is actually the residue between the original image and the reconstructed image. This mathematically computed *error* is used in [16] as an criteria to determine how good the denoising filter is. Ideally, this method noise should be as small as possible and as similar to a white noise as possible. Just like SNR, practically it should not be taken as the only valuable criteria to determine the quality of a filter, and can be misleading. See Figure 7.

1.4 Previous Methods

We had to make a selection of the denoising methods we wished to compare. Here a difficulty arises, as most original methods have caused an abundant literature proposing many improvements. We shall analyze :

1. Local filtering methods, including
    - Gaussian smoothing model (Gabor [7]), where the smoothness of u is measured by the Dirichlet integral $\int |Du|^2$.

(a) A clean chicken image

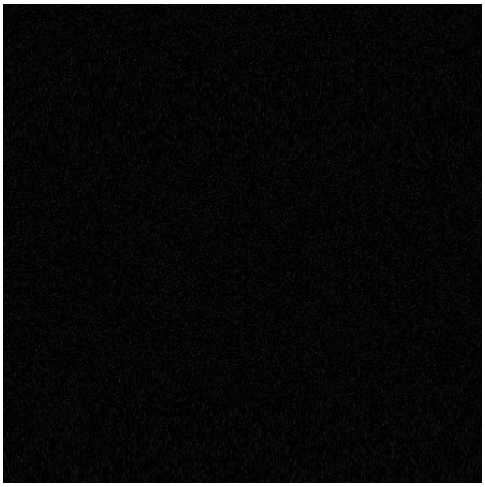
(b) Same chicken image with higher contrast, performed by *curve adjustment* in photoshop, with SNR=7.6476

(c) Additive gaussian noisy image, with SNR=26.0732

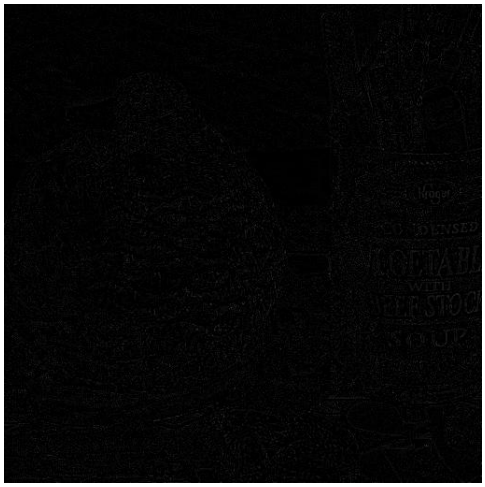
(d) Real noisy image, captured by setting a high ISO, with SNR=28.9343

Figure 6: (a) is set as the original clean image. (b) is also visually noiseless and have same geometric structure as (a), but with a low SNR. (c) and (d) are both noisy, but with much higher SNRs.

(a) Residue of (b) in previous figure



(b) Residue of (c) in previous figure



(c) Residue of (d) in previous figure

Figure 7: The first residue is obviously *bigger* than the other two.

- Yaroslavsky ([32, 31]) neighborhood filters and an elegant variant, the Susan filter (Smith and Brady) [15]
- the Bilateral filter [25].

2. PDE based methods, including
   - anisotropic filtering model (Perona-Malik [12], Alvarez et al. [1])
   - Rudin-Osher-Fatemi [6] total variation model
3. Frequency domain filters, including
   - local adaptive filters in transform domain
   - hard and soft thresholding
   - Zhou-Wang wavelet total variation [40]
4. Steering kernel regression method [33]
5. Non-local means (NL-means) algorithm [16]
6. cross-channel paradigm for color images

The last algorithm may deserve a lot more attention here because it is based on decomposition of the color space. Instead of filtering the channels independently, we will consider the correlation between channels to assist denoising. This paradigm can work as an extension for all previous schemes.

1.5 Plan for This Thesis

In this thesis, Chapter 2-6 reviewed previous methods, including local/nonlocal filters, frequency domain filters, PDEs methods and steering kernel regression method, etc. For some of them we analyze or recall the asymptotic expansion of the filter at smooth

points of the image and therefore obtain a formal expression of the method noise. We try to point out places where the filter performs well and where it fails. In Chapter 5, an adaptive bilateral filter is introduced as a complementary to enhance the steering kernel regressionn method. The enhancement will be shown when this method is dealing with almost clean images or removing salt and pepper noise. In Chapter 6, a new non-local filter based on the bilateral filter is introduced. In Chapter 7, the cross-channel paradigm is introduced, i.e. information exchange between different-channels is used for color images denoising, in which a new $mYCrCb$ color space decomposition is proposed. Meanwhile, a new mathematical frame is built to apply previous methods via cross-channel paradigm. In the last chapter, we compare all algorithms from several points of view and showed some experiment results.

CHAPTER II

LOCAL FILTERING METHODS

The original (gray value) image u is defined in a bounded domain $\Omega \subset \mathbb{R}^2$, and is denoted by $u(x)$ for $x \in \mathbb{R}^2$.

The noise is a discrete phenomenon on the sampling grid. According to the usual screen and printing visualization practice, we do not interpolate the noise samples n_i as a band limited functions, but rather as a piecewise constant function, constant on each pixel i and equal to n_i.

We write $|x|$ as L^2 norm and $x \cdot y$ as the inner product.

2.1 Gaussian Filter

This is the most commonly used blurring filter. It is actually a convolution of the image by a linear symmetric kernel. The smoothing requirement is usually expressed by the positivity of the kernel. The formula of such kernel is the so-called Gaussian Kernel

$$x \to G_h(x) = \frac{1}{(4\pi h^2)} e^{-\frac{|x|^2}{4h^2}}. \tag{2.1.1}$$

G_h has standard deviation h and the method noise is easily computed:

Theorem 2.1.1. *(Gabor 1960) The image method noise of the convolution with a gaussian kernel G_h is*

$$u - G_h * u = -h^2 \Delta u + o(h^2). \tag{2.1.2}$$

The estimate is valid if h is small enough. On the other hand, the noise reduction properties depend upon the fact that the neighborhood involved in the smoothing is large enough, so that the noise gets reduced by averaging. In the following, if we assume that $h = k\epsilon$, where k is the number of samples of the function u and of the

noise in an interval of length h, ϵ^2 is the size of a local window. k must be much larger than 1.

At a reference pixel $i = 0$, the Gaussian smoothing effect is evaluated as:

$$G_h * n(0) = \sum_{i \in I} \int_{P_i} G_h(x) n(x) dx = \sum_{i \in I} \epsilon^2 G_h(i) n_i, \tag{2.1.3}$$

where $n(x)$ is been interpolated as a piecewise function, the P_i square pixels centered in i have size ϵ^2 and $G_h(i)$ denotes the mean value of the function G_h on the pixel i.

The following theorem is proved in [16],

Theorem 2.1.2. *Let $n(x)$ be a piecewise constant noise, with $n(x) = n_i$ on each square pixel i. Assume that the n_i are i.i.d. with zero mean and variance σ^2. Then the 'noise residue' after a gaussian convolution of n by G_h satisfies*

$$Var(G_h * n(0)) \approx \frac{\epsilon^2 \sigma^2}{8\pi h^2}. \tag{2.1.4}$$

In other terms, the standard deviation of the noise, which can be identified with the noise amplitude, is multiplied by $\frac{\epsilon}{h\sqrt{8\pi}}$.

The first theorem tells us that the method noise of the gaussian denoising method is zero in harmonic parts of the image. A Gaussian convolution is optimal on harmonic functions, and performs instead poorly on singular parts of u, namely edges or texture, where the Laplacian of the image is large [16].

2.2 Neighborhood Filters

In stead of considering a notion of spatial neighborhood or proximity, neighborhood filters take into account grey level values to define neighboring pixels. In this case, the denoised value at pixel i is an (weighted) average of values at pixels which have a grey level value close to $u(i)$. We may define the grey level neighborhood as

$$U(i, h) = \{j \in I | u(i) - h < u(j) < u(i) + h\}. \tag{2.2.5}$$

So this is also a *local* scheme, i.e. *locally* in intensity domain. But it is non-local in spatial domain, since pixels belonging to the whole image are used for the estimation at pixel i. A popular variation of this algorithm is the following weighted average filter:

$$NF_h u(x) = \frac{1}{C(x)} \int_\Omega u(y) e^{-\frac{|u(y)-u(x)|^2}{h^2}} dy, \tag{2.2.6}$$

where $\Omega \subset R^2$ is an open and bounded set, and $C(x) = \int_\Omega u(y) e^{-\frac{|u(y)-u(x)|^2}{h^2}} dy$ is the normalization factor.

In [16], the method noise of this algorithm is also computed.

Lemma 2.2.1. *Suppose u is Lipschitz in Ω and $h > 0$, then $C(x) \geq O(h^2)$.*

Proof. Given $x, y \in \Omega$, by the Mean Value Theorem,

$$|u(x) - u(y)| \leq K|x - y| \tag{2.2.7}$$

for some real constant K. Then,

$$\begin{aligned} C(x) &= \int_\Omega u(y) e^{-\frac{|u(y)-u(x)|^2}{h^2}} dy \\ &\geq \int_{B(x,h)} u(y) e^{-\frac{|u(y)-u(x)|^2}{h^2}} dy \\ &\geq e^{-K^2} O(h^2). \end{aligned} \tag{2.2.8}$$

□

Proposition 2.2.2. *(Method noise estimate). Suppose u is a Lipschitz bounded function on Ω, where Ω is an open and bounded domain of R^2. Then $|u(x) - NF_h u(x)| = O(h\sqrt{-\log h})$, for h small, and, $x \in \Omega$.*

The Yaroslavsky [31, 32] neighborhood filters consider mixed neighborhoods $U(i,h) \cap B_{\rho(i)}$, where $B_\rho(i)$ is a ball of center i and radius ρ. So the method takes an average

of the values of pixels which are both close in grey level and spatial distance. The *Susan filter* [15] proposes the following:

$$YNF_{h,\rho}(x) = \frac{1}{C(x)} \int_{B_\rho(x)} u(y) e^{-\frac{|u(y)-u(x)|^2}{h^2}} dy, \tag{2.2.9}$$

where $C(x)$, as before, is the normalization factor.

In [16], Antoni et al studied the method noise in the 1D and 2D cases for the filter. They showed:

Theorem 2.2.3. *Suppose $u \in C^2((a,b))$, a,$b \in R$. Then, for h,$\rho \in R^+$ and h small enough*

$$u(s) - YNF_{h,\rho}(s) \approx -\frac{h^2}{2} u''(s) f(\frac{h}{\rho}|u'(s)|), \tag{2.2.10}$$

where

$$f(t) = \frac{\frac{1}{3} - \frac{3}{5}t^2}{1 - \frac{1}{3}t^2}.$$

This tells us the method noise of the neighborhood filter is actually a locally weighted Laplacian, with the weighting function f positive or negative. The zeros and the discontinuity points of f represent the singular points where the behavior of the method noise changes between Laplacian and inverse Laplacian. The magnitude of this change is much larger near the discontinuities of f producing a shock or staircase effect.

Theorem 2.2.4. *Suppose $u \in C^2(\Omega)$, $\Omega \subset R^2$. Then, for h,$\rho \in R^+$ and h small enough*

$$u(x) - YNF_{h,\rho}(x) \approx -\frac{h^2}{8} \left(\Delta u f(\frac{h}{\rho}|Du|) + D^2 u(\frac{Du}{|Du|}, \frac{Du}{|Du|}) g(\frac{h}{\rho}|Du|) \right) \tag{2.2.11}$$

where

$$f(t) = \frac{1 - \frac{1}{2}t^2}{1 - \frac{1}{4}t^2}, \qquad g(t) = \frac{-t^2}{1 - \frac{1}{4}t^2}.$$

It was stated in [16] that, this method is very unstable. If t is near zero, it behaves like a Gaussian smoothing. If t is near $\sqrt{2}$, the stuctures with a large value of $D^2u(\frac{Du}{|Du|}, \frac{Du}{|Du|})$ will not be preserved. And, edge points are points where $D^2u(\frac{Du}{|Du|}, \frac{Du}{|Du|})$ is zero; on both sides of the edges it can be instead large and the method actually enhances the edges by making the image flat on both sides. If $t \in (0, \sqrt{2})$ and $|f(t)| \approx |g(t)|$, the filter behaves like an anistropic filter. If t is near 2 where the functions $f(t)$ and $g(t)$ have an asymptotical discontinuity. This instability can deal to unwanted shock effects anf artifacts.

2.3 Bilateral Filters

The baliteral filter was first proposed by C. Tomasi and R. Manduchi [22] in 1998. It applies spatial weighted averaging without smoothing edges. This is achieved by combing two Gaussian filters: one filter works in spatial domain, the other filter works in intensity domain. Therefore, not only the spatial distance but also the intensity distance is important for the determination of weights.

For a given image $u(x)$, at a pixel location x, the output of a bilateral filter can be formulated as:

$$B(x) = \frac{1}{C(x)} \sum_{y \in N(x)} e^{-\frac{\|y-x\|^2}{2\sigma_d^2}} e^{-\frac{|u(y)-u(x)|^2}{2\sigma_r^2}} u(y) \tag{2.3.12}$$

where σ_d and σ_r are parameters controlling the fall-off of weights in spatial (distance) and intensity (radiometric) domains, $N(x)$ is a spatial neighborhood of pixel $u(x)$, and

$$C(x) = \sum_{y \in N(x)} e^{-\frac{\|y-x\|^2}{2\sigma_d^2}} e^{-\frac{|u(y)-u(x)|^2}{2\sigma_r^2}}$$

is the normalization factor.

Let's study method noise of the bilateral filter:

Proposition 2.3.1. *Suppose u is a Lipschitz bounded function on Ω, where Ω is an*

open and bounded domain of R^2. *then* $|u(x) - B_{\sigma_r,\sigma_d}u(x)| = O(\sigma_r\sqrt{-\log\sigma_d} + \sigma_d^3\Delta u)$, *for* σ_r, σ_d *small,* $0 < \sigma_r < 1$, $0 < \sigma_d < 1$, $x \in \Omega$.

Proof. Let x be a point of Ω and for a given B and σ_r, σ_d $B, \sigma_r, \sigma_d \in R$, consider the set $D_{\sigma_r} = \{y \in \Omega || u(y) - u(x)| \leq B\sigma_r\}$. Then

$$|u(x) - B_{\sigma_r,\sigma_d}u(x)| \leq \frac{1}{C}\int_{D_{\sigma_r}} W_1W_2|u(y) - u(x)|dy + \frac{1}{C}\int_{D^c_{\sigma_r}} W_1W_2|u(y) - u(x)|dy, \tag{2.3.13}$$

where $W_1 = e^{-\frac{|u(y)-u(x)|^2}{\sigma_r^2}}$, $W_2 = e^{-\frac{\|y-x\|^2}{\sigma_d^2}}$.

Considering that $|u(y) - u(x)| \leq B\sigma_r$ for $y \in D_{\sigma_r}$ and $\int_{D_{\sigma_r}} W_1W_2dy \leq C(x)$ one sees that the first term is bounded by $B\sigma_r$. And consider that $W_1 \leq e^{-B^2}$ for $y \notin D_{\sigma_r}$. And $\int_{D^c_{\sigma_r}} W_2|u(y) - u(x)|dy$ can be estimated by the method noise of Gaussian filter:

$$\int_{D^c_{\sigma_r}} W_2|u(y) - u(x)|dy \leq -\sigma_d^2\Delta u + o(\sigma_d^2). \tag{2.3.14}$$

So the second term has an order $O(e^{-B^2}\sigma_d^2\Delta u)$. Finally, choosing B such that $B^2 = -\log\sigma_d$ yields

$$\begin{aligned} |u(x) - B_{\sigma_r,\sigma_d}u(x)| &\leq B\sigma_r + O(e^{-B^2}\sigma_d^2\Delta u) \\ &= O(\sigma_r\sqrt{-\log\sigma_d}) + O(\sigma_d^3\Delta u). \end{aligned} \tag{2.3.15}$$

□

Although the bilateral filter was first proposed as an intuitive tool, recent papers have pointed out the connections with some well established techniques. [25] shows that the bilateral filter is identical to the first iteration of the Jacobi algorithm (diagonal normalized steepest descent) with a specific cost function. [28] and [29] relate the bilateral filter with the anisotropic diffusion.

The bilateral filter is widely used in applications. But there is not much theoretical basis on selecting the window size of the neighborhood. This parameter is typically selected by trial and experiments. The apparent fact is, if the window size is too small, the denoising effect is limited, since not much related pixels are used. But when we increase the window size, i.e. a global window size is selected, more and more unrelated pixels will be included to affect the denoising quality, and practically, a large window size will make the algorithm inefficient [20].

CHAPTER III

PDES MODEL METHODS

3.1 Anisotropic Diffusion Equation

Diffusion algorithms remove noise from an image without removing significant edges by modifying the image via a partial differential equation (PDE). This is done by a diffusion process in which the image is iteratively filtered that is locally adapted to the underlying image signal.

Let $I(x,t)$ be the image in the diffusion process at time t, $I(x,0) : R^2 \rightarrow R^+$ is the original image in the continuous domain, $x \in R^2$ is the position of a pixel in a 2D image. In the past, the isotropic diffusion equation (the heat equation)

$$\frac{\partial I(x,t)}{\partial t} = div(\nabla I), \quad I(x,0) = u \tag{3.1.1}$$

has been used to denoise images. It has been proved that modifying the image according to this isotropic diffusion equation is equivalent to filtering the image with a Gaussian filter (Witkin [21]).

Perona and Malik [12] replaced the classical isotropic diffusion equation with the following anisotropic diffusion in 1990 for image denoising:

$$\frac{\partial I(x,t)}{\partial t} = div[g(\|\nabla I\|)\nabla I)], \quad I(x,0) = u \tag{3.1.2}$$

where $\|\nabla I\|$ is the gradient magnitude, and $g(\|\nabla I\|)$ is an *edge stopping* function. This function is chosen to satisfy $g(z) \rightarrow 0$ when $z \rightarrow \infty$ so that the diffusion is stopped without crossing edges. The idea is that the smoothing process obtained by the equation is '*conditional*: if ∇I is large, then the diffusion will be low and therefore the exact localization of the edges will be kept; if ∇I is small, then the diffusion will tend to smooth still more around x.

In applications, Perona and Malik [12] proposed two functions for the diffusion coefficient:

$$g(\|\nabla I\|) = e^{-(\|\nabla I\|/K)^2},$$

and

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

where in both models, the constant K controls the sensitivity to edges and is usually chosen experimentally or as a function of the noise in the image. The authors pointed out that the difference between these two functions is: the first privileges high-contrast edges over low-contrast edges, the second privileges wide regions over smaller ones.

3.2 Total Variation

The Total Variation minimazation was introduced by Rudin, Osher and Fatemi [13, 14]. Let $v(x) = u(x) + n(x)$, where u denotes the uncontaminated underlying image and n denote the noise. To reconstruct u one considers the problem of minimizing

$$E(u) = \lambda\|u - v\|^2_{L^2(\Omega)} + R(u), \tag{3.2.3}$$

where $\lambda > 0$, Ω is the domain on which v is defined, and the $R(u)$ is a regularization functional. Earlier efforts focused on least square based functionals $R(u)$'s such as $\|\Delta u\|^2_{L^2(\Omega)}$, $\|\nabla u\|^2_{L^2(\Omega)}$ and others. While noise can be effectively removed, these regularization functionals penalize discontinuity, resulting in soft and smooth reconstructed images, with subtle details lost.

A better choice for the functional space modelling these properties is $BV(\Omega)$, the space of integrable functions with finite total variation $TV_\Omega(u) = \int |\nabla u|$. Given a noisy image $v(x)$, the above mentioned authors proposed to recover the original image $u(x)$ as the solution of the constrained minimization problem

$$\text{argmin}_u TV_\Omega(u)$$

subject to the noise constraints

$$\int_\Omega (u(x) - v(x))dx = 0 \quad \text{and} \quad \int_\Omega |u(x) - v(x)|^2 dx = \sigma^2.$$

The solution u must be as regular as possible in the sense of the total variation, while the difference $u - v$ is treated as an error, with a prescribed energy. The constraints prescribe the right mean and variance to $u - v$, but do not ensure that it be similar to a noise ([10]). So the preceding problem becomes

$$\arg\min_u (TV_\Omega(u) + \lambda \int_\Omega |v(x) - u(x)|^2 dx) \tag{3.2.4}$$

for a given Lagrange multiplier λ. The above function is strictly convex and lower semicontinuous with respect to the weak-star topology of BV. Therefore the minimum exists, is unique and computable [2]. The parameter λ controls the trade-off between the regularity and fidelity terms. As λ gets smaller the weight of the regularity term increases. Therefore, λ is related to the degree of filtering of the solution of the minimization problem.

Intensive studies have shown that the total variation better preserves edges in u, thus it allows for sharper reconstructions, e.g. [17, 2, 18, 19]. Among all the PDE based techniques, the TV minimization scheme is a candidate that offers the best combination of noise removal and feature preservation.

To solve the minimizers for the preceding TV minimization problem, it is similar to the anistropic diffusion scheme. For the TV minimization, it is easy to show that the minimizer u satisfies the following PDE:

$$\nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) - \lambda(u - v) = 0. \tag{3.2.5}$$

And in practice, one introduces the time variable t and solve for $u(x, t)$ by time-marching the equation

$$u_t = \nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) - \lambda(u - v), \quad u(x, 0) = v(x). \tag{3.2.6}$$

In this case, straight edges are maintained because of their small curvature. In fact significant edges are sharpened [40].

There are two approaches to be used for achieve the best combination of noise removal and feature preservation in this scheme. One is to tune the parameter λ. Obviously if λ is chosen too large, it may not be able to remove enough noise. However, if λ is chosen too small, details and texture will be over smoothed as, and it ends up with a Cartoon-like piecewise constant image.

3.3 Interated Total Variation

With a further study of the removed noise $u(x) - v(x)$ in the original TV model, Burger ([11]) proposed the following iterated TV model.

- Solve
$$u_1 = \arg\min_{u \in BV} \{ \int |\nabla u(x)| dx + \lambda \int (v(x) - u(x))^2 dx \}.$$
- Perform a correction step
$$u_2 = \text{argmin}_{\text{u}\in\text{BV}} \{ \int |\nabla u(x)| dx + \lambda \int (u_1(x) - u(x))^2 dx \},$$
where u_1 becomes the new observed image.
- Iteration: compute u_{k+1} as a minimizer of the modified total variation minimization,
$$u_{k+1} = \text{argmin}_{\text{u}\in\text{BV}} \{ \int |\nabla u(x)| dx + \lambda \int (u_k(x) - u(x))^2 dx \},$$

It was proved in [11] that:

- u_k converges monotonically in L^2 to v, the noisy image, as $k \to \infty$.
- u_k approaches the noisy free image monotonically in the Bregman distance associated with the BV seminorm, at least until $\|u_k - u\| \leq \sigma^2$, where u is the original image and σ is the standard deviation of the added noise.

These two results indicate how to stop the sequence and choose u_k. It is enough to proceed iteratively until the result gets noisier or the distance $\|u_k - u\|^2$ gets smaller than σ^2. The new solution has more details preserved. One can attempt an explanation of this improvement by computing the method noise. The above iterated denoising strategy being quite general, one can make the computations for a linear denoising operator T as well. In that case, the method noise after one iteration is

$$u - T(u + n_1) = n_1 - (T(u + n_1) - T(u)) = n_1 - T(n_1).$$

In the linear case, this amounts to say that the first estimated noise n_1 is filtered again and its smooth components added back to the original.

CHAPTER IV

FREQUENCY DOMAIN FILTERS

Let u be an image defined on R^n. u is supposed to be modified by a white noise n, where n is a random process and $n(i)$ are i.i.d, with zero mean and constant variance σ^2. So the observed image v can be written as:

$$v(i) = u(i) + n(i).$$

Let $B = \{g_\alpha\}_{\alpha \in A}$ be an orthogonal basis of R^n. Define

$$v_B(\alpha) = \langle v, g_\alpha \rangle, \quad u_B(\alpha) = \langle u, g_\alpha \rangle, \quad n_B(\alpha) = \langle n, g_\alpha \rangle$$

the scalar products. Then

$$v_B(\alpha) = u_B(\alpha) + n_B(\alpha)$$

is the transformed noise process.

Note that noise coefficients $n_B(\alpha)$ remain uncorrelated and zero mean, but the variances are multiplied by $\|g_\alpha\|^2$,

$$\begin{aligned} E[n_B(\alpha) n_B(\beta)] &= \sum_{i,j \in I} g_\alpha(i) g_\beta(j) E[n(i)n(j)] \\ &= \langle g_\alpha, g_\beta \rangle \sigma^2 \\ &= \sigma^2 \|g_\alpha\|^2 \delta[\alpha - \beta]. \end{aligned} \tag{4.0.1}$$

Frequency domain filters are applied independently to every transform coefficient $v_B(\alpha)$. When we reconstruct the image, we just perform the inverse transform of the new coefficients. Noisy coefficients $v_B(\alpha)$ are modified to $a(\alpha)v_B(\alpha)$, where $a(\alpha)$ are often restricted to be 0 or 1. This is a nonlinear algorithm since $a(\alpha)$ depends on the

value $v_B(\alpha)$. The inverse transform yields the estimate

$$\hat{U} = DV = \sum_{\alpha \in A} a(\alpha) v_B(\alpha) g_\alpha. \tag{4.0.2}$$

The classical frequency domain filter, the *Fourier Wiener Filter* used the Fourier basis for B. By the use of the Fourier basis, global image characteristics may prevail over local ones and create spurious periodic patterns. To avoid this effect, the basis must take into account more local features, as the wavelet and local DCT transforms do. The problem of finding an ideal basis for each given application is still an important problem in image processing.

4.1 Wavelet Thresholding

The *wavelet thresholding methods* were introduced by D. Donoho [3]. Let $B = \{g_\alpha\}_{\alpha \in A}$ be an orthonormal basis of wavelet [9]. A certain threshold μ is chosen to determine $a(\alpha)$, i.e.

$$a(\alpha) = \begin{cases} 1 & |v_B(\alpha)| > \mu \\ 0 & |v_B(\alpha)| \leq \mu \end{cases} \tag{4.1.3}$$

This is the so-called *hard thresholding*, which cancels coefficients smaller than the threshold. This procedure is based on the idea that the image is represented with large wavelet coefficients, which are kept, whereas the noise is distributed usually as small coefficients, which are cancelled. The performance of the method depends on the capacity of approximating u by a small set of large coefficients. We may denote this operator by $HWT_\mu(v)$.

But this algorithm will create some small oscillations, i.e. Gibbs-like phenomenon, near the edges, due to the simple cancellation of the wavelet coefficients lower than the threshold. D. Donoho [4] showed that these effects can be partially reduced if we use a *soft thresholding*,

$$a(\alpha) = \begin{cases} \frac{v_B(\alpha) - sgn(v_B(\alpha))\mu}{v_B(\alpha)} & |v_B(\alpha)| \geq \mu \\ 0 & |v_B(\alpha)| < \mu \end{cases} \tag{4.1.4}$$

which will be denoted by $SWT_\mu(v)$. The continuity of the soft thresholding operator preserves the structure of the wavelet coefficients and reduces the oscillation near edges.

4.2 Wavelet Total Variation

In 2006, Wang and Zhou proposed this algorithm for medical images baseon on a combination of the total variation minimization scheme and the wavelet scheme.

Let $z(x) = u_0(x) + n(x)$ be the observed image, where $n(x)$ is the noise, and $u_0(x)$ is to be recovered. All these three functions are in some functional space F, such as $L^2(\Omega)$ for some domain $\Omega \in R^2$. Let $\{\phi_j : j \in I\}$ be an orthonormal basis [41, 42] for F, if F is a Hilbert space. Let

$$z(x) = \sum_{j \in I} \alpha_j \phi_j(x),$$

and denote

$$u(x, \beta) = \sum_{j \in I} \beta_j \phi_j(x),$$

where $\beta = (\beta_j)$. Define the Total Variation function by

$$F(u) = \int_{R^2} |\nabla_x u(x, \beta)| dx + \frac{1}{2} \sum_{j \in I} \lambda_j (\beta_j - \alpha_j)^2, \tag{4.2.5}$$

where $\lambda_j > 0$. Then the goal of denoising is to minimize $F(u)$ and find the minimizer $u^* = u(x, \beta)$ such that $F(u^*) = \min_\beta F(u)$.

For $u = u(x, \beta)$, where $\beta = (\beta_j)$,

$$\begin{aligned} \frac{\partial F(u)}{\partial \beta_j} &= \int_{R^2} \frac{\nabla_x(u)}{|\nabla_x u|} \cdot \nabla_x \phi_j dx + \lambda(\beta_j - \alpha_j) \\ &= -\int_{R^2} \nabla_x \cdot \left[\frac{\nabla_x(u)}{|\nabla_x u|} \right] \phi_j dx + \lambda(\beta_j - \alpha_j). \end{aligned} \tag{4.2.6}$$

Then the Euler-Lagrange equation for the model is

$$-\int_{R^2} \nabla_x \cdot \left[\frac{\nabla_x(u)}{|\nabla_x u|} \right] \phi_j dx + \lambda(\beta_j - \alpha_j) = 0. \tag{4.2.7}$$

And practically, an artificial time parameter t and time-march image using gradient flow is introduced,

$$\frac{\partial \beta_j}{\partial t} = \int_{R^2} \nabla_x \cdot \left[\frac{\nabla_x(u)}{|\nabla_x u|} \right] \phi_j dx - \lambda(\beta_j - \alpha_j), \quad \beta_j(0) = \alpha_j. \tag{4.2.8}$$

Comparing to the wavelet hard/soft thresholding scheme, this algorithm will eliminate the Gibbs oscillation. One of the biggest issues in the traditional TV model is over-smoothing. For the wavelet TV model Wang and Zhou introduced an effective automatic stopping criterion. Comparing to the traditional TV models, by setting a auto-stopping time criterion based on certain statistical property of wavelet coefficients, the wavelet TV model will prevent the image from being over-smoothed.

CHAPTER V

STEERING KERNEL REGRESSION

In 2006, Hiroyuki Takeda, Sina Farsiu and Peyman Milanfar [33] introduced this denoising method based on the field of nonparametric statistics. Basically, a specific steering matrix is chosen to determine a footprint of *homogeneous* region, and in this region, a weighted kernel regression problem with that steering matrix is solved to smooth a homogeneous region of the image without crossing edges of the footprint.

5.1 Classical Steering Kernel Regression

Let

$$y_i = z(x_i) + \epsilon_i, \quad i = 1, \cdots, P$$

where $y_i = y(x_i)$ is the observed data, ϵ_i is the noise and we want to recover $z(x_i)$ from y_i.

Since x_i is a 2×1 vector, we may find the local expansion of the regression function of $z(x_i)$

$$z(x_i) = \beta_0 + \beta_1^T(x_i - x) + \beta_2^T \text{vech}\{(x_i - x)(x_i - x)^T\} + \cdots \tag{5.1.1}$$

where vech$(\cdot)$ is defined as the half-vectorization operator of the "lower-triangular" portion of a symmetric matrix, e.g.

$$\text{vech}\begin{pmatrix} a & b \\ b & d \end{pmatrix} = [a \;\; b \;\; d]^T$$

and

$$\text{vech}\begin{pmatrix} a & b & c \\ b & e & f \\ c & f & i \end{pmatrix} = [a \;\; b \;\; c \;\; e \;\; f \;\; i]^T$$

We know that $\beta_0 = z(x)$ is the pixel value of interest and the vectors β_1 and β_2 are

$$\beta_1 = \nabla z(x) = \left[\frac{\partial z(x)}{\partial x_1}, \frac{\partial z(x)}{\partial x_2}\right],$$
$$\beta_2 = \frac{1}{2}\left[\frac{\partial^2 z(x)}{\partial x_1^2}, 2\frac{\partial^2 z(x)}{\partial x_1 \partial x_2}, \frac{\partial^2 z(x)}{\partial x_2^2}\right].$$

The β_ns can be computed from the following optimazation problem:

$$\min_{\{\beta_n\}} \sum_{i=1}^{P} \left[y_i - \beta_0 - \beta_1^T(x_i - x) - \cdots\right]^2 K_H(x_i - x) \tag{5.1.2}$$

with

$$K_H(t) = \frac{1}{det(H)} K(H^{-1}t),$$

where K is the 2-D realization of the kernel function, and H is a 2×2 smoothing matrix.

In [33], the authors gave the solution of this weighted least-squares optimization problem:

$$\begin{aligned}\hat{b} &= \arg\min_b \|y - Xb\|_W^2 \\ &= \arg\min_b (y - Xb)^T W (y - Xb),\end{aligned} \tag{5.1.3}$$

where

$$y = [y_1, y_2, \cdots, y_P]^T, \quad b = [\beta_0, \beta_1^T, \cdots, \beta_N^T]^T, \tag{5.1.4}$$

$$W = \text{diag}[K_H(x_1 - x), K_H(x_2 - x), \cdots, K_H(x_P - x)], \tag{5.1.5}$$

$$X = \begin{bmatrix} 1 & (x_1 - x)^T & \text{vech}^T\{(x_1 - x)(x_1 - x)^T\} & \cdots \\ 1 & (x_2 - x)^T & \text{vech}^T\{(x_2 - x)(x_2 - x)^T\} & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ 1 & (x_P - x)^T & \text{vech}^T\{(x_P - x)(x_P - x)^T\} & \cdots \end{bmatrix}. \tag{5.1.6}$$

Since the only necessary computations are limited to the ones that estimate β_0, the least-squares estimation is simplified to

$$\hat{\beta_0} = e_1^T (X^T W X)^{-1} X^T W y \tag{5.1.7}$$

where $e_1 = (1, 0, \cdots, 0)^T$ is a column vector in $\mathbb{R}^P$.

The shape of the regression kernel as defined above, and consequently, the performance of the estimator depend on the choice of the smoothing matrix H [26]. In [33], a data-dependent *steering* kernel were choosed by

$$K = K_{H_i^{steer}}(x_i - x), \tag{5.1.8}$$

where

$$H_i^{steer} = h\mu_i C_i^{-\frac{1}{2}},$$

and μ_i's are scalars that captures the local density of data samples (nominally set to $\mu_i = 1$), h is the global smoothing parameter, C_i's are (symmetric) covariance matrices based on differences in the local gray-values. A good choice for C_i's will effectively spread the kernel function along the local edges.

Since the local edge structure is related to the gradient covariance(or equivalently the locally dominant orientation), a naive estimate of this covariance matrix would be:

$$C_i \approx \begin{bmatrix} \sum_{x_j \in w_i} z_{x_1}(x_j) z_{x_1}(x_j) & \sum_{x_j \in w_i} z_{x_1}(x_j) z_{x_2}(x_j) \\ \sum_{x_j \in w_i} z_{x_1}(x_j) z_{x_2}(x_j) & \sum_{x_j \in w_i} z_{x_2}(x_j) z_{x_2}(x_j) \end{bmatrix} \tag{5.1.9}$$

where $z_{x_1}(\cdot)$ and $z_{x_2}(\cdot)$ are the first derivatives along x_1 and x_2 directions and w_i is the local analysis window around the position of interest. The domaint local orientation of the gradients is then related to the eigenvectors of this estimated matrix.

This approach is simple and has nice tolerance to noise, but practically, the computation of steering matrices in this way may be rank deficient or unstable. Therefore, care must be taken not to take the inverse of the estimate directly in this case. [27] proposed an effective multiscale technique for estimating local orientations, which fits the requirements of this problem nicely.

Using the eigenvalues decomposition, C_i is decomposed into:

$$C_i = \gamma_i U_{\theta_i} \Lambda_i U_{\theta_i}^T, \tag{5.1.10}$$

where

$$U_{\theta_i} = \begin{bmatrix} \cos\theta_i & \sin\theta_i \\ -\sin\theta_i & \cos\theta_i \end{bmatrix}, \quad \Lambda_i = \begin{bmatrix} \sigma_i & 0 \\ 0 & \sigma_i^{-1} \end{bmatrix}$$

Now U_{θ_i} is the rotation matrix, Λ_i is the elongation matrix and γ_i is the scaling parameter. Following the work in [27], the dominant orientation of the local gradient field is the singular vector corresponding to the smallest (nonzero) singular value of the local gradient matrix arranged:

$$G_i = \begin{bmatrix} \vdots & \vdots \\ z_{x_1}(x_j) & z_{x_2}(x_j) \\ \vdots & \vdots \end{bmatrix} = U_i S_i V_i^T, \quad x_j \in w_i \tag{5.1.11}$$

where $U_i S_i V_i^T$ is the truncated singluar value decomposition of G_i, and S_i is a diagnal 2×2 matrix representing the energy in the domainant directions, with diagnals s_1 and s_2. The second column of the 2×2 orthogonal matrix V_i, $v_2 = [\nu_1, \nu_2]^T$, defines the dominant orientation angle θ_i

$$\theta_i = \arctan\left(\frac{\nu_1}{\nu_2}\right) \tag{5.1.12}$$

The elongation and scaling parameters σ_i and γ_i are also computed by

$$\sigma_i = \frac{s_1 + \lambda'}{s_2 + \lambda'}, \tag{5.1.13}$$

$$\gamma_i = \left(\frac{s_1 s_2 + \lambda''}{M}\right)^{\frac{1}{2}} \tag{5.1.14}$$

where $\lambda' \approx 0$ and $\lambda'' \approx 0$ are regularization parameters, which dapmpen the effect of the noise, restrict the restrict the ration and square root becoming degenerate and zero, respectively. M is the number of samples in the local analysis window.

Now all C_i have been set, if a Gaussian kernel is choosen, the steering kernel is mathematically represented as

$$K_{H_i^{steer}}(x_i - x) = \frac{\sqrt{det(C_i)}}{2\pi h^2 \mu_i^2} \exp\left\{-\frac{(x_i - x)^T C_i (x_i - x)}{2h^2 \mu_i^2}\right\}. \tag{5.1.15}$$

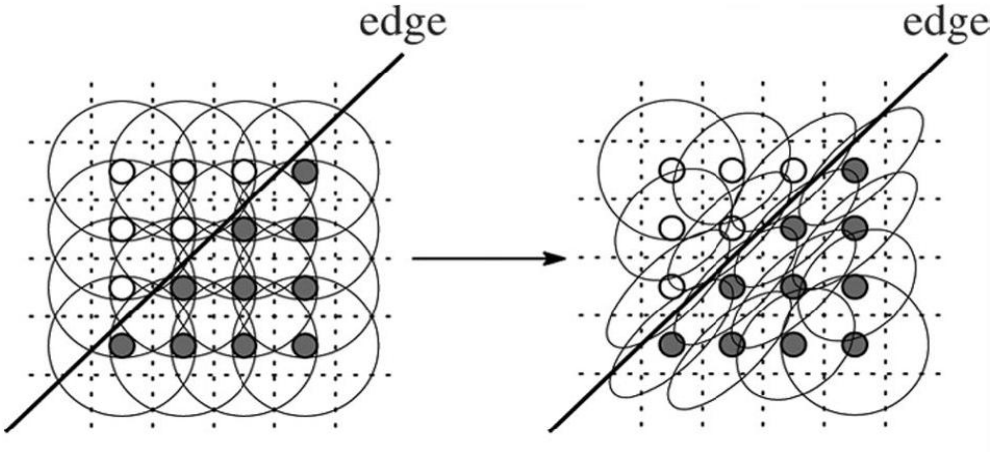


Figure 8: Kernels in the classical local method, footprints are local windows (left); Data-adapted kernels elongate with respect to the edge (right).

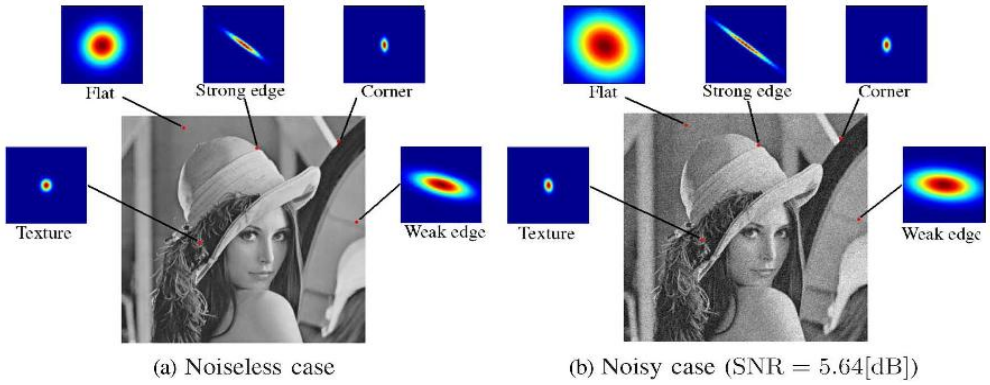


Figure 9: Footprint examples in a clean image (left); Footprint examples at the same position in a noisy image (right).

5.2 Bilateral Steering Kernel Regression

In [33], the steering kernel K_{adapt} is chosen to be $K_{H_i^{steer}}(x_i - x)$, resulting in elongated, elliptical contours spread along the directions of the local edge structure. With these locally adpated kernels, the denoising is effected most strongly along the edges, rather than across them, resulting in strong preservation of details in the final output.

Practically, the steering matrix is determined by the local (noise) information of the image, i.e. the size and shape of the fooprint is determined by the noise information. As the noise is quite soft (the extreme case is a clean image), the

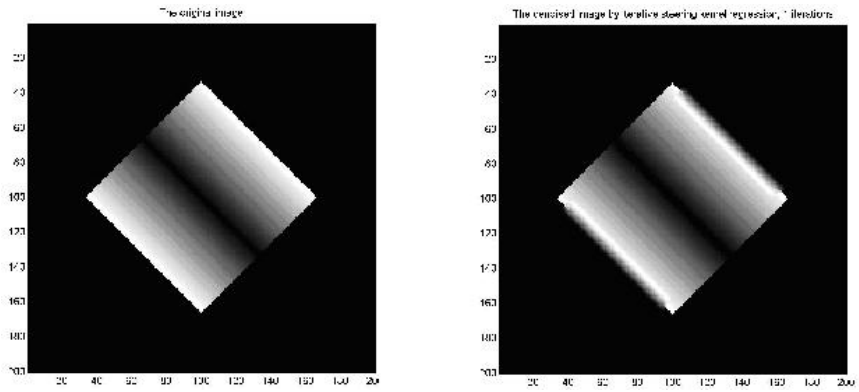


Figure 10: Left: a clean image, notice that the footprint determined by the steering kernel will elongate along with the edge. It can be seen that along with edges on the left top and right bottom (denoted by RT and LB), the regions are homogeneous ($|\frac{du}{dn_1}| \approx 0$, where n_1 is the corresponding direction); while along with the other two edges (denoted by RB and LT), the regions are not ($|\frac{du}{dn_2}| \approx 0$, where n_2 is the corresponding direction); right: Clean image denoised by original steering kernel regression method. Note that the *homogeneous* edges RT and LB are blurred, due to footprints crossing the edges, while the *nonhomogeneous* edges RB and LT are kept sharp.

footprint becomes quite large; when the effect of the steering kernel meets the edge, the footprint will inevitably crosses the edge, although just for a little. See Figure 9.

When noise is quite strong (the extreme case is salt and pepper noise), the footprints in noisy regions become quite small. So the smoothing effect is not good enough. See Figure 16.

So let's consider a bilateral steering kernel instead of the original steering kernel. Let

$$K_{adapt} = K_{H_i^{steer}}(x_i - x) \cdot K_{h_r}(y_i - y), \tag{5.2.16}$$

where

$$K_{h_r}(y_i - y) = e^{-\frac{|y_i - y|^\gamma}{2h_r^2}},$$

and γ is a bilateral parameter. Combining the Gaussian Kernel and Steering matrix

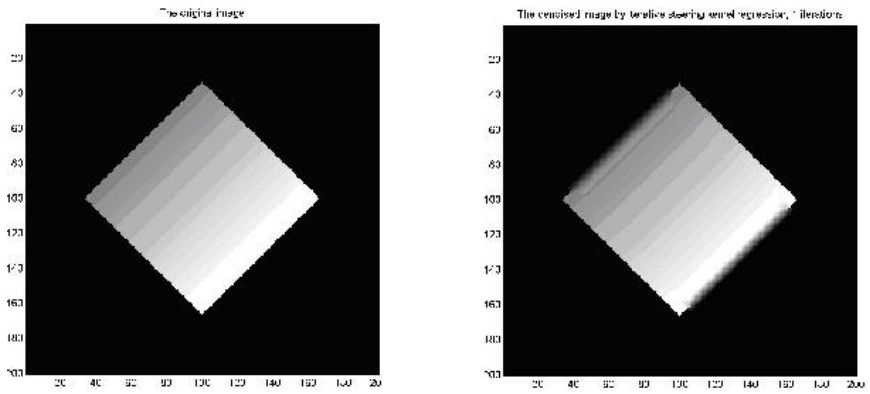


Figure 11: Another example. Left: an clean image; Right: clean image blurred by original steering kernel regression method.

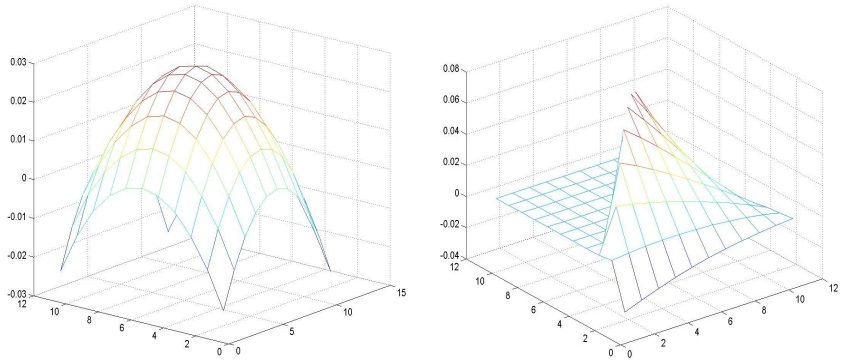


Figure 12: Left: The shape of steering kernel at a pixel in a homogeneous region near the edge in a clean image; Right: the shape of bilateral steering kernel (γ=1.5) at the same pixel.

C_i, we have

$$K_{adapt}(x_i - x, y_i - y) = \frac{\sqrt{det(C_i)}}{2\pi h^2 \mu_i^2} \exp\left\{-\frac{(x_i - x)^T C_i (x_i - x)}{2h^2\mu_i^2}\right\} \cdot \exp\left\{-\frac{|y_i - y|^\gamma}{2h_r^2}\right\}. \tag{5.2.17}$$

After combining this bilateral filter, the footprint close to the edge has been cut along the edge. See Figure 12.

Further more, let's consider an adaptive bilateral steering kernel, in which model, γ is a locally data dependent constant. The idea is, in more homogeneous region, the baliteral effect should be stronger, and in noisy region, the bilateral effect becomes less. There are numerous choices of γ. By experiments, the results are the best when

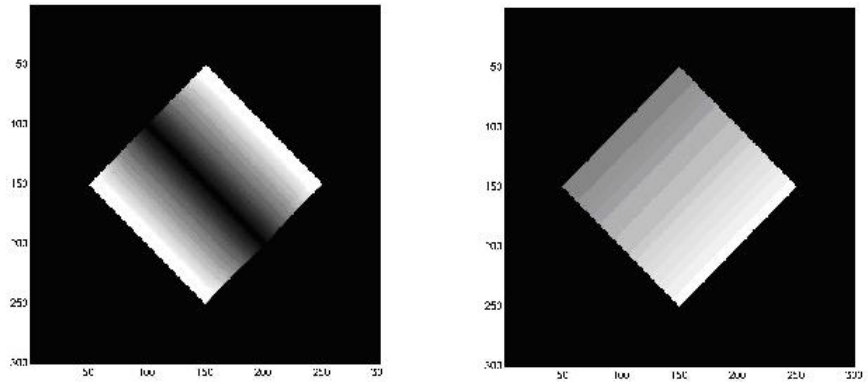


Figure 13: Previous clean figures performed by bilateral steering kernel regression $\gamma = 1.5$. Note that no blurring effect exists.

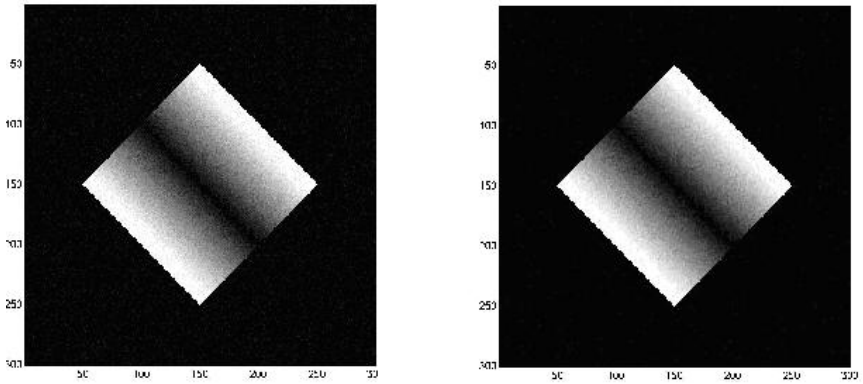


Figure 14: Left: image with gaussian noise, $\sigma^2 = 20$; right: denoised by original steering kernel regression.

$\gamma \in [0.85, 2]$. Generally we choose

$$\gamma = 0.85\sigma + 2(1 - \sigma),$$

where $\sigma \in [0, 1]$ is a parameter related to the local average gradient. When the region is more homogeneous, the local average gradient is smaller so σ is closer to 0. When the region is noisier, the local average gradient is bigger so σ is closer to 1.

Surprisingly, although neither the original bilateral filter nor the original steering kernel method do well to the salt and pepper noise, this adaptive bilateral steering kernel performs very well for such noise. See Figures 16-19.

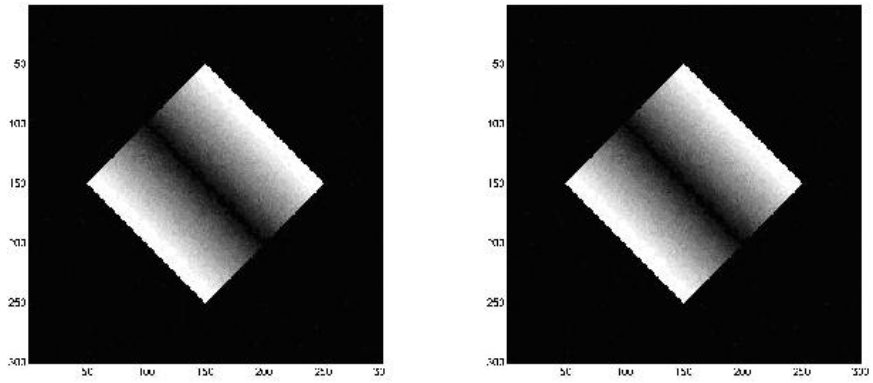


Figure 15: Left: previous gaussian noisy image denoised by Bilateral SKR, with $\gamma = 0.85$; right: same image denoised by Adaptive Bilateral SKR.

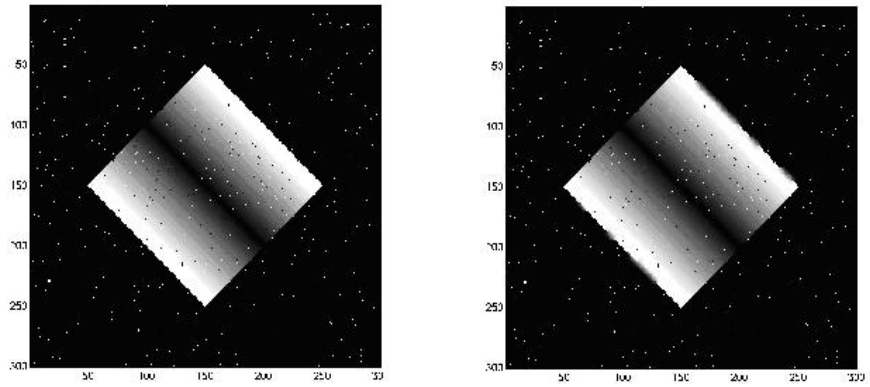


Figure 16: Left: image with salt and pepper noise; Right: denoised by original SKR.

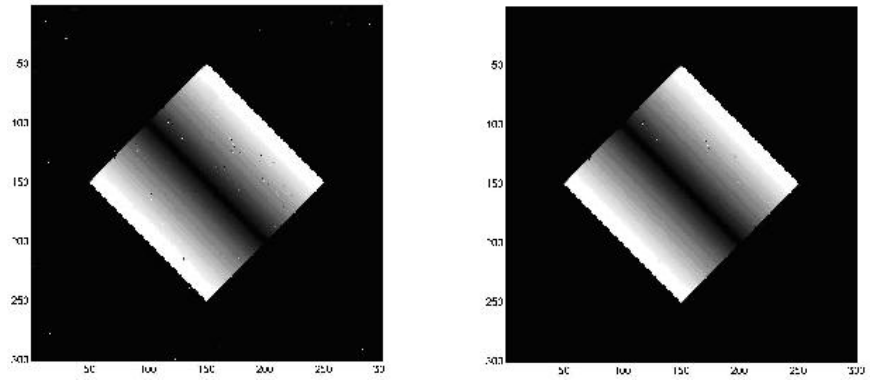


Figure 17: Left: previous image with salt and pepper noise denoised by Bilateral SKR, with $\gamma = 0.85$; Right: same image denoised by Adaptive Bilateral SKR.

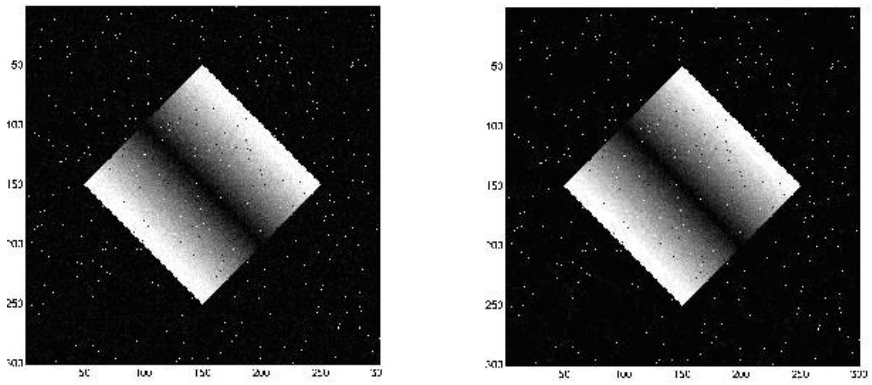


Figure 18: Left: image with gaussian and salt and pepper mixed noise; Right: denoised by original SKR.

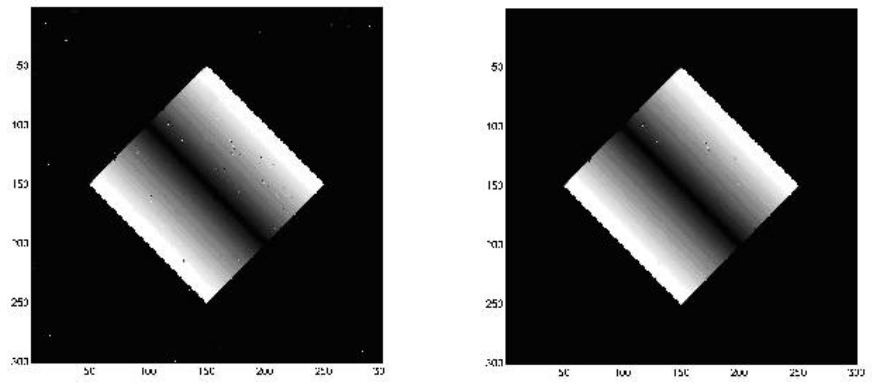


Figure 19: Left: previous image with mixed noise denoised by Bilateral SKR, with $\gamma = 0.85$; Right: same image denoised by Adaptive Bilateral SKR.

CHAPTER VI

NON LOCAL METHODS

In 2006, A.Buades, B.Coll and J.M.Morel developed the Non-local mean method for denoising and it has become quite popular. The idea of the NL-means filter comes from this fact: for each small window that a local smoothing method takes place, it has many similar windows in the same image. The difference of this method from previous adaptive spartial domain filtering methods is that the theory behand this method does not require a locality constraint. The method was further enhanced for speed in subsequent works by Mona Mahmoudi and Guillermo Sapiro in [20]. Different from previous neighborhood filtering methods, this method make use of similar patterns occuring in different parts of a image and use these similarites to denoise.

6.1 Non Local Mean Algorithm

Let $v(i)$ and $u(i)$ be the observed noisy and original images, respectively, where i is the pixel index. The restored values can be derived as the weighted average of all gray values in the image (indexed in the set I)

$$NL(v)(i) = \sum_{j \in I} w(i,j)v(j) \tag{6.1.1}$$

where $NL(v)(i)$ is the restored value at pixel i. The weights express the amount of similarity between the neighborhoods of each pair of pixels involved in the computation (i and j)

$$w(i,j) = \frac{1}{Z(i)} e^{-\frac{\|v(N_i)-v(N_j)\|^2_{2,a}}{h^2}}$$

where $\|.\|$ denotes the Gaussian weighted distance and $Z(i)$ is a normalizing factor $Z(i) = \sum_j w(i,j)$. In the above equation, $v(N_i)$ is the vector of neighborhood pixel

values $v(N_i) := (v(j)), j \in N_i$, where N_i defines the neighborhood of pixel i, normally a square-block of predefined size around i. h is the global smoothing parameter which controls the amount of blurring introduced in the denoising process. For higher values of noise present in an image, h is set to be larger.

Definition 6.1.1. *(Neighborhoods) A neighborhood system on I is a family $N = \{N_i\}_{i \in I}$ of subsets of I such that for all $i \in I$,*
(i) $i \in N_i$
(ii) $j \in N_i \Rightarrow i \in N_j$
Then subset N_i is called the neighborhood or the similarity window of i. The subset N_i will denote $N_i \setminus \{i\}$.

The similarity between two pixels i and j will depend on the similarity of the intensity gray level vectors $v(N_i)$ and $v(N_j)$. The pixels with a similar grey level neighborhood to $v(N_i)$ will have larger weights in the average.

To compute the similartity between two blocks, they use the gaussian weighted Euclidean distance, .

$$\|v(N_i) - v(N_j)\|_{2,a}^2 = (G_a * |v(N_i) - v(N_j)|^2)(0) \tag{6.1.2}$$

It was shown by Efros and Leung that the L^2 distance is a reliable measure for the comparison of image windows in a texture patch [5]. Also this measure is more adapted to any additive white noise. Indeed,

$$E\|v(N_i) - v(N_j)\|_{2,a}^2 = \|u(N_i) - u(N_j)\|_{2,a}^2 + 2\sigma^2 \tag{6.1.3}$$

where u and v are respectively the original and noisy images and σ^2 is the noise variance. This equality shows that, in expectation, the Euclidean distance preserves the order of similarity between pixels. So the most similar pixels to i in v also are expected to be the most similar pixels of i in u.

6.2 Faster Non Local Mean Algorithm

For an image with n pixels, n^2 weights have to be computed for each pixel. Computation n^2 of the overall weights makes the algorithm inefficient and impractical. And so in [2], Morel and Sapiro reduced the total number of computed weighted weights by neglecting in advance neighborhoods with expected small weights. This is done in following three ways:

(a) Fix a number M. For each pixel, only the closest $2M+1$ similar pixels are choosed to calculate weights.

(b) The further consideration of similarity are determined by an inequality $\eta_1 < (\overline{v}(i)/\overline{v}(j)) < \eta_2$, where $\overline{v}(i)$ and $\overline{v}(j)$ are the average gray values in the neighborhoods of pixels in i and j, and $\eta_1 < 1$ and $\eta_2 > 1$ are two constants close to 1. Only pixels satisfying this inequality will be considered to calculated weights.

(c) Another method to approximate the similarity between two neighborhoods is to compute $\overline{\bigtriangledown v}(i) = (\overline{v_x}(i), \overline{v_y}(i))$, where $\overline{v_x}(i)$ and $\overline{v_y}(i)$ are the average horizontal and vertical derivatives in the neighborhood of pixel i. Define $\theta(i,j) = \angle(\overline{\bigtriangledown v}(i), \overline{\bigtriangledown v}(j))$ and $\sigma_\theta = 1.4826 median_{I\times I}[|\theta(i,j) - median_{I\times I}(|\theta(i,j)|)|]$, where $\theta(i,j)$ is angle between the average gradient directions. The weight $w(i,j)$ is computed(nonzero) if the gradient in pixel i or j are small or $\theta(i,j) < \sigma_\theta$.

(a) and (b) make the closest blocks be easily accessed with $O(1)$ complexity lookup table addressed by the average gray value of the neighborhood for the current pixel being processed. (c) makes the angle between the average gradient directions at the corresponding pixels to be a measure to filter out unrelated neighborhoods, where the threshold is choosed by a statistical result following [23] and [24].

So in Sapiro's model, the wight $w(i,j)$ is calculated as the following equation:

$$w(i,j) = \begin{cases} \frac{1}{Z(i)} e^{-\frac{\|v(N_i)-v(N_j)\|_a^2}{h^2}}, & [(\| \bigtriangledown v(i)\| < \sigma_{\bigtriangledown}) \\ & \text{or } (\| \bigtriangledown v(j)\| < \sigma_{\bigtriangledown}) \\ & \text{or } (\theta(i,j) < \sigma_\theta)] \\ & \text{and } (\eta_1 < (\overline{v}(i)/\overline{v}(j)) < \eta_2) \\ 0, & \text{otherwise} \end{cases} \tag{6.2.4}$$

This effectively reduced the computational complexity of the original algorithm.

6.3 Nonlocal Bilateral Filter

In this section, a modified nonlocal bilateral fiter is suggested to preclassify the image blocks and thereby reduce the number of weight computations in the global window size denoising algorithm. The idea of this algorithm comes from the original NLM filter and the original Bilateral filter. In both methods, for each pixel, the *related* pixels are used to build the weights so to smooth it. In the original NLM filter, the pixels with close intensity distance from the whole image are considered as the *related pixels*, which does not take any pixels with close spatial distance into account. Some color leaking problems will occur due to this fact, mostly happened in a color image(See Figure 22). In the original bilateral filter, pixels with close intensity distance and close spatial distance are both considered, but fixed in a small local window, which somehow neglects the affect of pixels with close grey values but spatially far from center of the local window. What's more, how to choose a window size of a bilateral filter is still a problem. When one chooses a small size window, the denoising effect is not good enough, but when one chooses a big size window, too many weights of (unrelated) pixels computed will not only increase the complexity of the algorithm but also make the image look unreal and thus the SNR decreases.

In the example, the original size of the image is 512×512. We keep increasing the denoising window of the original bilateral image to see how well it works. Actually,

when the window size is 5×5, the SNR reaches the maximum 36.9857, with the running time $t = 16.953s$; after that, the SNR starts dereasing, and the running time becomes longer and longer. When we use a 20×20 window, the SNR becomes 32.587, and it takes $37.484s$; when we use a 50×50 window to denoise, the SNR has been reduced to 32.2007, and it takes $156.689s$. See Figure 21.

So let's consider:

$$NLB(v)(i) = \frac{1}{C(i)} \sum_{j \in \Omega_i} w_1(i,j) w_2(i,j) v(j) \tag{6.3.5}$$

where $w_1(i,j) = e^{-\frac{(v(i)-v(j))^2}{h_1^2}}$, $w_2(i,j) = e^{-\frac{\|i-j\|^2}{h_2^2}}$ and

$$\Omega_i = \{v(j) || v(i) - v(j)| < \sigma\}. \tag{6.3.6}$$

The advantage of this algorithm is, it considers the relevance of both pixels closed in spatial domain and intensity domain. The complexity of this algorithm relies on the choice of Ω_i. For (6.3.6), it is easy to estimate the method noise:

Proposition 6.3.1. *Suppose u is a Lipschitz bounded function on Ω, where Ω is an open and bounded domain of R^2, then $|u(x) - NLB_{\sigma_r,\sigma_d,\sigma} u(x)| < \sigma$.*

Proof. Let x be a point of Ω, then

$$\begin{aligned} |u(x) - NLB_{\sigma_r,\sigma_d,\sigma} u(x)| &= \frac{1}{C} \int_{\Omega_i} W_1 W_2 |u(y) - u(x)| dy \\ &\leq \frac{1}{C} \int_{\Omega_i} W_1 W_2 \cdot \sigma dy \\ &= \frac{\sigma}{C} \int_{\Omega_i} W_1 W_2 dy \\ &= \sigma \end{aligned} \tag{6.3.7}$$

where $W_1 = e^{-\frac{|u(y)-u(x)|^2}{\sigma_r^2}}$, $W_2 = e^{-\frac{\|y-x\|^2}{\sigma_d^2}}$.

□

Figure 20: Left Top: original lena image; right top: lena with additive gaussian noise $\sigma^2 = 25$; left bottom: denoised by original non local mean filter, 1 iteration, $SNR = 36.0472$; right bottom: denoised by non local bilateral filter, 1 iteration, $SNR = 37.1205$.

To make this algorithm fast and efficient, a more wise choice of Ω_i goes to the contribution of Sapiro's blocks preclassfication. We choose the $2M + 1$ closest similar pixels of $v(j)$ to compute weights. Here we use a Gaussian weighted average pixel value to compute distance. This change significantly reduces the complexity of computation and enhance quality of the image.

Figure 21: Lena image denoised by original bilateral filter with window size increasing. Left Top:5×5, SNR=36.9857, running time 16.953s; 20×20, SNR=32.587, running time 37.484s; left bottom: 50×50, $SNR = 32.2007$, running time 156.689s; right bottom: 100×100, $SNR = 32.2007$, running time 591.001s.

Figure 22: Left top: original chicken image; right top: chicken image denoised by NLM filter, 3 iterations. Note that color leaking appears on the hat of the pen; left bottom: chicken image denoised by NL Bilateral filter, 3 iterations; right bottom: chicken image denoised by original bilateral filter, 3 iterations.

CHAPTER VII

CROSS-CHANNEL PARADIGM

7.1 Color Image With Real Noise

In real-world scenarios, noise in color images comes from many sources, such as the underlying physics of the imaging sensor itself, sensor malfunction, flaws in the data transmission procedure, and electronic interference. Due to the complex nature of the noise process, the overall acquisition noise is usually modeled as a zero mean white Gaussian noise. Aside from this, image imperfections resulting from salt and pepper noise are generated during transmission through a communication channel, with sources ranging from human-made to natural. Thus, noise corruption process in simulated scenarios is usually modeled using additive Gaussian noise, salt and pepper noise, or mixed noise.

Real images are corrupted by real, non-approximated noise which may be different in characteristics and statistical properties depending on application.

One may argue that color images are no difference from three monochromatic images once we consider the three channels separately, and therefore to denoise a color photo one only needs to denoise the three monochromatic channels separately. This view, however, misses some important subtle characteristics in naturally captured color images that, when fully utilized, yield superior results. To denoise color photos we must understand the nature of the noise in these images, and take full advantages of all available informations.

7.2 Color Space Decomposition

A color model is an abstract mathematical model describing the way colors can be represented as tuples of numbers, typically as three or four values or color components. The three most popular color models are RGB, Lab(used in computer graphics); YIQ, YUV or YCrCb(used in video systems) and CMYK(used in color printing).

All of the color spaces are derived from the RGB information captured by devices such as cameras and scanners.

7.2.1 RGB Color Space

The red, green, and blue (RGB) color space is widely used throughout computer graphics. Red, green, and blue are three primary additive colors and are represented by a 3D coordinate system. The RGB data captured by digital cameras and scanners is the raw data and has the most information. This color space is the most prevalent choice for computer graphics because color displayes use red, green, and blue to creat the desired color. RGB is a convenient color model for computer graphics because the human visual system works in a way that is similar though not quite identical to an RGB color space. The most commonly used RGB color spaces are sRGB and Adobe RGB (which has a significantly larger gamut). Adobe has recently developed another color space called Adobe Wide Gamut RGB, which is even larger, in detriment to gamut density.

7.2.2 CIE XYZ Color Space

In the study of the perception of color, one of the first mathematically defined color spaces was the CIE 1931 XYZ color space (also known as CIE 1931 color space), created by the International Commission on Illumination (CIE) in 1931 [34, 35].

The human eye has receptors (called cone cells) for short (S), middle (M), and long (L) wavelengths. Thus in principle, three parameters describe a color sensation.

Figure 23: A color image(left top) and the G(left bottom), B(right top) and R(right bottom) elements. Note that the Green channel is the cleanest channel and the Blue channel has most noise

The tristimulus values of a color are the amounts of three primary colors in a three-component additive color model needed to match that test color. The tristimulus values are most often given in the CIE 1931 color space, in which they are denoted X, Y, and Z [36].

CIE XYZ color space is special because it is based on direct measurements of human visual perception, and serves as the basis from which many other color spaces are defined.

Mathematically, CIE XYZ is a linear transformation of RGB:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \frac{1}{0.17697} \begin{bmatrix} 0.49 & 0.31 & 0.20 \\ 0.17697 & 0.81240 & 0.01063 \\ 0.00 & 0.01 & 0.99 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

7.2.3 Lab Color Space

A Lab color space is a color-opponent space with dimension L for lightness (or luminance) and a and b for the color-opponent (or chrominance) dimensions, based on a nonlinear transformation of CIE XYZ color space.

The most advantage of Lab color space is, it is *perceptually uniform*, which means that a change of the same amount in a color value should produce a change of about the same visual importance. Thus, Lab color is designed to approximate human vision. It aspires to perceptual uniformity, and its L component closely matches human perception of lightness. It can thus be used to make accurate color balance corrections by modifying output curves in the a and b components, or to adjust the lightness contrast using the L component.

Mathematically, Lab is a nonlinear transformation of XYZ, thus it is nonlinear of

Figure 24: A color image(left top) and the L(left bottom), a(right top) and b(right bottom) elements. Note that the L channel is the luminance channel (cleanest) and the a and b channels has much more noise

RGB:

$$L = 116f(Y) - 16$$

$$a = 500[f(X) - f(Y)]$$

$$b = 200[f(Y) - f(Z)]$$

where

$$f(t) = \begin{cases} t^{1/3} & \text{if} \quad t > 0.008856 \\ 7.787t + 16/116 & \text{otherwise} \end{cases}$$

7.2.4 YCrCb Color Space

YCrCb color space decomposition is also a Luminance-Chrominance decomposition. Y is the luminance component and Cb and Cr are the blue-difference and red-difference chromainance components. It is offen used as a part of the Color image pipeline in video and digital photography systems. Due to the noise distribution in different channels of a digital color image, which will be discussed in next section, YCrCb is experimentaly the best decomposition in image denoising.

Mathematically, YCrCb is also a linear transformation of RGB. The basic equation to convert between 8-bit digital RGB data with a 16-235 nominal range YCrCb is:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.172 & -0.339 & 0.511 \\ 0.511 & -0.428 & -0.083 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix}$$

7.3 Noise Distribution in Digital Color Images

7.3.1 Natural Noise and Artificial Noise

Due to the similarity between the amplified noise and gaussian noise, many of the studies focus on normal images with artificially added Gaussian noise (or sometimes salt and pepper noise). The results are often either misleading or not optimal. Color images captured by digital cameras are often noisy, but the noise profile is nothing like those that have been added artificially [44].

It has been well studied that, for a grey value image, the natural noise distribution is quite different between light areas and dark areas. The dominant noise in the lighter parts of an image from an image sensor is typically caused by statistical quantum fluctuations, that is, variation in the number of photons sensed at a given exposure level; this noise is known as photon shot noise [45]. Shot noise has a Poisson distribution, which is usually not very different from Gaussian.

Figure 25: A color image(left top) and the Y(left bottom), Cb(right top) and Cr(right bottom) elements.

Figure 26: clean image(left top) and image with natural noise(right top), image with additive gaussian noise(left bottom) and image with additive salt and pepper noise(right bottom), note that artificial noise distributions are homogeneous while the natural noise is not.

Figure 27: A line is picked in a grey value image to see the noise distribution

While there is additional shot noise from the dark leakage current in the image sensor; this dark area noise is sometimes known as *dark shot noise* [45] or *dark-current shot noise* [46]. Dark current is greatest at *hot pixels* within the image sensor; the variable dark charge of normal and hot pixels can be subtracted off, leaving only the shot noise, or random component, of the leakage [47, 48]; if the exposure time is long enough that the hot pixel charge exceeds the linear charge capacity, the noise will be more than just shot noise, and hot pixels appear as salt-and-pepper noise.

Generally, we have a comparison between the natural noise and artificial noise:
(1) Additive artificial noise is homogeneous in the whole image, while real noise is often not, especially in dark areas;
(2) There is much more real noise in dark areas than in light areas;
(3) Additive artificial noise is independent from pixel to pixel, while real noise is often non-independent from pixels.

To see it more clearly, we pick a line from the image and see the noise distribution in a 2D way. See Figures 27-29.

7.3.2 Bayer Pattern and Noise Distribution in Color Images

Bayer pattern is known as a filter arrangement in an image sensor that captures

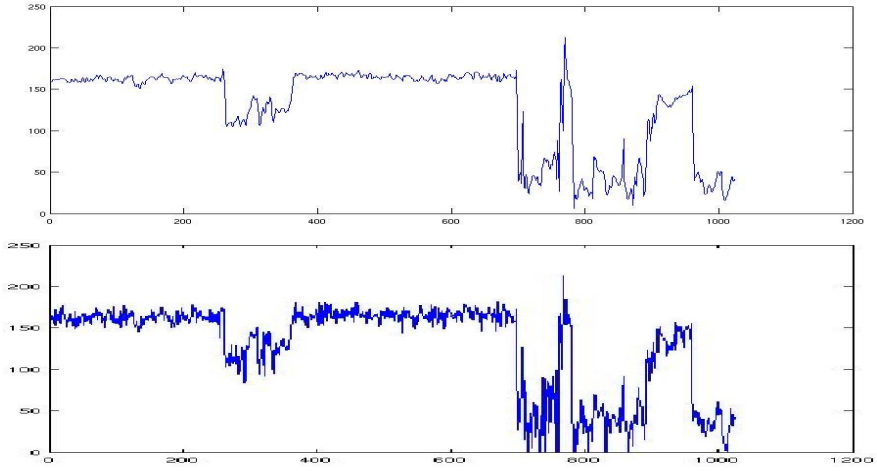


Figure 28: Pixel value in a clean image and an image with natural noise. Note that the noise is much more in the dark area and distributes non-homogeneously.

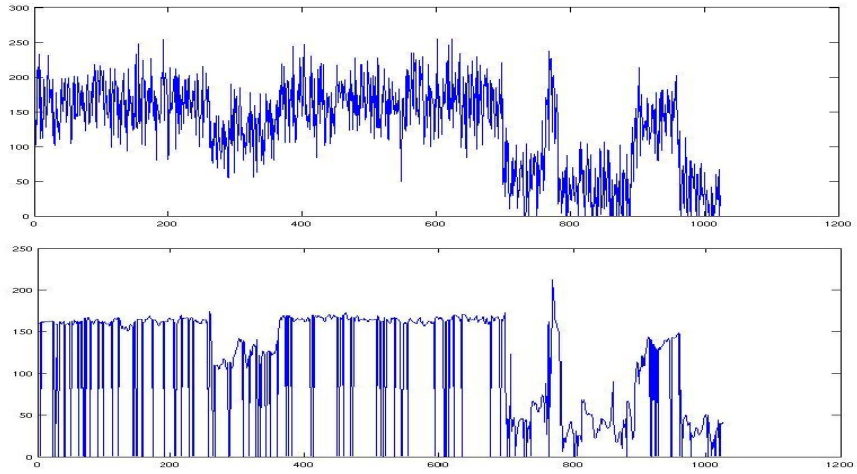


Figure 29: Pixel value in an image with additive gaussian noise (top) and salt and pepper noise (bottom). Note that the noise distributions in both images are homogeneous.

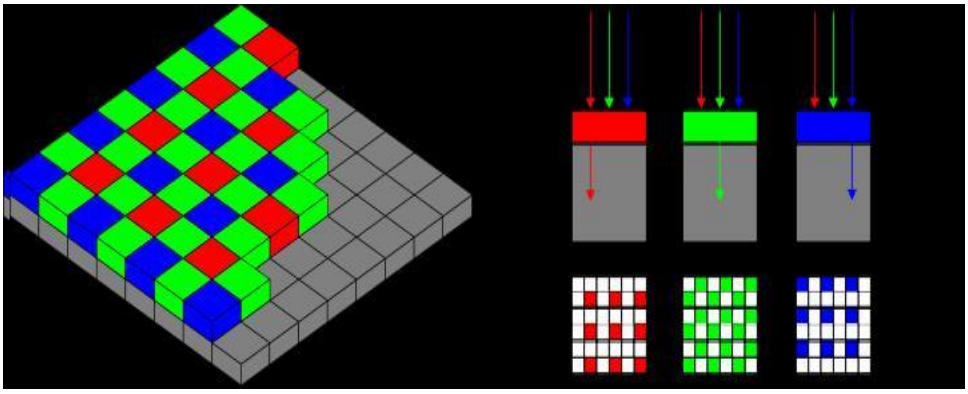


Figure 30: Bayer Pattern Filter Arrangement

natural light in RGB model. In a Bayer filter arrangement, green is given twice as many sensors as red and blue in order to achieve higher luminance resolution than chrominance resolution, see Figure 30. For every channel, missing pixels are obtained by interpolation in the demosaicing process to build up the complete image.

In this case, the green channel will be cleaner than the red channel and the blue channel. What's more, due to the reason that more amplification is used in the blue color channel than in the green or red channel, the blue channel is always the noisiest [45]. For the same reason, in a luminance-chrominance color space decomposition, the luminance channel is always cleaner than the chrominance channel. Since the cleaner channel(s) have better geometric properties (which are not annoyed by noise that much), it is natural to think that, we can use cleaner channel(s) as a *standard* in denoising process to work on noisier channels.

7.4 Discription of Cross-channel Paradigm

For a given 3-channel color space decomposition $(\vec{i}, \vec{j}, \vec{k})$, a color image

$$\vec{u}(x) = u_1(x)\vec{i} + u_2(x)\vec{j} + u_3(x)\vec{k}, \tag{7.4.1}$$

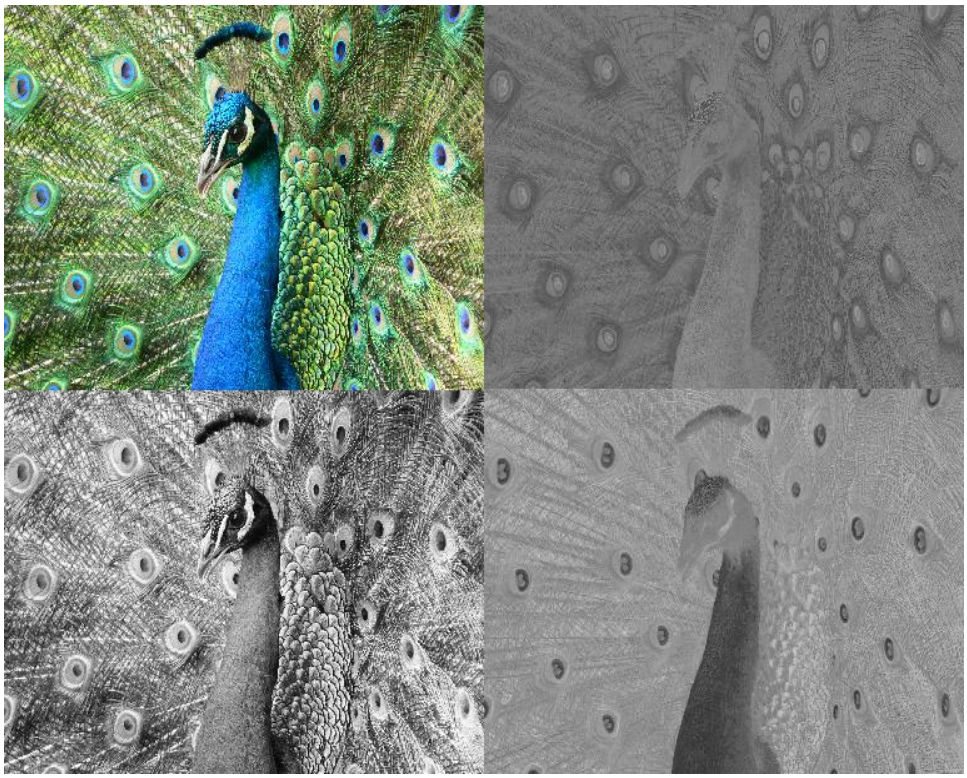


Figure 31: Lab Decomposition of a peacock image. From left to right, top to bottom: original image, a channel, L channel, b channel. Note that L channel is clean.

and a cross-channel paradigm will be

$$\begin{aligned} CR(\vec{u}(x)) &= CR(u_1(x)\vec{i} + u_2(x)\vec{j} + u_3(x)\vec{k}) \\ &= CR_1(u_1(x))\vec{i} + CR_2(u_2(x))\vec{j} + CR_3(u_3(x))\vec{k} \end{aligned} \tag{7.4.2}$$

where, CR_1, CR_2 and CR_3 are different but correlated filters defined on 3 channels.

For example, a naive algorithm using Lab decomposition and Gaussian filter is:

$$\begin{aligned} CR(\vec{u}(x)) &= CR(u_L(x)\vec{i} + u_a(x)\vec{j} + u_b(x)\vec{k}) \\ &= u_L(x)\vec{i} + G_{h_1}(u_a(x))\vec{j} + G_{h_2}(u_b(x))\vec{k}. \end{aligned} \tag{7.4.3}$$

Since L is a luminance channel, it is almost clean. We use Gaussian blur to denoise the chrominance channels a and b. See Figure 31 and 32.

One thing that needs to be paid attention is, the advantage of separating luminance from chrominance is that human vision is typically less sensitive to diffusion in

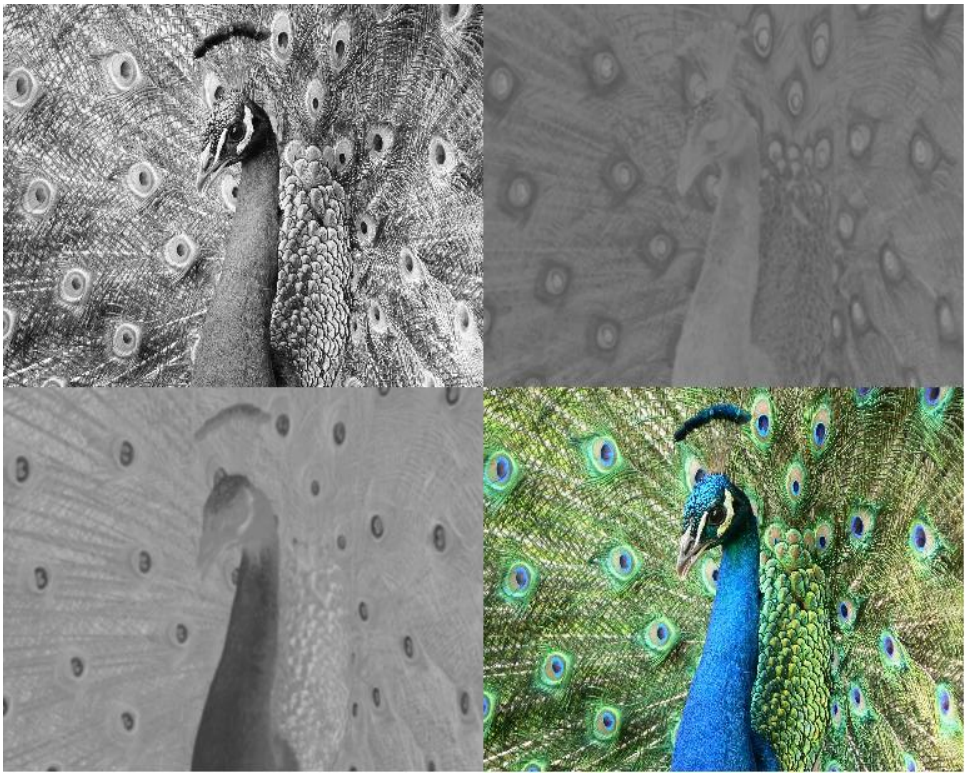


Figure 32: From left to right, top to bottom: L channel remained as before, a channel blurred by Gaussian filter with radius 5, b channel blurred by Gaussian filter with radius 5, recomposed image.

Figure 33: YCrCb Decomposition of an image with real noise. From left to right, top to bottom: original image, Cr channel, Y channel, Cb channel. Note that Y channel is clean.

chrominance [44]. which means, when we take a strong blurring effect on chrominance channels (a, b in Lab decomposition or Cr Cb in YCrCb decomposition), after recomposition, there is a very little discernible difference from the original image (except for denoising). However, to the luminance channel (L or Y), it is a different story. In fact, even a tiny blurring in the luminance channel will be immediately visible in the recomposed color image. Given these characteristics of the luminance-chrominance decomposition, we would be more aggressive in denoising the chrominance channels while less so in denoising the luminance channel. See Figure 33 and 34.

7.5 A New YCrCb Decomposition for Denoising

In the standard luminance-chrominance decomposition, such as Lab and YCrCb, the luminance is *contaminated* by the blue channel, where noise concentrates as we

Figure 34: From left to right, top to bottom: original Y channel; using wavelet thresholding to severely blur the Cr channel, note that most details are removed; using wavelet thresholding to severely blur the Cb channel, similar as Cr, most details are removed; the recomposed color image, note that the blurring Cr Cb channels do not the sharpness of the original color image.

have pointed out earlier. As a result, the luminance channel can be somewhat noisy, and therefore substantial denoising will have to be performed on it. This can be adversely affect the quality of the denoised color image. To get around this problem we introduce a new color space, the *modified YCrCb* color space ($mYCrCb$). Different from the original YCrCb color space, the *luminance* channel Y in $mYCrCb$ is a linear combination of only the green and the red channels. More precisely, the $mYCrCb$ color space is obtained via a linear transform from RGB space:

$$\begin{aligned} Y_m &= 0.666G + 0.334R, \\ Cr_m &= (R - Y)/1.6, \\ Cb_m &= (B - Y)/2. \end{aligned}$$

7.6 Cross-channel ENO/WENO schemes

This scheme is particularly designed based on the cross-channel idea. By color space decomposition, we have some clean channels and some noisy channels. There are several kinds useful information that clean channels can provide, i.e. geometric property, contrast, the feature that it is in a better functional space, etc. But the most useful information is: it provides the most accurate edges. All denoising filters are facing a same problem: how to preserve the sharp edge while blurring the noise? Now we can take the accurate edges from the given color image and then perform a blurring filter without cross the edge.

There are a lot of edge detectors in 2D images, i.e. *Canny, LoG, Zero-crossing....* Here we use J. Canny's method (1986 [49]), which is the most widely used algorithm in image processing. In matlab, it provides the edges corresponding to a threshold γ, which controls the strength of the edges. See Figure 35.

Define

$$CD(u,\gamma)(x) = \begin{cases} 1 & \text{if x is Canny}'\text{s Edge with threshold } \gamma, \\ 0 & \text{otherwise.} \end{cases} \tag{7.6.4}$$

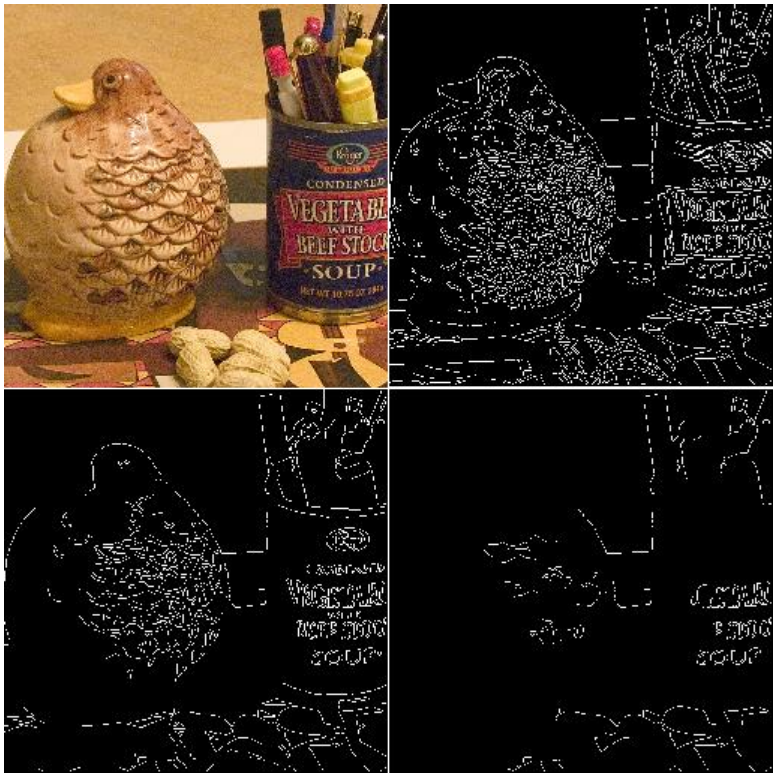


Figure 35: Canny's Edge Detection. Left top: original image with real noise; Right top: Canny's edge with threshold 0.15; Left bottom: Canny's edge with threshold 0.40; Right bottom: Canny's edge with threshold 0.15.

And let's define an operator between two matrces with same dimensions. For $A = (a_{ij})_{i,j=1,\dots,n}$, $B = (b_{ij})_{i,j=1,\dots,n}$,

$$A \odot B = (a_{ij} \cdot b_{ij})_{i,j=1,\dots,n}.$$

Using $mYCrCb$ color space and Gaussian blur (inside the edge), the cross-channel ENO scheme (CE) is defined as:

$$\begin{aligned} CE(\vec{u}) &= CE(u_Y\vec{i} + u_{Cr}\vec{j} + u_{Cb}\vec{k}) \\ &= CE_1(u_Y)\vec{i} + CNL_2(u_{Cr})\vec{j} + CNL_3(u_{Cb})\vec{k}, \end{aligned} \tag{7.6.5}$$

where $CE_1(\cdot) = Identity$, and

$$CE_2(u_{Cr}) = \tilde{G_{h_r}}(u_{Cr} \setminus (u_{Cr} \odot CD(u_Y, \gamma_r))),$$

$$CE_3(u_{Cb}) = \tilde{G_{h_b}}(u_{Cb} \setminus (u_{Cb} \odot CD(u_Y, \gamma_b))).$$

The explanation of $\tilde{G_h}(A \setminus B)$ should be made here: A and B are both given matrices and $B \subset A$, where B is the set of edges of A. $\tilde{G_h}(A \setminus B)$ means he locally gaussian filter (with parameter h) is taken on A without crossing edges in B. In details, in a local window of A, denoted by I, if the center i is on an edge in B, e.g. $i \in B$, then nothing will be done in I; if $i \notin B$, and if there exists edge points in I, then the gaussian filter will be taken at i without crossing edges. See Figure 36.

For some case, soft edges also need to be blurred a little bit, to make the whole image look smooth. So we take full consideration of the strength of edges, i.e. each pixel in the image is taken as an edge with a weight, we will have a $WENO$ scheme.

Definition 7.6.1. *(Pixel Edge Weight):*

$$WCD(u,\gamma)(x) = \begin{cases} \lambda(|u_x|^2 + |u_y|^2 + |u_z|^2), & \text{if x is Canny}'\text{s Edge with threshold } \gamma, \\ 0, & \text{otherwise,} \end{cases} \tag{7.6.6}$$

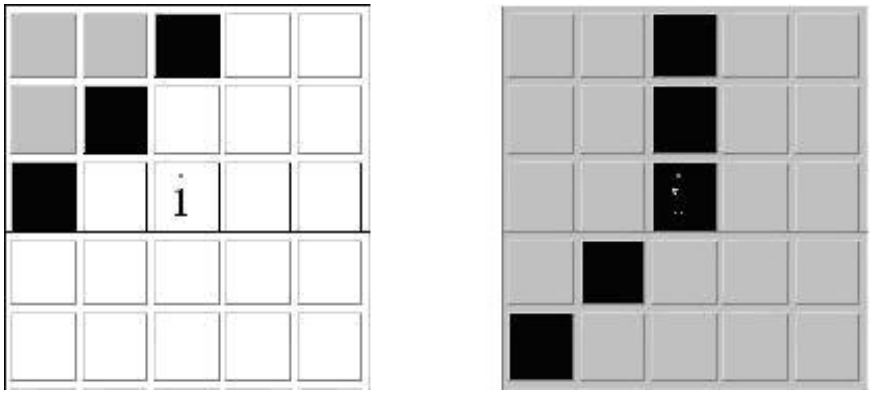


Figure 36: $\tilde{G_h}$ filter taken on a local 5×5 window. i: center point; Black areas: edge; white areas: effective when taken the gaussian filter; grey areas: ineffective when taken gaussian filter; Left: the center i is not on an edge; Right: the center i is on an edge.

where λ is an adjustment constant and

$$u_x = u(i,j) - u(i-1,j), \quad u_y = u(i,j) - u(i,j-1), \quad u_z = u(i,j) - u(i-1,j-1).$$

So for WENO scheme,

$$\begin{aligned} CWE(\vec{u}) &= CWE(u_Y\vec{i} + u_{Cr}\vec{j} + u_{Cb}\vec{k}) \\ &= CWE_1(u_Y)\vec{i} + CWE_2(u_{Cr})\vec{j} + CWE_3(u_{Cb})\vec{k}, \end{aligned} \tag{7.6.7}$$

where

$$CWE_1(u_Y) = \tilde{G}_{h_Y}(u_Y \setminus (u_Y \odot CD(u_Y, \gamma_Y))) + G_{h_Y, WCD_{u_Y,\gamma_Y}(x)}(u_Y)(x),$$

$$CWE_2(u_{Cr}) = \tilde{G}_{h_{Cr}}(u_{Cr} \setminus (u_{Cr} \odot CD(u_Y, \gamma_{Cr}))) + G_{h_{Cr}, WCD_{u_Y,\gamma_{Cr}}(x)}(u_{Cr}),$$

$$CWE_3(u_{Cb}) = \tilde{G}_{h_{Cb}}(u_{Cb} \setminus (u_{Cb} \odot CD(u_Y, \gamma_{Cb}))) + G_{h_{Cb}, WCD_{u_Y,\gamma_{Cb}}(x)}(u_{Cb}),$$

where $G_{h,\beta}$ is the gaussian filter, with radius h and $\sigma = \beta$.

In the above formula, the gaussian filter is taken on the non-edge pixels without crossing the edges, and for each edge pixel, it is blurred slightly or strongly, depending on the weakness of the edge. In this case, the noise on soft edges would also be removed and strong edges are still kept sharp. See Figures 37-39.

Figure 37: Left: chicken image denoised by ENO scheme, performed in $mYCrCb$ space, 3 iterations; Right: 100 % crop of the hat area.

Figure 38: Left to right: chicken image denoised by ENO and WENO schemes, both performed in $mYCrCb$ space, 3 iterations.

Figure 39: 100% crop of the above images. Note that edges in the right image are more smooth than in the left one.

7.7 Cross-channel Non Local Means Filter

In this section, we will see how cross-channel paradigm is performed upon the Non Local Filter. Let

$$\vec{u}(x) = u_Y(x)\vec{i} + u_{Cr}(x)\vec{j} + u_{Cb}(x)\vec{k},$$

where $\vec{i}, \vec{j}, \vec{k}$ is the $mYCrCb$ color space decomposition. The

$$\begin{aligned} CNL(\vec{u}(x)) &= CNL(u_1(x)\vec{i} + u_2(x)\vec{j} + u_3(x)\vec{k}) \\ &= CNL_1(u_Y(x))\vec{i} + CNL_2(u_{Cr}(x))\vec{j} + CNL_3(u_{Cb}(x))\vec{k} \end{aligned} \tag{7.7.8}$$

Note that, in NLM filter,

$$NL(u)(i) = \sum_{j \in I} w(i,j)u(j)$$

where

$$w(i,j) = \frac{1}{Z(i)} e^{-\frac{\|u(N_i) - u(N_j)\|^2_{2,a}}{h^2}}.$$

Now let

$$CNL_1(u_Y)(i) = \sum_{j \in I} w_{Y,h_0}(i,j)u_Y(j),$$

$$CNL_2(u_{Cr})(i) = \sum_{j \in I} w_{Y,h_1}(i,j)u_{Cr}(j),$$

$$CNL_3(u_{Cb})(i) = \sum_{j \in I} w_{Y,h_2}(i,j) u_{Cb}(j),$$

where

$$w_{Y,h}(i,j) = \frac{1}{Z(i)} e^{-\frac{\|u_Y(N_i) - u_Y(N_j)\|_{2,a}^2}{h^2}}. \tag{7.7.9}$$

Note that Y channel has the best geometric property, which rarely annoyed by noise, we may perform the filter very slightly, so we choose a small h_0. For Cr, Cb channels, we choose larger h_1 and h_2 to obtain strong blurring effect. We also notice that in CNL_2 and CNL_3, the weights were computed by grey values in Y channel, which means the similarity of blocks will be determined by Y channel, and then performed in Cr and Cb channels.

One thing that needs to be pointed out is, images denoised by the original NLM filter might have a color leaking problem, which was discussed in previous chapters. The reason is, it only used the pixels closed in grey values (as related pixels) to modify the image (without considering the the effect of distance closed pixels). It was also pointed out by the authors in [16], that the best performance of this filter will obtain is when denoising images with periodic patterns, since many similar blocks could be found and used in denoising. But this will bring another problem, for images without many periodic patterns, it smoothes (sometimes over-smoothes) the homogeneous regions quite well, but for a region with subtle details and a lot of noise, (frequently happened in a dark region), the performance is not that good. Fortunately, extended by the cross-channel paradigm, a slight blurring effect on Y channel will preserve enough details while the aggressive blurring in Cr and Cb channels will remove the noise as much as possible. See Figure 42 and 43.

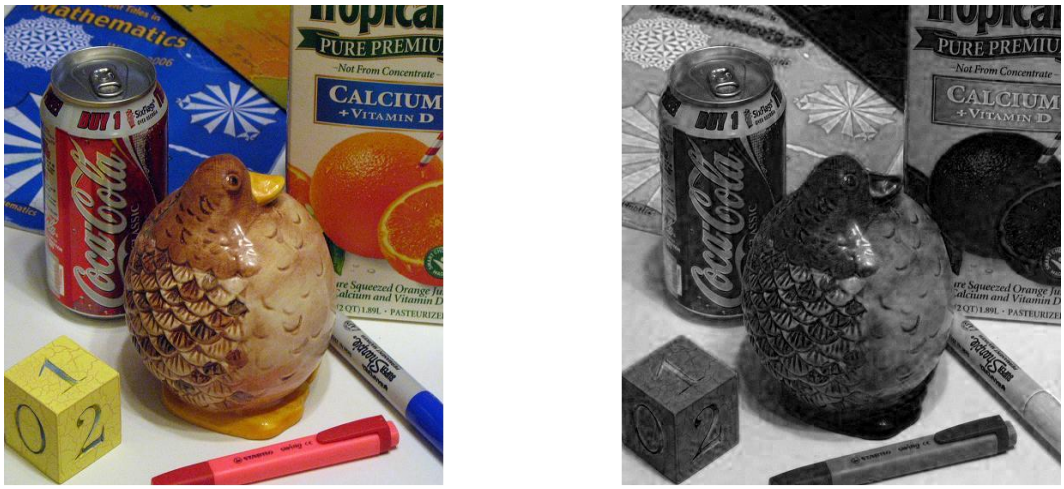


Figure 40: A chicken image with real noise and its blue channel.

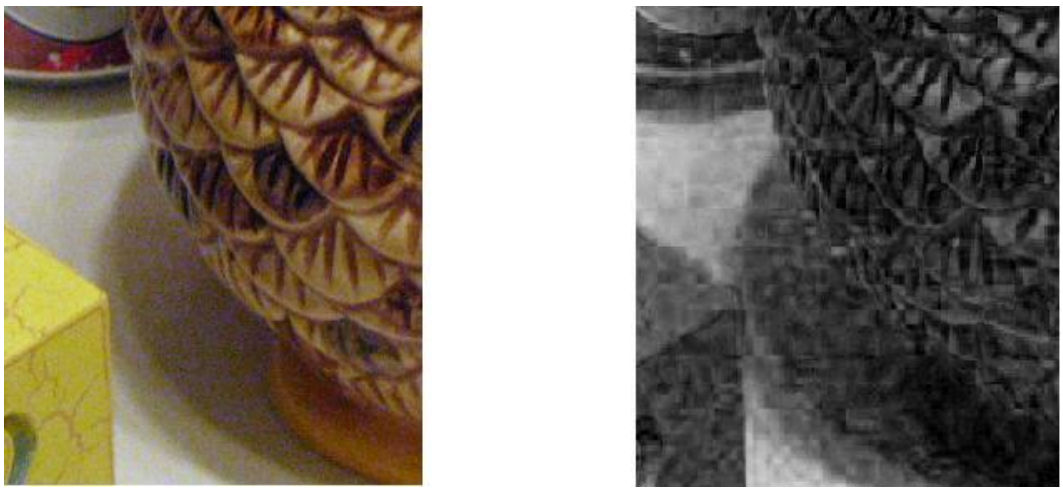


Figure 41: Left: 100% crop part from a chicken image. Right: blue channel of this image.

Figure 42: Left to right: the crop area denoised by the original NLM filter (3 iterations) and cross -channel NLM fiter (3 iterations), respectively. Note that the both are clean, but the left one is a little bit over-smoothed.

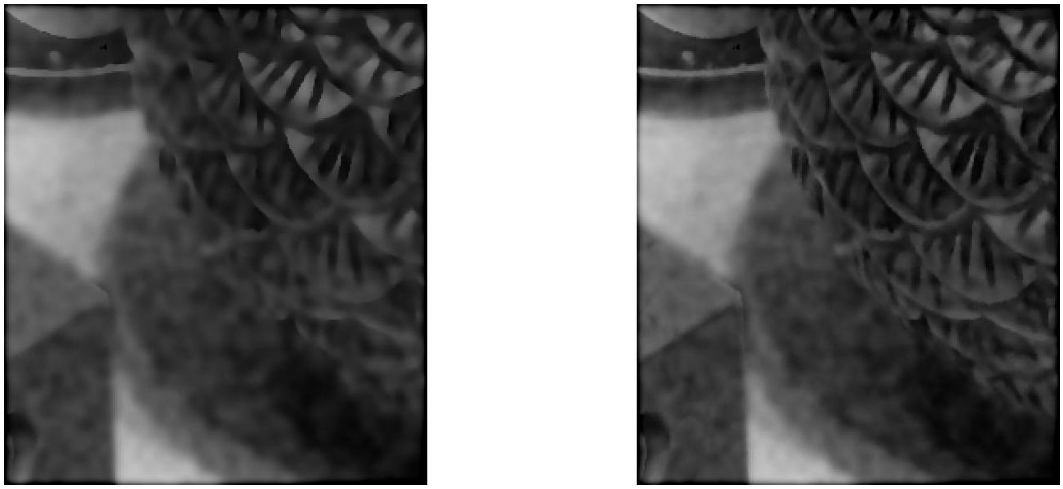


Figure 43: Left to right: blue channels of previous denoised images. Note that much more details are kept in the right one.

7.8 Cross-channel Steering Kernel Regression Method

Recall the original SKR method, the estimation of the $u(x)$ is computed from a weighted least-squares optimization problem:

$$\min_{\{\beta_n\}} \sum_{i=1}^{P} \left[y_i - \beta_0 - \beta_1^T (x_i - x) - \cdots \right]^2 K_{H_i^{steer}}(x_i - x)$$

with

$$K_{H_i^{steer}}(x_i - x) = \frac{\sqrt{det(C_i)}}{2\pi h^2 \mu_i^2} exp\left\{ -\frac{(x_i - x)^T C_i (x_i - x)}{2h^2 \mu_i^2} \right\}.$$

where C_i is computed from equation (5.1.10).

The solution of β_0 for the least-squares estimation is

$$\hat{\beta}_0 = e_1^T (X^T W X)^{-1} X^T W y$$

where $e_1 = (1, 0, \cdots, 0)^T$ is a column vector in R^P,

$$y = [y_1, y_2, \cdots, y_P]^T, \quad b = [\beta_0, \beta_1^T, \cdots, \beta_N^T]^T,$$

$$W = \text{diag}[K_{H_1^{steer}}(x_1 - x), K_{H_2^{steer}}(x_2 - x), \cdots, K_{H_P^{steer}}(x_P - x)],$$

and

$$X = \begin{bmatrix} 1 & (x_1 - x)^T & \text{vech}^T\{(x_1 - x)(x_1 - x)^T\} & \cdots \\ 1 & (x_2 - x)^T & \text{vech}^T\{(x_2 - x)(x_2 - x)^T\} & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ 1 & (x_P - x)^T & \text{vech}^T\{(x_P - x)(x_P - x)^T\} & \cdots \end{bmatrix}. \tag{7.8.10}$$

Now consider a color image in $mYCrCb$ space. Since the size and shape of footprints determines *where* to blur, and parameters P (local window size) and h determines *how much* to blur. The natural thinking would be using Y channel to compute steering matrices (which determines the footprint), and using larger values of P and h in Cr and Cb channels to obtain a stronger blurring effect.

In details,

$$\begin{aligned} CNL(\vec{u}(x)) &= CNL(u_1(x)\vec{i} + u_2(x)\vec{j} + u_3(x)\vec{k}) \\ &= CNL_1(u_Y(x))\vec{i} + CNL_2(u_{Cr}(x))\vec{j} + CNL_3(u_{Cb}(x))\vec{k} \end{aligned} \tag{7.8.11}$$

where

$$CNL_1(u_Y)(x) = e_1^T(X_Y^T W_Y X_Y)^{-1} X_Y^T W_Y y_Y,$$

$$CNL_2(u_{Cr})(x) = e_1^T(X_{Cr}^T W_{Cr} X_{Cr})^{-1} X_{Cr}^T W_{Cr} y_{Cr},$$

$$CNL_3(u_{Cb})(x) = e_1^T(X_{Cb}^T W_{Cb} X_{Cb})^{-1} X_{Cb}^T W_{Cb} y_{Cb},$$

where y_Y, y_{Cr}, y_{Cb} and X_Y, X_{Cr}, X_{Cb} are defined similarly as in (5.1.4) and (5.1.5), but in different channels and with differnt Ps. For W_Y, W_{Cr} and W_{Cb}:

$$W_Y = \text{diag}[K_{H^{steer}_{1,Y}}(x_1 - x), K_{H^{steer}_{2,Y}}(x_2 - x), \cdots, K_{H^{steer}_{P_1,Y}}(x_{P_1} - x)],$$

$$W_{Cr} = \text{diag}[K_{H^{steer}_{1,Cr}}(x_1 - x), K_{H^{steer}_{2,Cr}}(x_2 - x), \cdots, K_{H^{steer}_{P_2,Cr}}(x_{P_2} - x)],$$

$$W_{Cb} = \text{diag}[K_{H^{steer}_{1,Cb}}(x_1 - x), K_{H^{steer}_{2,Cb}}(x_2 - x), \cdots, K_{H^{steer}_{P_3,Cb}}(x_{P_3} - x)],$$

where we choose P_2, $P_3 > P_1$ and

$$K_{H^{steer}_{i,Y}}(x_i - x) = \frac{\sqrt{det(C_i^Y)}}{2\pi h_1^2 \mu_i^2} exp\left\{-\frac{(x_i - x)^T C_i^Y (x_i - x)}{2h_1^2 \mu_i^2}\right\},$$

$$K_{H^{steer}_{i,Cr}}(x_i - x) = \frac{\sqrt{det(C_i^Y)}}{2\pi h_2^2 \mu_i^2} exp\left\{-\frac{(x_i - x)^T C_i^Y (x_i - x)}{2h_2^2 \mu_i^2}\right\},$$

$$K_{H^{steer}_{i,Cb}}(x_i - x) = \frac{\sqrt{det(C_i^Y)}}{2\pi h_3^2 \mu_i^2} exp\left\{-\frac{(x_i - x)^T C_i^Y (x_i - x)}{2h_3^2 \mu_i^2}\right\},$$

here h_2, $h_3 > h_1$. So 3 channels share same steering matrices (at each pixel) computed from Y channel, and when denoising Cr and Cb channels, the larger Ps and hs will obtain stronger noise blurring effect. See Figure 44-47.

Figure 44: Left: chicken image denoised by SKR method, 1iteration; Right: cross-channel method performed on cross-channel SKR method, 1iteration.

7.9 Comparisons and Experiments

Here are some examples showing the cross-channel effect. Figure 47-50 were introduced in [44]. We use the reconstructed blue channel to indicate the effectiveness of denoising. We have to make the comment that, although after several iterations, filters without cross-channel may clean the noise quite well, the cross-channel reduces the number of iterations and preserve the sharpness of edges.

Figure 45: Blue channel of the previous images. The right one looks cleaner in many areas.

Figure 46: Left to right: 100 % crop of the previous images denoised by SKR schemes with and without cross-channel, respectively, 1 iteration. The noticeable noise in the left one is much more than in the right one.

Figure 47: The blue channels of previous crops. Obviously the right one is much better.

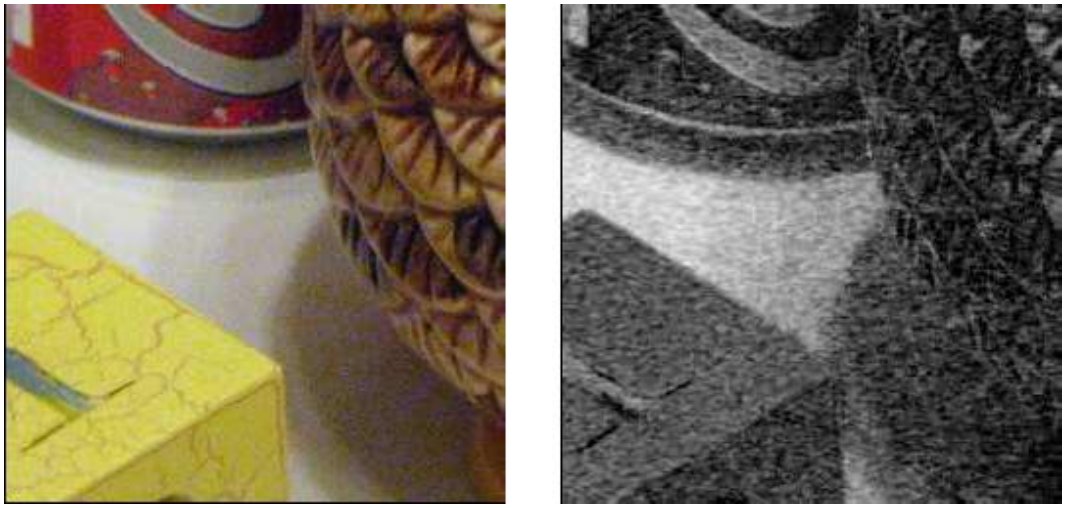


Figure 48: Another cut of the chicken image and its blue channels.

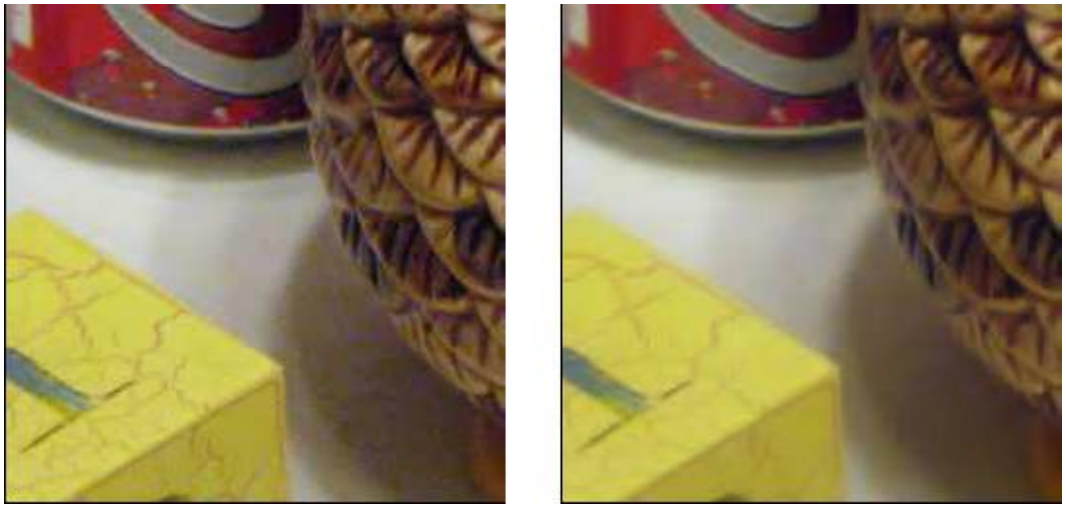


Figure 49: The denoised images by wavelet hard (left) and soft (right) thresholdings in the RGB space. Either noticable noise still exists due to high noise in blue channel, or the image is excessively smeared.

Figure 50: The denoised image by MTV in the RGB space (left) and its blue channel (right). Noticable noise still exists due to high noise in blue channel even the recomposed image has most of the noise removed.

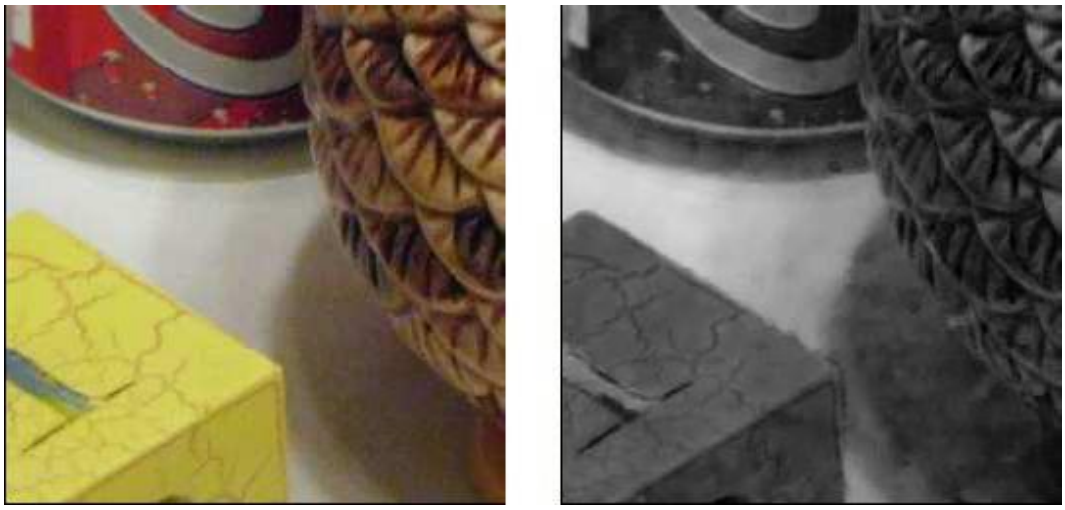


Figure 51: The denoised image by MTV in $mYCrCb$ color space (left) and its blue channel (right) which is much cleaner than the original one.

CHAPTER VIII

CONCLUSION

We have reviewed the previous mathematical models of denoising. For the grey value images, we analyzed method noise of some methods.

Particularly, for Kernel Regression model, an adaptive bilateral filter is introduced as complementary to enhance it. Significant improvement can be seen when we denoise an image with noise in discrete distribution, e.g. salt and pepper noise, and, the extreme case, a clean image with no noise.

Also a non-local bilateral filter is proposed based on the idea of non-local means filter. In this part, the idea of *related* pixels is taken out, e.g. all pixels closed both in spatial distance and intensity distance of the whole image should be taken into account to denoise. Here comes out the non-local bilateral fiter. Using Sapiro's blocks pre-classification, this algorithm works fast and efficiently. Also it solved the color leaking problem which exists in performance of the original NLM filter.

For digital color images with real noise, based on the analysis of color space decomposition and noise distribution, we proposed a cross-channel paradigm. A new $mYCrCb$ color space is built particularly for this method, which significantly makes the algorithm efficiently remove the noise in chrominance channels and keep the geometry and details well relying on the luminance channel. This method can be widely used together with almost any of the previous denoising methods but saves the number of iterations.

For all models, more details preservation and more noise removal are always a trade-off task. So the essential question in denoising is: What on earth is a noise? Do we have an exact mathematical definition of it? Unfortunately, the answer is no. The

reason can be described by the brilliant maxim circulated among signal processing experts:

"*What to one is a noise is often a signal to another.*"

REFERENCES

[1] L. Alvarez and P-L. Lions and J-M. Morel, *Image selective smoothing and edge detection by nonlinear diffusion (II)*, SIAM Journal of numerical analysis 29, pp. 845-866, 1992.

[2] A. Chambolle and P. L. Lions, *Image recovery via total variation minimization and related problems*, Numer. Math. 76, pp. 167-188, 1997.

[3] D. Donoho and I. Johnstone, *Ideal spatial adaptation via wavelet shrinkage*, Biometrika, 81 pp.425-455, 1994.

[4] D. Donoho, *Denoising by soft-thresholding*, IEEE Transactions on Information Theory, 41, pp.613-627, 1995.

[5] A. Efros and T. Leung, *Texture synthesis by non parametric sampling*, Proc. Int. Conf. Computer Vision (ICCV 99), Vol. 2, pp. 1033-1038, 1999.

[6] F. Guichard, J.M Morel and R. Ryan, *Image Analysis and P.D.E.'s*.

[7] M. Lindenbaum and M. Fischer and A. M. Bruchkstein, *On Gabor Contribution To Image Enhancement*, Pattern Recognization 27, pp. 1-8, 1994.

[8] F. Malgouyres, *A noise selection approach of image restoration*, Applications in signal and image processing IX, Vol 4478, pp. 34-41, 2001.

[9] S. Mallat, *A wavelet tour of signal processing*, Academic Press, 1997.

[10] Y. Meyer, *Oscillating Patterns in Image Processing and Nonlinear Evolution Equations*, University Lecture Series Vol 22, AMS 2002.

[11] S. Osher, M. Burger, D. Goldfarb, J. Xu and W. Yin, *Using Geometry and iterated refinement for inverse problems (1): Total variation based image restoration*, CAM-Report 04-13 UCLA, 2004.

[12] P. Perona and J. Malik, *Scale space and edge detection using anisotropic diffusion*, IEEE Trans. Patt. Anal. Mach. Intell., 12, pp. 629-639, 1990.

[13] L. Rudin and S. Osher, *Total Variation based image restoration with free local constraints*, Proc. IEEE ICIP, Vol 1, pp. 31-35, 1994.

[14] L. Rudin and S. Osher, *Nonlinear total variation based noise removal algorithms*, Physica D, 60, pp. 259-268, 1992.

[15] S.M. Smith and J.M. Brady, *Susan - a new pproach to low level image processing*, International Journal of Computer Vision, Vol 23(1), pp. 45-78, 1997.

[16] A. Buades, B. Coll, and J.-M. Morel, *Non-Local Algorithm for Image Denoising*, in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2, pp. 605,IEEE Computer Society, (Washington, DC, USA), October 2005.

[17] R. Acar and C. R. Vogel, *Analysis of total variation penalty methods for ill-posed problems*. Inverse Prob., 10: 1217-1229, 1994.

[18] T. F. Chan, S, Osher, and J. Shen, The Digital TV Filter and Nonlinear Denoising, IEEE Trans, Image Proc., 10(2), pp. 231-241, 2001.

[19] D. Dobson and C. R. Vogel, *Convergence of an Iterative Methods for Total Variation Denoising*, SIAM Journal on Numerical Analysis, 34(1997), pp. 1779-1971.

[20] M. Mahmoudi and G. Sapiro, *Fast Image and Video Denoising via Nonlocal Means of Similar Neighborhoods*, IEEE Signal Processing Letters 12, pp. 83942, December 2005.

[21] A. P. Witkin, *Scale-space filtering*, in Proceedings of IJCAI, Karlsruhe, 1983, pp.1019-1021.

[22] C. Tomasi and R. Manduchi, *Bilateral Filtering for Gray and Color Images*, in Proc. 6th Int. Conf. Computer Vision, New Delhi, India, 1998.

[23] M. Black, G. Sapiro, D. Marimont, and D. Heeger, *Robust Anisotropic Diffusion*, IEEE Trans. Image Process., vol. 7, no. 3, pp. 42132, Mar. 1998.

[24] P. J. Rousseeuw and A. M. Leroy, *Robust Regression and Outlier Detection* New York: Wiley, 1987.

[25] Michael Elad, *On the Origin of the Bilateral Filter and Ways to Improve It*

[26] D. Ruppert and M. P. Wand, *Multivariate locally weighted least squares regression*, Ann. Statist., Vol. 22, no. 3, pp. 1346-1370, Sept. 1994.

[27] X. Feng and P. Milanfar, *Multiscale principal components analysis for image local orientation estimation*, presented at the 36th Asilomar Conf. Signalss, Systems and Computers, Pacific Grove, CA, Nov.2002.

[28] D. Barash, *A Fundamental Relationship Between Bilateral Filtering, Adaptive Smoothing, and the Nonlinear Diffusion Equation*, IEEE Trans. Pattern Anal. Mach. Intell., vol. 24, no. 6, pp. 84447, Jun. 2002.

[29] A. Buades, B. Coll, and J. Morel, *Neighborhood Filters and PDE*, Numer. Math., vol. 105, pp. 1-4, 2006.

[30] . Caselles, B. Coll, and J. Morel, *Geometry and color in natural images*, Journal of Mathematical Imaging and Vision, 16(2):89-105, 2002.

[31] L.P. Yaroslavsky, *Digital Picture Processing - An Introduction*, Springer Verlag, 1985.

[32] L. Yaroslavsky, M. Eden *Fundamentals of Digital Optics*. Birkhauser, Boston, 1996.

[33] H. Takeda, S. Farsiu, P. Milanfar, *Kernel Regression for Image Processing and Reconstruction*, IEEE Trans. Image Process., vol. 16, no. 2. Feb. 2007.

[34] CIE (1932). *Commission internationale de l'Eclairage proceedings*, 1931. Cambridge University Press, Cambridge.

[35] Smith, Thomas; Guild, John (1931-32). *The C.I.E. colorimetric standards and their use*. Transactions of the Optical Society 33 (3): 73134.

[36] Hunt, R. W. (1998). *Measuring colour (3rd ed.)*. Fountain Press, England. ISBN 0-86343-387-1.

[37] Junichi Nakamura (2005). *Image Sensors and Signal Processing for Digital Still Cameras*.

[38] Linda G. Shapiro and George C. Stockman (2001). *Computer Vision*.

[39] Charles Boncelet (2005). *Image Noise Models*. in Alan C. Bovik. *Handbook of Image and Video Processing*.

[40] Y. Wang and H. M. Zhou, *A Total Variation Wavelet Algorithm for Medical Image Denoising* , the International Journal on Biomedical Imaging, Volume 2006, article ID 89095, 6 pages, 2006.

[41] I. Daubechies, *Ten lectures on wavelets*. SIAM, Philadelphia, 1992.

[42] G. Strang and T. Nguyen, *Wavelets and Filter Banks, Wellesley-Cambridge Press*, Wellesley, MA, 1996.

[43] E. Ordentlich, G. Seroussi, S. Verdu, M. Weinberger and T. Weissman, *A discrete universal denoiser and its application to binary images*. Proc. IEEE ICIP, Vol 1, pp. 117-120, 2003.

[44] T. Chan , Y. Wang and H. M. Zhou, *Denoising Natural Color Photos in Digital Photography* , submitted to IEEE Transcation on Image Processing.

[45] Lindsay MacDonald (2006). *Digital Heritage*. Butterworth-Heinemann.

[46] James R. Janesick (2001). *Scientific Charge-coupled Devices*. SPIE Press.

[47] Michael A. Covington (2007). *Digital SLR Astrophotography*. Cambridge University Press.

[48] R. E. Jacobson, S. F. Ray, G. G. Attridge, and N. R. Axford (2000). *The Manual of Photography*. Focal Press.

[49] Canny, J., *A Computational Approach To Edge Detection*, IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679-714, 1986.

VITA

Hao Deng was born in Changsha, P.R.China in September, 1980. He went to University of Science and Technology of China in 1997 and obtained his bachelor and master's degrees from Department of Mathematics, in 2001 and 2004, respectively. In July of 2004, he moved to Atlanta, GA and joined Georgia Institute of Technology to work in Image Denoising with Dr. Yang Wang and Dr. Haomin Zhou.