UNIVERSITY OF CALIFORNIA, SAN DIEGO

Numerical Optimization for Image and Video Restoration

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Electrical Engineering (Signal and Image Processing)

by

Ho Chan

Committee in charge:

Professor Truong Q. Nguyen, Chair
Professor James R. Bunch
Professor Philip E. Gill
Professor Gert Lanckriet
Professor Nuno Vasconcelos

2011

The dissertation of Ho Chan is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____

_____

_____

Chair

University of California, San Diego

2011

DEDICATION

To my parents.

EPIGRAPH

"And ye shall know the truth, and the truth shall make you free."

John 8:32

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

Studying abroad is always challenging for international students. Yet, when I look back to my Ph.D. life in San Diego, I never feel lonely. The Lord, as He promised, is always with me. His grace and providence is abundant.

I would like to give my most sincere thanks to my advisor Professor Truong Nguyen. Professor Nguyen is a fantastic advisor who gives me so much freedom to explore. He is full of creative ideas with high standard. As a teacher, he not only teaches me technical skills, but also shows me a role model of how to be a good and responsible researcher.

I want to give a special thanks to Professor Philip Gill for his continuous support on my research. Professor Gill is a great mathematician and has a strong insight in numerical optimization. It is he who suggested to me to consider augmented Lagrangian methods for image restoration, and later suggested to me to extend the ideas to video restoration.

I would like to express my gratitude to other committee members, Professors James Bunch, Gert Lanckriet, and Nuno Vasconcelos for their time and help in my research. I also want to give thanks to professors who helped me during my Ph.D. study: Professors Michael Saunders (Stanford), Thomas Wu (Univ. of Central Florida), Michael Ng (Hong Kong Baptist Univ.) and Edmund Lam (Univ. of Hong Kong). I must thank my fellow video processing lab members, who share with me happiness and sadness: Cheolhong An, Haleh Azartash, Can Bal, Kris Gibson, Shay Har-noy, Natan Jacobson, Ankit Jain, Jason Juang, Ramsin Khoshabeh, Kyoung-rok Lee, Yen-lin Lee, Lester Liu, Karl Ni, Lam Tran, Dung Vo, and Yujia Wang.

The publication of this dissertation is not possible without the generous support of the Croucher Foundation Hong Kong, who provided me a Croucher Foundation Scholarship for full-time overseas Ph.D. studies. I thank the ECE department and Prof Nguyen for supporting me financially during last two years' research.

Finally, I would like to dedicate this dissertation to my parents, Kwok-hoi and Fung, for their unconditioned love. I also thank my fiancée Vivian for her continuous support of my Ph.D. studies.

Part of this dissertation is a reprint of published/ submitted papers. Chapter 3, in part, is a reprint of a published paper in IEEE Transactions on Image Processing, Aug

2011. Chapter 4 and 5, in part, is a reprint of a published paper in IEEE Transactions on Image Processing, Dec 2011, and a reprint of a conference paper presented in IEEE International Conference on Acoustics, Speech and Signal Processing 2010. Chapter 7 and 8, in part, is a reprint of a submitted paper to IEEE Transactions on Image Processing, Jun 2011 and a reprint of a conference paper presented in IEEE International Conference on Image Processing 2011. Chapter 6, in part, in a reprint of two published conference papers in IEEE International Conference on Image Processing 2010 and OSA Topical Meetings on Signal Recovery and Synthesis 2011. The dissertation author was the primary research and author of the above mentioned papers. Co-authors Truong Q. Nguyen and Philip E. Gill listed in these publications supervised the research which became the basis for this dissertation. Other co-authors contributed to the papers in the form of writing and simulation.

The major portion of the research in this dissertation is conducted during the tenure of Croucher Foundation Scholarship. The research of LCD related problems in Chapter 3 is partially supported by Samsung Information Systems America, and AMD/Broadcomm Inc.

VITA

2007	B. Eng. in Electrical Engineering (*First Class Honour*), University of Hong Kong
2009	M. A. in Mathematics (Applied), University of California, San Diego
2011	Ph. D. in Electrical Engineering (Signal and Image Processing), University of California, San Diego

PUBLICATIONS

Stanley H. Chan, "Single Image Spatial Variant Out-of-focus Blur Removal," submitted to *IEEE Transactions on Image Processing*, Oct 2011.

Stanley H. Chan, Ramsin Khoshabeh, Kristofor B. Gibson, Philip E. Gill and Truong Q. Nguyen, "An augmented Lagrangian method for total variation video restoration," *IEEE Transactions on Image Processing*, vol. 20, issue 11, pp.3097-3111, Nov 2011.

Stanley H. Chan and Truong Q. Nguyen, "LCD motion blur: modeling, analysis and algorithm," *IEEE Transactions on Image Processing*, vol. 20, issue 8, pp.2352-2365, Aug 2011.

Stanley H. Chan, Thomas X. Wu and Truong Q. Nguyen, "Comparison of two frame rate conversion schemes for reducing LCD motion blurs," *IEEE Signal Processing Letters* vol. 17, issue 9, pp.783-786, 2010.

Stanley H. Chan and Truong Q. Nguyen, "Single image spatial variant out-of-focus blur removal," to appear in *Proceedings of IEEE International Conference on Image Processing (ICIP '11)*, 2011.

Stanley H. Chan, Ankit K. Jain, Truong Q. Nguyen and Edmund Y. Lam, "Bounds for the condition numbers of spatially-variant convolution matrices in image restoration problems," *OSA Topical Meeting in Signal Recovery and Synthesis*, paper SMA4, Toronto, July 2011.

Ramsin Khoshabeh, Stanley H. Chan and Truong Q. Nguyen, "Spatio-temporal consistency in video disparity estimation," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '11)*, pp.885-888, Prague, May 2011.

Stanley H. Chan, Ramsin Khoshabeh, Kristofor B. Gibson, Philip E. Gill and Truong Q. Nguyen, "An augmented Lagrangian method for video restoration," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '11)*, pp.941-944, Prague, May 2011.

Stanley H. Chan, "Constructing a sparse convolution matrix for shift varying image restoration problems," in *Proceedings of IEEE International Conference on Image Processing (ICIP '10)*, pp.3601-3604, Hong Kong, Sept 2010.

Stanley H. Chan and Truong Q. Nguyen, "LCD motion blur modeling and simulation," in *Proceedings of IEEE International Conference on Multimedia and Exposition (ICME '10)*, pp.400-405, Singapore, July 2010.

Stanley H. Chan, Dung Vo and Truong Q. Nguyen, "Sub-pixel motion estimation without interpolation," in *Proceedings of IEEE Conference on Acoustics, Speech and Signal Processing (ICASSP '10)*, pp.722-725, Dallas, March 2010.

Shay Har-Noy, Stanley H. Chan and Truong Q. Nguyen, "Demosaicing images with motion blur," in *Proceedings of IEEE Conference on Acoustics, Speech and Signal Processing (ICASSP '10)*, pp.1006-1009, Dallas, March 2010.

Stanley H. Chan and Truong Q. Nguyen, "Fast LCD motion deblurring by decimation and optimization," in *Proceedings of IEEE Conference on Acoustics, Speech and Signal Processing (ICASSP '09)*, pp.1201-1204, Taipei, April 2009 Taipei.

ABSTRACT OF THE DISSERTATION

Numerical Optimization for Image and Video Restoration

by

Ho Chan

Doctor of Philosophy in Electrical Engineering (Signal and Image Processing)

University of California, San Diego, 2011

Professor Truong Q. Nguyen, Chair

Image restoration is an inverse problem where the goal is to recover an image from a blurry and noisy observation. An image restoration problem can be formulated as a total variation regularized least-squares minimization where the objective function is the l_2-norm squares of the residue between the observation and the prediction. Since the total variation norm is not differentiable, existing methods are inefficient.

In this dissertation, a fast numerical optimization method is proposed to solve total variation image restoration problems. The method transforms the original unconstrained problem to an equivalent constrained problem and uses an augmented Lagrangian method to handle the constraints. The transformation allows the differentiable and non-differentiable parts of the objective function to be separated into different subproblems where each subproblem may be solved efficiently. An alternating strategy is

then used to combine the subproblem solutions.

The image restoration method is extended to handle video restoration problems. The proposed method considers a video as a space-time volume, and introduces a three-dimensional total variation regularization function to enhance the spatial and temporal consistency. The new video restoration framework opens a wide range of applications, including video deblurring and denoising, disparity map refinement, and hot-air turbulence removal.

Practical image and video restoration methods need to take into account spatially variant blur and blind deconvolution issues. Therefore, spectral properties of the spatially variant convolution matrices are studied. A fast and robust blind deconvolution method for single image spatially variant out-of-focus blur removal is proposed.

Chapter 1

Introduction

1.1 Motivation

Image restoration is an inverse problem where the goal is to estimate a sharp image from noisy and blurry observations. Often, the degradation process is not reversible, thus making image restoration problems ill-posed. The goal of this dissertation is to propose numerically stable and fast algorithms to perform image restoration, and extend the idea to video problems.

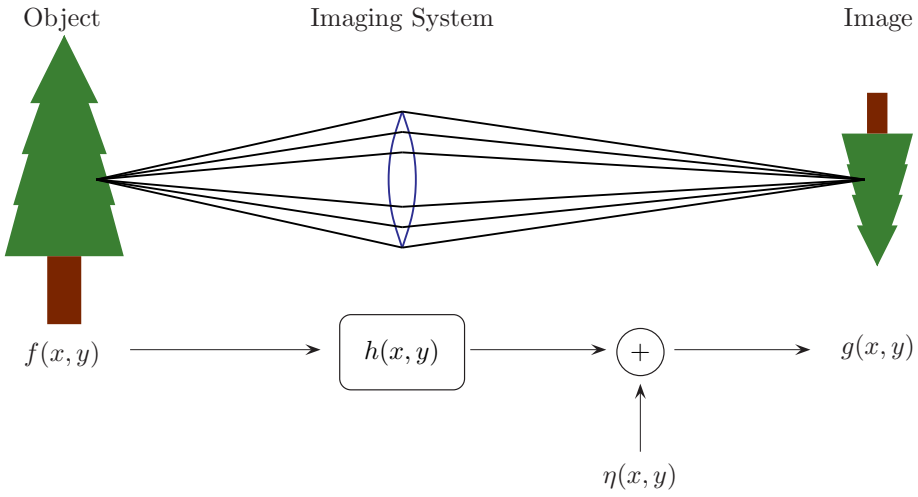


Figure 1.1: Block diagram of an imaging system

Shown in Fig. 1.1 is the classical linear shift invariant imaging system [49]. The light intensity at coordinate (x, y) [1] in space is denoted by a two-dimensional function $f(x, y)$. The imaging process is characterized by its impulse response (also known as the point spread function, PSF) $h(x, y)$. The observed image $g(x, y)$ is

$$g(x, y) = h(x, y) * f(x, y) + \eta(x, y),$$

where $\eta(x, y)$ is the noise term and $*$ denotes the convolution operation:

$$h(x, y) * f(x, y) = \sum_{u,v} h(x - u, y - v) f(u, v).$$

Recovering $f(x, y)$ from $g(x, y)$ can be formulated as the minimization problem

$$\underset{f(x,y)}{\text{minimize}} \sum_{x,y} |h(x, y) * f(x, y) - g(x, y)|^2 . \tag{1.1}$$

However, the global minimizer of (1.1) may not be unique because the convolution operator can be singular. In many cases, even if the convolution operator is non-singular, finding the unique global minimizer is still challenging as the convolution operator may be numerically rank deficient.

To obtain a good recovery result, prior knowledge of the image must be incorporated in solving (1.1). This leads to the question of choosing regularization functions for (1.1), or equivalently the prior of $f(x, y)$. However, the difficulty is that computationally efficient priors are not necessarily effective (e.g., Tikhonov regularization) whereas effective priors may not be efficient (e.g., total variation regularization). Therefore, seeking a good prior and developing an algorithm becomes the cornerstone of image restoration, which is also the theme of this dissertation.

The concept of image restoration can be extended to video restoration. Considering a video as a sequence of consecutive images, the observed video sequence is

$$g(x, y, t) = h(x, y, t) * f(x, y, t) + \eta(x, y, t),$$

where the third coordinate t denotes time. Recovering $f(x, y, t)$ from $g(x, y, t)$ is essen-

[1] In this dissertation, all coordinates are integer valued. Therefore, $f(x, y)$ is a two-dimensional discrete-time signal.

tially the same as solving (1.1) in the three-dimensional space. Due to the additional time coordinate, the prior of $f(x, y, t)$ should consider the temporal smoothness. We will discuss, later in this dissertation, that the temporal prior opens a wide range of new applications in image and video processing. Specifically, motion blur problems, video flickering issues, and distortion caused by hot-air turbulence can be solved using the video restoration framework.

Finally, two restrictive assumptions made in solving (1.1) should be addressed. First, $h(x, y)$ is assumed to be known exactly, which never happens in practice. Second, the imaging system is assumed to be spatially invariant (i.e., all pixels are blurred equally), which is also not valid because most of the blurs are position dependent. Therefore, for practical considerations, these two issues must also be solved.

1.2 Contribution

The dissertation achieves the following goals.

1. We propose an augmented Lagrangian method for solving total variation minimization problems. The proposed method supersedes the existing augmented Lagrangian method by introducing an automatic parameter update scheme to improve convergence. Convergence properties, warm start behavior and parameter sensitivity are discussed.

2. We extend the augmented Lagrangian method for video restoration problems by introducing the space-time total variation regularization function. Applications of the new algorithm include video deblurring, video denoising, disparity refinement, and hot-air turbulence removal.

3. We analyze the characteristics of spatially variant blurs. In particular, we propose a systematic way of constructing a spatially variant convolution matrix. We also provide an accurate upper and lower bounds on the eigenvalues of the convolution matrix.

4. We seek invariant structures in spatially variant blurs. For spatially variant motion blur problems, we transform the variant blur in space into an invariant blur in time by embedding the blurred image into a space-time volume. For spatially variant

out-of-focus blur problems, we transform the variant blur in space into two invariant blurs in depth by introducing artificial backgrounds for different objects.

1.3 Organization

The organization of this dissertation is as follows.

Chapter 2 provides background materials of this dissertation. We start by introducing point spread functions and noise models. Since a bulk of this dissertation is about numerical optimization, linkage between image processing and linear algebra is discussed. In particular, the structures and properties of convolution matrices, diagonalization of block-circulant-with-circulant-block matrices (BCCB matrices) are mentioned.

Chapter 3 reviews existing methods for image restoration. The methods we consider include Tikhonov regularization, isotropic and anisotropic total variation (TV) norms, and bilateral total variation (BTV) norms. For anisotropic TV, primal dual interior point method for convex objectives (`PDCO`) is tested for feasibility study. For bilateral TV problems, a projected sub-gradient method is tested.

Chapter 4 is one of the core chapters of this dissertation, which presents the proposed image restoration algorithm in detail. Our proposed method differs from existing augmented Lagrangian methods by introducing an automatic parameter update. Comparisons with other methods are discussed. Convergence proof is provided.

Chapter 5 is an extension of image restoration to video problems. In this chapter, a video is considered as a space-time volume. A three-dimensional regularization function is used to enforce the spatial and temporal smoothness of the solution. Applications including video deblurring and denoising, disparity map refinement and hot-air turbulence removal are discussed.

Chapter 6 addresses the issue of blind deconvolution, which is used in the subsequent discussions in Chapter 7 and 8.

Chapter 7 concerns about the theoretical aspects of spatially variant blur problems. In particular, we propose a systematic method of constructing the spatially variant convolution matrix. Having the convolution matrix constructed, we discuss the spectral properties of the convolution matrix. We show the upper and lower bounds on the eigenvalues of the convolution matrix.

Chapter 8 discusses the problem of removing spatially variant out-of-focus blur

using a single image. It is an application of the materials developed in Chapter 4-7. In this chapter, we show that it is possible to perform blind deconvolution on a two-layer out-of-focus blurred image.

Chapter 2

Background

This objective of this chapter is to provide necessary prerequisites on image processing and numerical linear algebra.

2.1 Point Spread Functions

We begin our discussion on the point spread function (PSF). PSF is the impulse response of an optical system subjected to a point source input. There are three commonly used PSFs in practice, namely the Gaussian PSF, disk PSF, and motion PSF.

- Gaussian PSF is defined as

$$h(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}},$$

  where σ^2 is the variance of the Gaussian PSF and it controls the spread of the PSF. Gaussian PSF is used in astronomical imaging systems to model atmospheric blur in which the light coming from outer space (e.g., stars) are scattered when traveling through the atmosphere.

- Disk PSF is defined as

$$h(x,y) = \begin{cases} 1, & \text{if } \sqrt{x^2+y^2} \leq r, \\ 0, & \text{otherwise }, \end{cases}$$

where r is the radius. Disk PSF is used to model out-of-focus blur, a phenomenon occurs when the image plane does not coincide with focal point of the lens. On a single lens imaging system, out-of-focus blur is unavoidable sometimes because when one object in a scene is focused, other objects at different depths are out of focus.

- Motion PSF is defined as

$$h(x,y) = \begin{cases} 1, & \text{if } v_x y = v_y x, \text{ and } \sqrt{x^2+y^2} = \sqrt{v_x^2+v_y^2} \\ 0, & \text{otherwise ,} \end{cases}$$

where the vector $\mathbf{v} = (v_x, v_y)$ is the motion vector. The direction of $\mathbf{v}$ determines the orientation of $h(x,y)$, and the magnitude of $\mathbf{v}$ determines the length of $h(x,y)$. Motion blur is a common problem when the object is moving or when the camera is not stationary.

2.2 Noise

We consider a noisy pixel as a random variable on the image grid. In this dissertation, two types of noise will be studied - Gaussian noise and impulsive noise. For simplicity, noise is assumed to be identically and independently distributed (i.i.d).

Gaussian noise is characterized by the normal distribution, with the probability density function

$$p_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

where μ is the mean, and σ^2 is the variance. To denote a pixel $\eta(x,y)$ as Gaussian noise with mean μ and variance σ^2, we write $\eta(x,y) \sim \mathcal{N}(\mu, \sigma^2)$. Fig. 2.1(b) illustrates an image corrupted by Gaussian noise with $\mu = 0$ and $\sigma^2 = 0.01$.

Impulsive noise is characterized by the probability of an observed image pixel hitting the maximum (or minimum) of the allowed pixel intensity range, i.e., $0 \leq g(x,y) \leq 255$ for 8-bit gray-scaled images, or $0 \leq g(x,y) \leq 1$ for normalize-scaled images. A precise definition of impulse noise is as follows: Let $\tilde{g}(x,y) = g(x,y) + \eta(x,y)$ be a random variable representing the sum of the image intensity $g(x,y)$ and the noise $\eta(x,y)$, the

distribution

$$\begin{cases} \text{Prob}\{\tilde{g}(x,y) = g(x,y)\} = 1 - \alpha \\ \text{Prob}\{\tilde{g}(x,y) = 1\} = \frac{\alpha}{2} \\ \text{Prob}\{\tilde{g}(x,y) = 0\} = \frac{\alpha}{2}, \end{cases}$$

defines the probability that a pixel is corrupted, where $0 \leq \alpha \leq 1$ controls the amount of impulsive noise. Fig. 2.1(c) illustrates an image corrupted by impulsive noise with $\alpha = 0.05$.

(a) Image (b) Gaussian Noise ($\sigma^2 = 0.01$) (c) Impulsive Noise ($\alpha = 0.05$)

Figure 2.1: An illustration of two types of noise: Gaussian noise and impulsive noise.

2.3 Convolution Matrix

The convolution between an image $f(x,y)$ and a PSF $h(x,y)$ is

$$h(x,y) * f(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} h(x-u, y-v) f(u,v), \tag{2.1}$$

where $*$ denotes the convolution operation, M and N are number of rows and columns of $f(x,y)$, respectively. Since $f(x,y)$ are $h(x,y)$ are arrays, we write $\mathbf{f}$ and $\mathbf{h}$ for simplicity. So (2.1) may be written as $\mathbf{h} * \mathbf{f}$.

For the discussion of numerical linear algebra, it is often easier to express linear operations as matrix-vector multiplications. Since convolution is linear, we define a matrix known as the *convolution matrix* for representation. To do so, we first stack the

columns of $f(x,y)$ into a vector $\mathbf{f}$ according to the lexicographic order. That is,

$$\mathbf{f} = \mathbf{vec}(f(x,y)),$$

where **vec** is the vectorization operator that transforms a matrix $\mathbf{A} = (a_{i,j})$ to a vector as $\mathbf{vec}(\mathbf{A}) = [a_{1,1}, a_{2,1}, a_{3,1}, \ldots, a_{m,n}]^T$. Vectorization of an $M \times N$ image $f(x,y)$ yields an $MN \times 1$ vector $\mathbf{f}$. The array $\mathbf{f}$ and the vector $\mathbf{f}$ will be used interchangeably, and should be clear from the context: For $\mathbf{h} * \mathbf{f}$, $\mathbf{f}$ means an array, and for $\mathbf{Hf}$, $\mathbf{f}$ means a vector.

The convolution matrix $\mathbf{H}$ is a linear operator that maps a vectorized image $\mathbf{f}$ to another vectorized image $\mathbf{g}$ following the rule

$$\mathbf{g} = \mathbf{Hf} = \mathbf{vec}\left(h(x,y) * f(x,y)\right).$$

There are several important properties of $\mathbf{H}$. First, the dimension of $\mathbf{H}$ is $MN \times MN$, which can be large if the image size is large. However, the number of non-zero entries of $\mathbf{H}$ is only $|\Omega_h| \times |\Omega_f|$, where Ω_h is the support of $h(x,y)$, Ω_f is the support of $f(x,y)$ and $|\Omega|$ is the cardinality of the set Ω. Therefore, despite the large number of columns and rows of $\mathbf{H}$, $\mathbf{H}$ is *sparse*. Consequently, the computational cost of matrix-vector multiplications, $\mathbf{Hf}$ and $\mathbf{H}^T\mathbf{f}$, are inexpensive. Even if $|\Omega_h| \times |\Omega_f|$ is not small, the matrix-vector multiplications $\mathbf{Hf}$ and $\mathbf{H}^T\mathbf{f}$ can still be efficiently performed if $\mathbf{H}$ is a block-circulant-with-circulant-block (BCCB) matrix [61]. BCCB matrices can be diagonalized using discrete Fourier Transform (DFT) matrices, which will be discussed next.

2.4 Matrix-vector Multiplications via Fourier Transforms

Definition 1. *Given a two-dimensional discrete-time signal $f(x,y)$, the two-dimensional discrete Fourier Transform (2D-DFT) is [16]*

$$[\mathcal{F}f](u,v) \stackrel{def}{=} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j\frac{2\pi}{M}ux} e^{-j\frac{2\pi}{N}vy},$$

where (u,v) is the coordinate in the Fourier domain, and $j = \sqrt{-1}$.

We also define the discrete Fourier Transform matrix:

Definition 2. *The two-dimensional discrete Fourier Transform matrix is defined as*

$$\mathbf{F} = F_M \otimes F_N,$$

where $\otimes$ *is the Kronecker product operator, and* F_N *is the* N*-point one-dimensional DFT matrix:*

$$F_N = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N & W_N^2 & \cdots & W_N^N \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^N & W_N^{2N} & \cdots & W_N^{N^2} \end{bmatrix},$$

where $W_N = e^{-j2\pi/N}$*, and* $j = \sqrt{-1}$*.*

With the definition of 2D-DFT matrix, we write the 2D-DFT of an image as

$$\mathbf{Ff} = \mathbf{vec}\left(\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j\frac{2\pi}{M}ux} e^{-j\frac{2\pi}{N}vy} \right).$$

2D-DFT matrices are unitary, i.e.,

$$\mathbf{F}^H\mathbf{F} = \mathbf{F}\mathbf{F}^H = \mathbf{I},$$

where $(\cdot)^H$ is the Hermitian operator. Also, 2D-DFT matrices are symmetric, meaning that

$$\mathbf{F}^T = \mathbf{F}, \quad \mathbf{F}^* = (\mathbf{F}^*)^T = \mathbf{F}^H,$$

where $(\cdot)^*$ denotes the complex conjugate.

The following theorem links convolution and the Fourier Transform.

Theorem 1. *If* $\mathbf{H}$ *is a BCCB matrix, then it can be diagonalized by the 2D-DFT matrix*

$$\mathbf{H} = \mathbf{F}^H\mathbf{\Lambda}\mathbf{F},$$

where $(\cdot)^H$ *is the Hermitian operator, and* $\mathbf{\Lambda}$ *is a diagonal matrix with entries being the eigenvalues of* $\mathbf{H}$*.*

Theorem 1 implies that matrix-vector multiplication $\mathbf{Hf}$ can be performed efficiently as $\mathbf{Hf} = \mathbf{F}^H\mathbf{\Lambda}\mathbf{Ff}$, which corresponds to the following 4 steps:

1. Pre-compute the eigenvalue matrix $\mathbf{\Lambda}$ by applying a Fourier Transform to $h(x, y)$.
2. Apply a Fourier Transform to the image $f(x, y)$ to yield $\mathbf{Ff}$.
3. Perform element-wise multiplication to yield $\mathbf{\Lambda Ff}$, because $\mathbf{\Lambda}$ is a diagonal matrix.
4. Apply an Inverse Fourier Transform to $\mathbf{\Lambda Ff}$ and get $\mathbf{F}^H\mathbf{\Lambda Ff}$.

The computational cost of $\mathbf{Ff}$ is in the order of $n \log n$, where n is the number of variables of $\mathbf{f}$. $\mathbf{Ff}$ is typically implemented by Fast Fourier Transform (FFT).

2.5 Forward Difference Operators

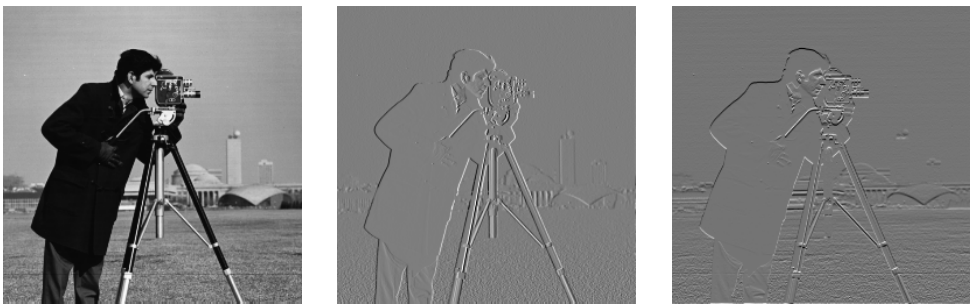
Finally, we discuss the forward difference operators. The horizontal and vertical forward difference operators $\mathbf{D}_x$ and $\mathbf{D}_y$ are defined by their operations on the image $\mathbf{f}$:

$$\mathbf{D}_x\mathbf{f} = \begin{cases} \mathbf{vec}\left(f(x+1, y) - f(x, y)\right) & , \quad 0 \leq x < N-1 \\ \mathbf{vec}\left(f(0, y) - f(N-1, y)\right) & , \quad x = N-1 \end{cases}$$

and

$$\mathbf{D}_y\mathbf{f} = \begin{cases} \mathbf{vec}\left(f(x, y+1) - f(x, y)\right) & , \quad 0 \leq y < M-1 \\ \mathbf{vec}\left(f(x, 0) - f(x, M-1)\right) & , \quad y = M-1. \end{cases}$$

Fig. 2.2 illustrates the effects of these two operators.



(a) Image $\mathbf{f}$ (b) $\mathbf{D}_x\mathbf{f}$ (c) $\mathbf{D}_y\mathbf{f}$

Figure 2.2: An illustration of $\mathbf{D}_x$ and $\mathbf{D}_y$. $\mathbf{D}_x\mathbf{f}$ is the horizontal gradient of the image $\mathbf{f}$ and $\mathbf{D}_y\mathbf{f}$ is the vertical gradient of the image $\mathbf{f}$.

The operators $\mathbf{D}_x\mathbf{f}$ and $\mathbf{D}_y\mathbf{f}$ can be performed through convolutions

$$\mathbf{D}_x\mathbf{f} = \mathbf{vec}(f(x,y) * d_x(x,y)),$$
$$\mathbf{D}_y\mathbf{f} = \mathbf{vec}(f(x,y) * d_y(x,y)),$$

where $d_x(x,y) = [1,\ -1]$ and $d_y(x,y) = [1,\ -1]^T$.

We also define the gradient operator ∇ on an image $\mathbf{f}$ as

$$\nabla\mathbf{f} \stackrel{def}{=} \begin{bmatrix} \partial_x\mathbf{f} \\ \partial_y\mathbf{f} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_x \\ \mathbf{f}_y \end{bmatrix},$$

where $\mathbf{f}_x = \partial_x\mathbf{f} = \mathbf{D}_x\mathbf{f}$ and $\mathbf{f}_y = \partial_y\mathbf{f} = \mathbf{D}_x\mathbf{f}$.

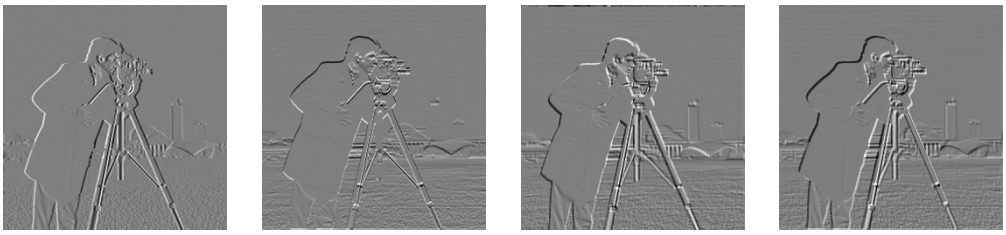
The transpose of $\mathbf{D}_x$ and $\mathbf{D}_y$ are defined as

$$\mathbf{D}_x^T\mathbf{f} = \begin{cases} \mathbf{vec}\left(f(x-1,y) - f(x,y)\right) & , \quad 0 < x \leq N-1 \\ \mathbf{vec}\left(f(N-1,y) - f(0,y)\right) & , \quad x = 0 \end{cases}$$

and

$$\mathbf{D}_y^T\mathbf{f} = \begin{cases} \mathbf{vec}\left(f(x,y-1) - f(x,y)\right) & , \quad 0 < y \leq M-1 \\ \mathbf{vec}\left(f(x,0) - f(x,M-1)\right) & , \quad y = 0. \end{cases}$$

In terms of convolution, $\mathbf{D}_x^T\mathbf{f} = \mathbf{vec}(f(x,y) * [-1,\ 1])$, and $\mathbf{D}_y^T\mathbf{f} = \mathbf{vec}(f(x,y) * [-1,\ 1]^T)$.



$\mathbf{D}_{-2,0}$ $\mathbf{D}_{2,0}$ $\mathbf{D}_{2,-2}$ $\mathbf{D}_{-2,2}$

Figure 2.3: An illustration of finite difference operators $\mathbf{D}_{i,j}$.

As an extension to the two standard forward difference operators, we define the generalized forward difference operator $\mathbf{D}_{i,j}$, with $-P \leq i \leq P$ and $-Q \leq j \leq Q$ for

some constants P and Q, as

$$\mathbf{D}_{i,j}\mathbf{f} = \begin{cases} \mathbf{vec}\left(f(x+i, y+j) - f(x,y)\right) & , \quad 0 \le x < N-i,\ 0 \le y < M-j \\ \mathbf{vec}\left(f(i,j) - f(N-i, M-j)\right) & , \quad N-i \le x \le N-1,\ M-j \le y \le M-1. \end{cases}$$

The interpretation of $\mathbf{D}_{i,j}$ is as follows. It is the forward difference operator with difference interval i in the horizontal direction and j in the vertical direction. For example, if $i = 1$ and $j = 0$, then the operator $\mathbf{D}_{1,0}$ is the standard forward difference operator along the horizontal direction. Fig. 2.3 illustrates some $\mathbf{D}_{i,j}$.

Chapter 3

Existing Image Restoration Methods

The objective of this chapter is to provide an overview of the existing image restoration methods. Since the literature on image restoration is abundant, it is not possible to discuss every single piece of work here. In order to align with the theme of this dissertation, we focus on the category of numerical optimization based methods.

3.1 Least-squares Minimization

Using the matrix-vector notion of the convolution, we write Problem (1.1) in the following way:

$$\underset{\mathbf{f}}{\text{minimize}} \quad \|\mathbf{H}\mathbf{f} - \mathbf{g}\|^2 . \tag{3.1}$$

Problem (3.1) is known as the unconstrained least-squares minimization problem. It can be solved by considering the solution of the normal equation

$$\mathbf{H}^T\mathbf{H}\mathbf{f} = \mathbf{H}^T\mathbf{g}, \tag{3.2}$$

which gives the pseudo inverse solution

$$\mathbf{f} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{g}. \tag{3.3}$$

Due to the large dimension of $\mathbf{H}$, the normal equation (3.2) has to be calculated via large-scale linear system solvers, such as `LSQR` [86]. However, $\mathbf{H}$ can be singular or numerically rank deficient. Thus, $(\mathbf{H}^T\mathbf{H})^{-1}$ in (3.3) causes serious perturbation when there is noise.

Another way to understand the limitation of (3.1) is by means of Fourier Transforms. Letting

$$\begin{aligned} F(u,v) &= \mathcal{F}[f(x,y)] \qquad \text{and} \qquad G(u,v) = \mathcal{F}[g(x,y)], \\ H(u,v) &= \mathcal{F}[h(x,y)] \qquad \text{and} \qquad N(u,v) = \mathcal{F}[\eta(x,y)], \end{aligned}$$

the relation $g(x,y) = h(x,y) * f(x,y) + \eta(x,y)$ can be written as

$$G(u,v) = H(u,v)F(u,v) + N(u,v).$$

Applying $H^*(u,v)$ to both sides yields

$$H^*(u,v)G(u,v) = |H(u,v)|^2 F(u,v) + H^*(u,v)N(u,v),$$

and hence

$$F(u,v) = \frac{H^*(u,v)G(u,v)}{|H(u,v)|^2} - \frac{H^*(u,v)N(u,v)}{|H(u,v)|^2},$$

assuming $H(u,v) \neq 0$ for all u and v.

If $N(u,v) = 0$, i.e., the noiseless case, we have

$$F(u,v) = \frac{H^*(u,v)G(u,v)}{|H(u,v)|^2},$$

which is equivalent to (3.3) in the Fourier domain. This method is known as the Inverse Filter, or a special case of Wiener deconvolution.

If $N(u,v) \neq 0$, then the term $H^*(u,v)N(u,v)$ does not vanish. Therefore, if $|H(u,v)|$ contains small values, then applying $|H(u,v)|^{-2}$ to $H^*(u,v)N(u,v)$ amplifies the noise as

$$\frac{H^*(u,v)N(u,v)}{|H(u,v)|^2}.$$

In the extreme case where $\min |H(u,v)| \to 0$, $\frac{H^*(u,v)N(u,v)}{|H(u,v)|^2} \to \infty$.

To summarize, the least-squares minimization (3.1) is not an appropriate method.

3.2 Tikhonov Regularized Least-Squares

To improve the condition number of $\mathbf{H}^T\mathbf{H}$, Tikhonov regularization can be used. Tikhonov regularized least-squares problem is in the form

$$\underset{\mathbf{f}}{\text{minimize}} \quad \|\mathbf{Hf}-\mathbf{g}\|^2 + \lambda \|\mathbf{Df}\|^2, \tag{3.4}$$

where λ is a regularization parameter. The operator $\mathbf{D}$ can be a forward difference operator, or the identity operator.

To solve Tikhonov regularized least-squares problem, we realize that (3.4) is equivalent to a damped least-squares minimization

$$\underset{\mathbf{f}}{\text{minimize}} \quad \left\| \begin{pmatrix} \mathbf{H} \\ \sqrt{\lambda}\mathbf{D} \end{pmatrix} \mathbf{f} - \begin{pmatrix} \mathbf{g} \\ \mathbf{0} \end{pmatrix} \right\|^2, \tag{3.5}$$

which can be solved using conjugate gradient methods.

The optimality condition of (3.5) implies the following normal equation

$$(\mathbf{H}^T\mathbf{H} + \lambda \mathbf{D}^T\mathbf{D})\mathbf{f} = \mathbf{H}^T\mathbf{g}. \tag{3.6}$$

Comparing (3.6) and (3.2), we note that in (3.6) the k-th eigenvalue of $\mathbf{H}^T\mathbf{H}$ is perturbed by the k-th eigenvalue of $\mathbf{D}^T\mathbf{D}$. Therefore, the smallest eigenvalue of $\mathbf{H}^T\mathbf{H} + \lambda\mathbf{D}^T\mathbf{D}$ is always larger than that of $\mathbf{H}^T\mathbf{H}$. Consequently, $\text{cond}(\mathbf{H}^T\mathbf{H} + \lambda\mathbf{D}^T\mathbf{D}) \leq \text{cond}(\mathbf{H}^T\mathbf{H})$. This explains why Tikhonov regularized least-squares problem is always more numerically stable than the original least-squares minimization.

The choice of the regularization parameter λ is an important research topic, but it is not the main subject of this chapter. We refer the readers to references such as the methods of generalized cross-validation and the L-curve methods [54, Chapter 7], [82], [55].

An extension of Tikhonov regularized least-squares is the simple bound constraints least-squares:

$$\begin{aligned} \underset{\mathbf{f}}{\text{minimize}} \quad & \|\mathbf{Hf}-\mathbf{g}\|^2 + \lambda \|\mathbf{Df}\|^2 \\ \text{subject to} \quad & l \leq \mathbf{f} \leq u, \end{aligned} \tag{3.7}$$

where l and u are the lower and upper bounds of the optimization variable $\mathbf{f}$. The bound constraints are motivated by the fact that physical devices cannot display pictures beyond the brightness limits: $0 \leq \mathbf{f} \leq 255$ for 8-bit gray scaled images, or $0 \leq \mathbf{f} \leq 1$ for images in the normalized scale.

In [61], Kim considered a special case of (3.7) where $\mathbf{f}$ is constrained by $\mathbf{f} \geq 0$ and the operator $\mathbf{D}$ is the identify operator:

$$\begin{aligned} \underset{\mathbf{f}}{\text{minimize}} \quad & \|\mathbf{Hf} - \mathbf{g}\|^2 + \lambda \|\mathbf{f}\|^2 \\ \text{subject to} \quad & \mathbf{f} \geq 0. \end{aligned} \tag{3.8}$$

(3.8) is known as the Non-Negative Least-Squares (NNLS). The algorithm Kim used to solve NNLS is the Primal Dual interior point method for Convex Objectives, abbreviated as `PDCO`. In each major iteration of `PDCO`, a linear system is solved (inexactly) using `LSQR` to determine a step. After a step is taken, the barrier parameter is reduced and a new system of linear equations has to be solved. The algorithm stops when the dual gap is reduced to a target value, or satisfies other stopping criteria.

3.3 Total Variation (Isotropic)

Since the introduction of the ROF model by Rudin, Osher and Fatemi in 1992 [95], total variation (TV) problem has been a popular research problem for more than a decade. Total Variation norm (TV-norm) is a norm defined as

$$\|f\|_{TV} = \sum_{u,v} \sqrt{|f_x(u,v)|^2 + |f_y(u,v)|^2},$$

where f_x and f_y are the partial derivatives of f with respect to x and y, respectively. Using matrix-vector notation, the total variation of an image is

$$\|\mathbf{f}\|_{TV} = \sum_i \sqrt{[\mathbf{f}_x]_i^2 + [\mathbf{f}_y]_i^2},$$

where $[\mathbf{x}]_i$ is the i-th component of the vector $\mathbf{x}$.

There are various ways of using TV-norms in image restoration problems, for

examples,

$$\begin{aligned} \underset{\mathbf{f}}{\text{minimize}} \quad & \|\mathbf{f}\|_{TV} \\ \text{subject to} \quad & \mathbf{Hf} = \mathbf{g}, \end{aligned} \tag{3.9}$$

or

$$\begin{aligned} \underset{\mathbf{f}}{\text{minimize}} \quad & \|\mathbf{f}\|_{TV} \\ \text{subject to} \quad & \|\mathbf{Hf} - \mathbf{g}\|^2 = \sigma^2. \end{aligned} \tag{3.10}$$

In either case, when Lagrangian of the constrained problem is considered, both (3.9) and (3.10) can be transformed to an unconstrained minimization problem

$$\underset{\mathbf{f}}{\text{minimize}} \quad \tfrac{1}{2}\|\mathbf{Hf} - \mathbf{g}\|^2 + \lambda\|\mathbf{f}\|_{TV}, \tag{3.11}$$

for some appropriate choice of λ.

Problems (3.9)-(3.11) and other variations are known as the TV minimization problems. Rudin, Osher and Fatemi [94, 95] tackled them by solving a time dependent partial differential equation on a manifold determined by the constraints. However, their approach is limited to the case where $\mathbf{H} = \mathbf{I}$. Later, Marquina and Osher [74] extended the idea to $\mathbf{H}$ with BCCB structures. Blomgren, Chan and Mulet [14], Chan, Golub and Mulet [22], Krishan, Lin and Yip [63] used interior point methods to solve Problem (3.11). Chambolle [17] used a gradient projection method to solve (3.11) for the case where $\mathbf{H} = \mathbf{I}$. The iterative shrinkage methods are discussed in [13], [41], [12] and [6].

3.4 Total Variation (Anisotropic)

Associated with the TV regularization is its l_1 approximation problem

$$\underset{\mathbf{f}}{\text{minimize}} \quad \tfrac{1}{2}\|\mathbf{Hf} - \mathbf{g}\|^2 + \lambda\|\mathbf{Df}\|_1. \tag{3.12}$$

The underlying approximation of this problem is that

$$\|\mathbf{f}\|_{TV} = \sum_i \sqrt{[\mathbf{D}_x\mathbf{f}]_i^2 + [\mathbf{D}_y\mathbf{f}]_i^2} \approx \sum_i |[\mathbf{D}_x\mathbf{f}]_i| + |[\mathbf{D}_y\mathbf{f}]_i| = \|\mathbf{Df}\|_1,$$

where $\mathbf{D} = [\mathbf{D}_x^T,\ \mathbf{D}_y^T]^T$. For clarity of the notation, we define

$$\|\mathbf{f}\|_{TV2} = \sum_i \sqrt{[\mathbf{D}_x\mathbf{f}]_i^2 + [\mathbf{D}_y\mathbf{f}]_i^2} \tag{3.13}$$

$$\|\mathbf{f}\|_{TV1} = \sum_i |[\mathbf{D}_x\mathbf{f}]_i| + |[\mathbf{D}_y\mathbf{f}]_i| = \|\mathbf{D}\mathbf{f}\|_1 . \tag{3.14}$$

Problems associated with $\|\mathbf{f}\|_{TV2}$ are known as the *isotropic* total variation, and problems associated with $\|\mathbf{f}\|_{TV1}$ are known as the *anisotropic* total variation.

Before the breakthrough of operator splitting method (See Ch. 4), there are only few numerical solvers for anisotropic total variation problems, partly due to the fact that the l_1 approximation tends to perform slightly worse than the original TV problem, as reported in [81], but mainly due to the fact that both isotropic TV (3.11) and the anisotropic TV (3.12) involve non-differentiable terms.

3.4.1 Difficulty of Solving TV problems

To demonstrate the difficulty of solving the TV problems (both isotropic and anisotropic), we extended the work of Kim and Saunders [61, 96] by using interior point methods (`PDCO`). For simplicity, we only consider the anisotropic case, and we denote $\mathbf{D}$ to be the horizontal forward difference operator only, i.e., $\mathbf{D} = \mathbf{D}_x$.

`PDCO` solves the following minimization

$$\begin{aligned} \underset{\mathbf{x},\mathbf{r}}{\text{minimize}} \quad & \phi(\mathbf{x}) + \tfrac{1}{2}\|\mathbf{D}_1\mathbf{x}\|_2^2 + \tfrac{1}{2}\|\mathbf{r}\|_2^2 \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{D}_2\mathbf{r} = \mathbf{b} \\ & l \le \mathbf{x} \le u, \end{aligned} \tag{3.15}$$

for some convex quadratic function $\phi(x)$, diagonal matrices $\mathbf{D}_1$ and $\mathbf{D}_2$, linear operator $\mathbf{A}$ and the bounds l, u. We want to reformulate (3.12) to the form of `PDCO`.

3.4.2 Primal Method

We first consider the primal problem of (3.12) by introducing an intermediate variable $\mathbf{u}$:

$$\begin{aligned} \underset{\mathbf{f},\mathbf{u}}{\text{minimize}} \quad & \frac{1}{2}\left\|\mathbf{Hf}-\mathbf{g}\right\|_2^2+\gamma\left\|\mathbf{Df}-\mathbf{u}\right\|_2^2+\lambda\|\mathbf{u}\|_1 \\ \text{subject to} \quad & 0\leq\mathbf{f}\leq 1, \end{aligned} \tag{3.16}$$

where γ is a parameter. The motivation is that if the constraint $\mathbf{Df}=\mathbf{u}$ is satisfied, then $\|\mathbf{Df}-\mathbf{u}\|_2^2$ vanishes. As a result, (3.16) and (3.12) coincides. (3.16) can be expressed as

$$\begin{aligned} \underset{\mathbf{f},\mathbf{u},\mathbf{r}_1,\mathbf{r}_2}{\text{minimize}} \quad & \tfrac{1}{2}\|\mathbf{r}_1\|_2^2+\tfrac{1}{2}\|\mathbf{r}_2\|_2^2+\lambda\|\mathbf{u}\|_1 \\ \text{subject to} \quad & \mathbf{Hf}+\delta_1\mathbf{r}_1=\mathbf{g} \\ & \mathbf{Df}-\mathbf{u}+\delta_2\mathbf{r}_2=0 \\ & 0\leq\mathbf{f}\leq 1 \end{aligned} \tag{3.17}$$

by introducing two slack variables $\mathbf{r}_1$ and $\mathbf{r}_2$ such that

$$\mathbf{Hf}+\delta_1\mathbf{r}_1=\mathbf{g} \qquad \text{and} \qquad \mathbf{Df}-\mathbf{u}+\delta_2\mathbf{r}_2=0.$$

Next, we partition the slack variable $\mathbf{u}$ into its positive and negative parts so that $\mathbf{u}=\mathbf{v}-\mathbf{w}$, where $\mathbf{v}=[\mathbf{u}]_+>0$ and $\mathbf{w}=[\mathbf{u}]_->0$. Therefore, the l_1-normed term in (3.17) can be rewritten as

$$\|\mathbf{u}\|_1=\mathbf{1}^T\mathbf{v}+\mathbf{1}^T\mathbf{w}.$$

Hence Problem (3.17) becomes

$$\begin{aligned} \underset{\mathbf{f},\mathbf{v},\mathbf{w},\mathbf{r_1},\mathbf{r_2}}{\text{minimize}} \quad & \frac{1}{2}\|\mathbf{r}_1\|_2^2+\frac{1}{2}\|\mathbf{r}_2\|_2^2+\lambda\mathbf{1}^T(\mathbf{v}+\mathbf{w}) \\ \text{subject to} \quad & \mathbf{Hf}+\delta_1\mathbf{r}_1=\mathbf{g} \\ & \mathbf{Df}-(\mathbf{v}-\mathbf{w})+\delta_2\mathbf{r}_2=0 \\ & 0\leq\mathbf{f}\leq 1 \end{aligned}$$

Re-scaling the constant λ, we have

$$\begin{aligned} \underset{\mathbf{f},\mathbf{v},\mathbf{w},\mathbf{r_1},\mathbf{r_2}}{\text{minimize}} \quad & \tfrac{1}{2}\|\mathbf{r}_1\|_2^2 + \tfrac{1}{2}\|\mathbf{r}_2\|_2^2 + \mathbf{1}^T(\mathbf{v}+\mathbf{w}) \\ \text{subject to} \quad & \mathbf{H}\mathbf{f} + \gamma_1\mathbf{r}_1 = \mathbf{g} \\ & \mathbf{D}\mathbf{f} - (\mathbf{v}-\mathbf{w}) + \gamma_2\mathbf{r}_2 = 0 \\ & 0 \le \mathbf{f} \le 1, \end{aligned} \tag{3.18}$$

where $\gamma_1 = \delta_1\sqrt{\lambda}$ and $\gamma_2 = \delta_2\sqrt{\lambda}$.

Finally, we realize that Problem (3.18) fits the form of `PDCO`, because

$$\underset{\mathbf{f},\mathbf{v},\mathbf{w},\mathbf{r_1},\mathbf{r_2}}{\text{minimize}} \quad \underbrace{\begin{pmatrix}\mathbf{0}^T & \mathbf{1}^T & \mathbf{1}^T\end{pmatrix}\begin{pmatrix}\mathbf{f}\\ \mathbf{v}\\ \mathbf{w}\end{pmatrix}}_{\phi(\mathbf{x})} + \frac{1}{2}\underbrace{\begin{pmatrix}\mathbf{r}_1^T & \mathbf{r}_2^T\end{pmatrix}}_{\mathbf{r}^T}\underbrace{\begin{pmatrix}\mathbf{r}_1\\ \mathbf{r}_2\end{pmatrix}}_{\mathbf{r}}$$

$$\text{subject to} \quad \underbrace{\begin{pmatrix}\mathbf{H} & \mathbf{0} & \mathbf{0}\\ \mathbf{D} & -\mathbf{I} & \mathbf{I}\end{pmatrix}}_{\mathbf{A}}\underbrace{\begin{pmatrix}\mathbf{f}\\ \mathbf{v}\\ \mathbf{w}\end{pmatrix}}_{\mathbf{x}} + \underbrace{\begin{pmatrix}\gamma_1\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \gamma_2\mathbf{I}\end{pmatrix}}_{\mathbf{D}_2}\underbrace{\begin{pmatrix}\mathbf{r}_1\\ \mathbf{r}_2\end{pmatrix}}_{\mathbf{r}} = \underbrace{\begin{pmatrix}\mathbf{g}\\ \mathbf{0}\end{pmatrix}}_{\mathbf{b}}$$

$$\begin{pmatrix}\mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\end{pmatrix} \le \begin{pmatrix}\mathbf{f}\\ \mathbf{v}\\ \mathbf{w}\end{pmatrix} \le \begin{pmatrix}\mathbf{1}\\ \infty\\ \infty\end{pmatrix}.$$

Theoretically, the above formulation would allow us to solve (3.12) using `PDCO`. However in practice, this is not possible for two reasons. First, the solution is very sensitive to the choice of parameters γ_1 and γ_2. Small perturbations of γ_1 and γ_2 is enough to cause divergence. Second, the matrix $\mathbf{A} = \begin{pmatrix}\mathbf{H} & \mathbf{0} & \mathbf{0}\\ \mathbf{D} & -\mathbf{I} & \mathbf{I}\end{pmatrix}$ is extremely large in dimension, which causes many inner iterations (`LSQR`) for each outer iteration in `PDCO`.

3.4.3 Dual Method

Another way to solve Problem (3.12) is to consider its dual:

$$\begin{aligned} \underset{\mu,\nu}{\text{maximize}} \quad & -\tfrac{1}{2}\|\mu\|_2^2 - \mu^T \mathbf{g} \\ \text{subject to} \quad & |\nu_i| \le \lambda, \quad \forall i, \\ & \mathbf{H}^T\mu + \mathbf{D}^T\nu = 0, \end{aligned} \tag{3.19}$$

where μ and ν are the dual variables. The derivation can be found in the Appendix A. With (3.19), the corresponding `PDCO` problem is

$$\begin{aligned} \underset{\mu,\nu,\mathbf{r}}{\text{minimize}} \quad & \phi(\mu,\nu) + \tfrac{1}{2}\left\|\mathbf{D}_1 \begin{pmatrix}\mu\\ \nu\end{pmatrix}\right\|^2 + \tfrac{1}{2}\|\mathbf{r}\|^2 \\ \text{subject to} \quad & \begin{pmatrix}\mathbf{H}^T & \mathbf{D}^T\end{pmatrix}\begin{pmatrix}\mu\\ \nu\end{pmatrix} + \mathbf{D}_2\mathbf{r} = 0, \\ & \begin{pmatrix}-\lambda\\ -\infty\end{pmatrix} \le \begin{pmatrix}\mu\\ \nu\end{pmatrix} \le \begin{pmatrix}\lambda\\ \infty\end{pmatrix}, \end{aligned}$$

where the convex function is $\phi(\mu,\nu)$, its gradient and hessian are defined as $\phi = \mathbf{g}^T\mu$, $\nabla\phi = [\mathbf{g}^T,\ \mathbf{0}]^T$ and $\nabla^2\phi = \mathbf{0}$, respectively. The matrix $\mathbf{A}$, diagonal matrices $\mathbf{D}_1$ and $\mathbf{D}_2$ in `PDCO` are respectively

$$\mathbf{A} = \begin{pmatrix}\mathbf{H}^T & \mathbf{D}^T\end{pmatrix}, \qquad \mathbf{D}_1 = \begin{pmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & 10^{-3}\mathbf{I}\end{pmatrix}, \qquad \mathbf{D}_2 = 10^{-3}\mathbf{I}.$$

The bounds on ν are numerically represented as $-10^{20} \le \nu \le 10^{20}$.

Fig. 3.1 shows the results of a deblurring problem

$$\underset{\mathbf{f}}{\text{minimize}} \quad \frac{1}{2}\|\mathbf{H}\mathbf{f} - \mathbf{g}\|^2 + \lambda\|\mathbf{D}\mathbf{f}\|_1,$$

where the operator $\mathbf{H}$ is characterized by the point spread function $h(x,y) = 1/10$ for $x = 0,\ldots,9$ and $y = 0$. The regularization parameter is $\lambda = 10^{-3}$. The observed image $\mathbf{g}$ is not corrupted with noise. As shown in the result, the recovered image is satisfactory. However, when the noise term is not zero, or when the length of point spread function is larger than 20 pixels, the computing time of `PDCO` increases exponentially.

Fig. 3.2 shows the results of a denoising problem

$$\underset{\mathbf{f}}{\text{minimize}} \;\; \frac{1}{2}\|\mathbf{f}-\mathbf{g}\|^2 + \lambda \|\mathbf{D}\mathbf{f}\|_1 ,$$

where $\lambda = 5 \times 10^{-1}$. The observed image $\mathbf{g}$ is corrupted with Gaussian noise with variance $\sigma^2 = 0.001$. Similar to the deblurring problem, the computing time of `PDCO` increases significantly when the noise level increases. While it is still possible to find a reasonable solution, careful tuning of the parameters is needed.

Original $\mathbf{f}$ Observed $\mathbf{g}$ Solution μ Solution ν

Figure 3.1: Deblurring the image "cameraman.tif" using `PDCO`. The image is blurred by a motion blur PSF, with $(v_x, v_y) = (10, 0)$, and is not corrupted with noise. The regularization parameter is chosen as $\lambda = 10^{-3}$.

Original $\mathbf{f}$ Observed $\mathbf{g}$ Solution μ Solution ν

Figure 3.2: Denoising the image "cameraman.tif" using `PDCO`. The image is corrupted by Gaussian noise, with variance $\sigma^2 = 0.001$. The regularization parameter is chosen as $\lambda = 5 \times 10^{-1}$.

3.4.4 Summary of TV Problems

To summarize our finding of `PDCO`, we find that `PDCO` is sensitive to parameters, such as γ_1, γ_2, the tolerance level for `LSQR` iterations, the rate of reduction of the barrier parameter, etc. Examples above are two exceptional cases where we can tune the

parameters appropriately. In most of other problems, as long as the image is changed, or the point spread function is changed, or even if the noise statistics is changed, the parameters need to be tuned again. Any slight deviation from the optimal parameter would cause divergence.

3.5 Bilateral TV Regularization

The last method that we consider in this chapter is the bilateral total variation (BTV), introduced by Farsiu, Robinson, Elad and Milanfar [37–39]. In 2009, Chan and Nguyen applied the same idea to LCD motion deblurring problems [20]. BTV problem has the following form:

$$\begin{aligned} \underset{\mathbf{f}}{\text{minimize}} \quad & \tfrac{1}{2}\left\|\mathbf{Hf}-\mathbf{g}\right\|^2 + \sum_{i=1}^{P}\sum_{j=1}^{Q}\lambda_{i,j}\left\|\mathbf{D}_{i,j}\mathbf{f}\right\|_1 \\ \text{subject to} \quad & l \le \mathbf{f} \le u. \end{aligned} \tag{3.20}$$

Clearly BTV problem is a generalization of the l_1-approximation problem. Since there are only few algorithms for the l_1-approximation problems, there are also very few algorithms for BTV problems. In the following paragraphs we present a sub-gradient projection algorithm [20].

3.5.1 Gradient Project Method

Gradient projection algorithm is an iterative method that updates the current solution by making a step along the direction of negative gradient. If the objective function is not differentiable, then a sub-gradient is used instead of the gradient. After a step is taken, the current solution is projected onto the constraint set to maintain feasibility.

To use the sub-gradient projection method for BTV problem, we first compute the sub-gradient of the objective function. Let

$$\phi(\mathbf{f}) = \frac{1}{2}\left\|\mathbf{Hf}-\mathbf{g}\right\|^2 + \sum_{i=1}^{P}\sum_{j=1}^{Q}\lambda_i\left\|\mathbf{D}_{i,j}\mathbf{f}\right\|_1,$$

the sub-gradient of $\phi(\mathbf{f})$ is

$$\nabla\phi(\mathbf{f}) = \mathbf{H}^T(\mathbf{H}\mathbf{f} - \mathbf{g}) + \sum_{i=1}^{P}\sum_{j=1}^{Q} \lambda_i \mathbf{D}_{i,j}^T \operatorname{sign}(\mathbf{D}_{i,j}\mathbf{f}),$$

where $\operatorname{sign}(x) = 1$ if $x > 0$, $\operatorname{sign}(x) = -1$ if $x < 0$, and $\operatorname{sign}(x) = 0$ if $x = 0$.

The projection is defined as

$$[\mathcal{P}(\mathbf{f})]_i = \begin{cases} u, & \text{if } [\mathbf{f}]_i \geq u, \\ l, & \text{if } [\mathbf{f}]_i \leq l. \end{cases} \tag{3.21}$$

where $[\mathbf{f}]_i$ denotes the i-th component of $\mathbf{f}$.

Given the k-th iterate f_k, the $k+1$-th iterate is

$$\mathbf{f}_{k+1} = \mathcal{P}[\mathbf{f}_k - \alpha_k \nabla\phi(\mathbf{f}_k)],$$

for some step size α_k which satisfies the "square summable but not summable" rule [11, 15, 100]:

$$\sum_{k=0}^{\infty} \alpha_k^2 < \infty \quad \text{and} \quad \sum_{k=0}^{\infty} \alpha_k = \infty.$$

Here, we choose $\alpha_k = M/(M+k)$, for some maximum number of iterations M, typically $M = 1000$.

3.5.2 Application of BTV

An application of the sub-gradient projection method is the liquid crystal display (LCD) motion blur reduction problem presented in [20]. LCDs are well known for its slow response time, caused by the slow phase orientation of the liquid crystals in the presence of electric field. Because of the slow response, fast moving objects in a scene are often perceived as blurred on an LCD. To resolve this issue, one method is to synthesize an image such that when it is displayed on the LCD, the perceived image looks sharp. Or in other words, we need to over-sharpen an image so that it compensates the distortion caused by the LCD. This problem is known as the inverse synthesis problem,

and mathematically it is expressed as a BTV problem

$$\begin{aligned} \underset{\mathbf{f}}{\text{minimize}} \quad & \|\mathbf{Hf}-\mathbf{g}\|_2^2 + \lambda \sum_{i,j} \|\mathbf{D}_{i,j}\mathbf{f}\|_1 \\ \text{subject to} \quad & 0 \le \mathbf{f} \le 1. \end{aligned}$$

Here, the image $\mathbf{g}$ is the target image, and $\mathbf{H}$ is the blur caused by the LCD. Our goal is to synthesize $\mathbf{f}$ so that the residue $\mathbf{Hf}-\mathbf{g}$ is minimized. The regularization function involves four terms: $\mathbf{D}_{1,0}\mathbf{f}$, $\mathbf{D}_{0,1}\mathbf{f}$, $\mathbf{D}_{1,1}\mathbf{f}$ and $\mathbf{D}_{-1,1}$. The parameter λ is set as $\lambda = 0.0015$.

Fig. 3.3 and Fig. 3.4 show two examples. In both cases, the blurring operation $\mathbf{H}$ is characterized by the motion PSF with $(v_x, v_y) = (3, 0)$. The sub-gradient projection method is compared with two state-of-the-art inverse synthesis algorithms, namely the modified Lucy-Richardson algorithm by Har-noy and Nguyen [56] and the motion compensated inverse filtering (MCIF) by Klompenhouwer and Velthoven [62]. Peak-signal-to-noise-ratio (PSNR) and the error $\sum_i \|\mathbf{D}_i\mathbf{f}\|_1$ is listed in Table 3.1. It can be seen that the BTV method reaches high PSNR while keeping the error $\sum_i \|\mathbf{D}_i\mathbf{f}\|_1$ low. In contrast, Lucy-Richardson method can give higher PSNR values, the error $\sum_i \|\mathbf{D}_i\mathbf{f}\|_1$ is also more.

The drawback of sub-gradient projection method is the slow convergence. In general, the rate of convergence depends on the spectral property of $\mathbf{H}$, the update scheme of step size, and the choice of sub-gradient.

Table 3.1: Comparisons between MCIF, Lucy Richardson and proposed method

Video Name	Methods	Peak Signal to Noise Ratio PSNR (dB)	Regularization Error $\sum_i \|\mathbf{D}_i\mathbf{f}\|_1$
Stockholm	Original	34.43	4.8286×10^3
	MC Inverse Filter [62]	34.357	9.8488×10^3
	Lucy Richardson [56]	40.35	1.0914×10^4
	BTV	36.38	4.1443×10^3
Shield	Original	36.879	3.586×10^3
	MC Inverse Filter [62]	36.943	7.432×10^3
	Lucy Richardson [56]	48.241	7.825×10^3
	BTV	38.540	3.437×10^3

3.6 Summary

In this chapter we reviewed several algorithms for image restoration from the perspective of regularization functions. For all discussed methods, the computation

(a) Original (b) MCIF (c) LR (d) Proposed

Figure 3.3: Inverse Synthesis Example 1: The upper row shows the synthesized signal that is sent to the LCD. The lower row shows the (simulated) perceived LCD signal. (a) Original Signal (b) Signal synthesized by MCIF [62], (c) Signal synthesized by Lucy Richardson [56], (d) Signal synthesized by solving BTV.

(a) Original (b) MCIF (c) LR (d) Proposed

Figure 3.4: Inverse Synthesis Example 2: The upper row shows the synthesized signal that is sent to the LCD. The lower row shows the (simulated) perceived LCD signal. (a) Original Signal (b) Signal synthesized by MCIF [62], (c) Signal synthesized by Lucy Richardson [56], (d) Signal synthesized by solving BTV.

time is long, because the differentiable function $\|\mathbf{Hf} - \mathbf{g}\|^2$ and the non-differentiable function $\|\mathbf{Df}\|_1$ are handled simultaneously. In next chapter, we show that by splitting the differentiable and non-differentiable functions into two different subproblems, the problem can be solved more efficiently.

3.7 Acknowledgment

Section 5 of this chapter, in part, is a reprint of the following paper

Stanley H. Chan and Truong Q. Nguyen, "LCD motion blur: modeling, analysis and algorithm," *IEEE Transactions on Image Processing*, vol. 20, issue 8, pp. 2352-65, Aug 2011.

Chapter 4

`deconvtv` for Image Restoration

This chapter presents a major contribution of the dissertation. We begin with the development of operator splitting methods, followed by the proposed augmented Lagrangian method. Properties of the algorithm including convergence, warm start and automatic parameter selection will be discussed consequently.

Our main focus in this chapter is the following TV problem with l_2-norm square objective:

$$\underset{\mathbf{f}}{\text{minimize}} \quad \frac{\mu}{2} \|\mathbf{Hf} - \mathbf{g}\|^2 + \|\mathbf{f}\|_{TV}. \tag{4.1}$$

Here, both isotropic and anisotropic TV norms will be discussed. We refer to this class of problems as TV/L2. We are also interested in the following TV problem with l_1-norm objective:

$$\underset{\mathbf{f}}{\text{minimize}} \quad \mu \|\mathbf{Hf} - \mathbf{g}\|_1 + \|\mathbf{f}\|_{TV}. \tag{4.2}$$

(4.2) is referred to as TV/L1 problem. Again, both isotropic and anisotropic TV norms will be discussed.

4.1 Operator Splitting Methods

This section provides an overview of the development of operator splitting methods. Without lost of generality, we will focus on the following anisotropic TV/L2 problem:

$$\underset{\mathbf{f}}{\text{minimize}} \quad \tfrac{\mu}{2} \|\mathbf{Hf} - \mathbf{g}\|^2 + \|\mathbf{Df}\|_1, \tag{4.3}$$

where $\mathbf{D} = [\mathbf{D}_x^T,\ \mathbf{D}_y^T]^T$.

Operator splitting method is an important milestone of solving TV problems. The idea is to separate the original TV problem into two (or more) relatively easy subproblems so that the solution of the original problem can be found by iteratively solving the subproblems. In the development of operator splitting methods, two major branches should be considered - "half quadratic penalty method" and "augmented Lagrangian method".

4.1.1 Half Quadratic Penalty Method

The half-quadratic penalty method is introduced by Geman, Reynolds and Yang [45, 46]. In this approach, Problem (4.3) is replaced by the equivalent problem

$$\underset{\mathbf{f},\mathbf{u}}{\text{minimize}} \quad \frac{\mu}{2} \|\mathbf{Hf} - \mathbf{g}\|^2 + Q(\mathbf{Df}, \mathbf{u}) + \psi(\mathbf{u}),$$

where $Q(\mathbf{t}, \mathbf{s})$ and $\psi(\mathbf{s})$ are chosen such that $Q(\mathbf{t}, \mathbf{s})$ is quadratic in $\mathbf{t}$. The functions $Q(\mathbf{t}, \mathbf{s})$ and $\psi(\mathbf{s})$ are related to $\|\mathbf{Dt}\|_1$ by

$$\|\mathbf{Dt}\|_1 = \underset{\mathbf{s}}{\text{minimize}} \quad Q(\mathbf{t}, \mathbf{s}) + \psi(\mathbf{s}).$$

The motivation of introducing the half-quadratic penalty function is to separate the smooth objective and the non-smooth regularization function. Similar approaches have been proposed, see, e.g., [3, 58, 83].

Huang, Ng and Wen [57] considered the following half-quadratic penalty function

$$\underset{\mathbf{f},\mathbf{u}}{\text{minimize}} \quad \frac{\mu}{2} \|\mathbf{Hf} - \mathbf{g}\|^2 + \frac{\alpha_k}{2} \|\mathbf{f} - \mathbf{u}\|^2 + \|\mathbf{Du}\|_1, \tag{4.4}$$

where $\{\alpha_k\}$ is an increasing sequence of penalty parameters. Given initial guesses $\mathbf{f}_0$ and

$\mathbf{u}_0$, the algorithm defines a sequence of $\mathbf{f}_k$ and $\mathbf{u}_k$ such that

$$\mathbf{f}_{k+1} = \underset{\mathbf{f}}{\operatorname{argmin}} \ \frac{\mu}{2}\|\mathbf{Hf}-\mathbf{g}\|^2 + \frac{\alpha_k}{2}\|\mathbf{f}-\mathbf{u}_k\|^2 \tag{4.5}$$

$$\mathbf{u}_{k+1} = \underset{\mathbf{u}}{\operatorname{argmin}} \ \frac{\alpha_k}{2}\|\mathbf{f}_{k+1}-\mathbf{u}\|^2 + \|\mathbf{Du}\|_1 . \tag{4.6}$$

This method is called Fast-TV. However, solving (4.6) is a difficult task. Although Huang *et al.* used a fast algorithm by Chambolle [17] to solve (4.6), the overall speed of Fast-TV is slow.

Wang *et al.* [112] considered another half-quadratic penalty function

$$\underset{\mathbf{f},\mathbf{u}}{\text{minimize}} \ \frac{\mu}{2}\|\mathbf{Hf}-\mathbf{g}\|^2 + \frac{\alpha_k}{2}\|\mathbf{Df}-\mathbf{u}\|^2 + \|\mathbf{u}\|_1 . \tag{4.7}$$

This method is named FTVd 3.0. The difference between Fast-TV and FTVd 3.0 is the way of splitting the variables. In Fast-TV, the additional intermediate term is $\|\mathbf{u}-\mathbf{f}\|^2$ whereas in FTVd 3.0, the term is $\|\mathbf{u}-\mathbf{Df}\|^2$. Due to the similarity between these two methods, it is expected that the subproblems are also solved similarly. For FTVd 3.0, the two subproblems are

$$\mathbf{f}_{k+1} = \underset{\mathbf{f}}{\operatorname{argmin}} \ \frac{\mu}{2}\|\mathbf{Hf}-\mathbf{g}\|^2 + \frac{\alpha_k}{2}\|\mathbf{Df}-\mathbf{u}_k\|^2 \tag{4.8}$$

$$\mathbf{u}_{k+1} = \underset{\mathbf{u}}{\operatorname{argmin}} \ \frac{\alpha_k}{2}\|\mathbf{Df}_{k+1}-\mathbf{u}\|^2 + \|\mathbf{u}\|_1 . \tag{4.9}$$

Note that closed form solution exists for (4.9), and hence FTVd 3.0 is faster than Fast-TV.

4.1.2 Augmented Lagrangian Method

The augmented Lagrangian method follows from the work of Rockafellar [91,92], Bertsekas [10, 11, 33], and research in solving optimization problems associated with partial differential equations [43, 47, 105, 106]. The idea of the augmented Lagrangian method is to transform the original unconstrained minimization problem (4.3) to an equivalent constrained problem:

$$\underset{\mathbf{f},\mathbf{u}}{\text{minimize}} \ \frac{\mu}{2}\|\mathbf{Hf}-\mathbf{g}\|^2 + \|\mathbf{u}\|_1 \quad \text{subject to } \mathbf{Df}=\mathbf{u}. \tag{4.10}$$

This constrained problem is solved using the properties of the augmented Lagrangian function

$$L(\mathbf{f}, \mathbf{u}, \mathbf{y}, \rho) = \frac{\mu}{2} \|\mathbf{Hf} - \mathbf{g}\|^2 + \|\mathbf{u}\|_1 - \mathbf{y}^T(\mathbf{u} - \mathbf{Df}) + \frac{\rho}{2} \|\mathbf{u} - \mathbf{Df}\|^2, \tag{4.11}$$

where the vector $\mathbf{y}$ approximates the Lagrange multipliers associated with the constraints $\mathbf{Df} = \mathbf{u}$, and ρ is a penalty parameter. To solve (4.11), one can solve a sequence of unconstrained subproblems in the form

$$\underset{\mathbf{f}_k, \mathbf{u}_k}{\text{minimize}} \quad L(\mathbf{f}_k, \mathbf{u}_k, \mathbf{y}_k, \rho), \quad \text{for } k = 1, 2, \ldots,$$

by using an alternating direction method (ADM)

$$\begin{aligned}
\mathbf{f}_{k+1} &= \underset{\mathbf{f}}{\text{argmin}} \quad \frac{\mu}{2} \|\mathbf{Hf} - \mathbf{g}\|^2 - \mathbf{y}_k^T(\mathbf{u}_k - \mathbf{Df}) + \frac{\rho}{2} \|\mathbf{u}_k - \mathbf{Df}\|^2 \\
\mathbf{u}_{k+1} &= \underset{\mathbf{u}}{\text{argmin}} \quad \|\mathbf{u}\|_1 - \mathbf{y}_k^T(\mathbf{u} - \mathbf{Df}_{k+1}) + \frac{\rho}{2} \|\mathbf{u} - \mathbf{Df}_{k+1}\|^2 \\
\mathbf{y}_{k+1} &= \mathbf{y}_k - \rho(\mathbf{u}_{k+1} - \mathbf{Df}_{k+1}).
\end{aligned}$$

The augmented Lagrangian method is named FTVd 4.0 [104]. Around the same period, the augmented Lagrangian method is also discussed in a number of papers, such as [2, 48, 117]. Esser [34], Goldstein and Osher [48], Wu and Tai [115] showed connections between the augmented Lagrangian method to split Bregman iterative methods.

4.1.3 Other Methods

There are other methods that are worth mentioning, such as iterative shrinkage/ thresholding (IST) [41], two-step IST (TwIST) [12], Fast IST algorithm (FISTA) [6] and sparse reconstruction by separable approximation (SpaRSA) [114]. However, these methods are more focused on sparse reconstruction problems, i.e., the special case of problem (4.3) where $\mathbf{D}$ is the identity operator.

4.2 Proposed Algorithm: `deconvtv` for TV/L2

In this section, we discuss the proposed algorithm for TV/L2 problems. The proposed algorithm is named `deconvtv`, which stands for deconvolution for total variation

problems. The algorithm shares a few common features with the existing augmented Lagrangian methods. The major difference is the parameter update scheme which will be discussed.

4.2.1 Overall Algorithm

To solve Problem (4.3), the proposed method follows from the classical augmented Lagrangian method by reformulating the unconstrained minimization problem as an equivalent constrained minimization problem:

$$\underset{\mathbf{f},\mathbf{u}}{\text{minimize}} \quad \frac{\mu}{2}\left\|\mathbf{Hf}-\mathbf{g}\right\|^2+\left\|\mathbf{u}_x\right\|_1+\left\|\mathbf{u}_y\right\|_1 \quad \text{subject to} \quad \mathbf{D}_x\mathbf{f}=\mathbf{u}_x, \quad \mathbf{D}_y\mathbf{f}=\mathbf{u}_y$$

For simplicity, we let $\mathbf{D}=[\mathbf{D}_x^T,\,\mathbf{D}_y^T]^T$ and $\mathbf{u}=[\mathbf{u}_x^T,\,\mathbf{u}_y^T]^T$. Thus, the problem is

$$\underset{\mathbf{f},\mathbf{u}}{\text{minimize}} \quad \frac{\mu}{2}\left\|\mathbf{Hf}-\mathbf{g}\right\|^2+\left\|\mathbf{u}\right\|_1 \quad \text{subject to} \quad \mathbf{Df}=\mathbf{u}. \tag{4.12}$$

From Bersekas [11, Chapter 2], (4.12) can be solved using the properties of the augmented Lagrangian function

$$L(\mathbf{f},\mathbf{u},\mathbf{y},\rho)=\frac{\mu}{2}\left\|\mathbf{Hf}-\mathbf{g}\right\|^2+\left\|\mathbf{u}\right\|_1-\mathbf{y}^T(\mathbf{u}-\mathbf{Df})+\frac{\rho}{2}\left\|\mathbf{u}-\mathbf{Df}\right\|^2. \tag{4.13}$$

Equation (4.13) states that the augmented Lagrangian function is the sum of three functions. The first term is the objective $\frac{\mu}{2}\left\|\mathbf{Hf}-\mathbf{g}\right\|^2+\left\|\mathbf{u}\right\|_1$. The second one is a function involving the Lagrange multipliers associated with the constraint $\mathbf{u}=\mathbf{Df}$, i.e., $\mathbf{y}^T(\mathbf{u}-\mathbf{Df})$. The sign of $\mathbf{y}^T(\mathbf{u}-\mathbf{Df})$ is negative, but a positive sign can also be used as long as other steps are consistent. The third term is the quadratic penalty $\frac{\rho}{2}\left\|\mathbf{u}-\mathbf{Df}\right\|^2$, where ρ is a regularization parameter. In (4.13), the Lagrange multipliers $\mathbf{y}$ can be partitioned as $\mathbf{y}=[\mathbf{y}_x^T,\,\mathbf{y}_y^T]^T$.

The motivation of using the augmented Lagrangian function is that the saddle point of $L(\mathbf{f},\mathbf{u},\mathbf{y},\rho)$ is also the solution of the constrained problem (4.13). To this end, we find the saddle point of $L(\mathbf{f},\mathbf{u},\mathbf{y},\rho)$ by solving a sequence of sub-problems

$$\underset{\mathbf{f}_k,\mathbf{u}_k}{\text{minimize}} \quad L(\mathbf{f}_k,\mathbf{u}_k,\mathbf{y}_k,\rho), \quad \text{for } k=1,2,\ldots, \tag{4.14}$$

with an update in $\mathbf{y}_k$ as

$$\mathbf{y}_{k+1} = \mathbf{y}_k - \rho(\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1}). \tag{4.15}$$

Problem (4.14) can be solved by considering the following two sub-problems

$$\mathbf{f}_{k+1} = \underset{\mathbf{f}}{\operatorname{argmin}} \ \frac{\mu}{2}\|\mathbf{H}\mathbf{f} - \mathbf{g}\|^2 - \mathbf{y}_k^T(\mathbf{u}_k - \mathbf{D}\mathbf{f}) + \frac{\rho}{2}\|\mathbf{u}_k - \mathbf{D}\mathbf{f}\|^2 \tag{4.16}$$

$$\mathbf{u}_{k+1} = \underset{\mathbf{u}}{\operatorname{argmin}} \ \|\mathbf{u}\|_1 - \mathbf{y}_k^T(\mathbf{u} - \mathbf{D}\mathbf{f}_{k+1}) + \frac{\rho}{2}\|\mathbf{u} - \mathbf{D}\mathbf{f}_{k+1}\|^2, \tag{4.17}$$

where (4.16) is known as the **f**-subproblem and (4.17) is known as the **u**-subproblem. The methods to solve these two sub-problems will be discussed in the next subsection.

The parameter ρ is increased by $\rho \leftarrow \gamma\rho$ for some constant $\gamma > 1$ when

$$\|\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1}\|_2 \geq \alpha\|\mathbf{u}_k - \mathbf{D}\mathbf{f}_k\|_2,$$

for some constant $0 < \alpha < 1$. Details will be discussed in the coming sections.

Two pseudo-algorithms are shown in Algorithms 1 and 2. Algorithm 1 is known as the *exact* method and Algorithm 2 is known as the *inexact* methods. The exact method requires $(\mathbf{f}_k, \mathbf{u}_k)$ to be solved simultaneously. It will be used in the convergence proof. In practice, the inexact method is preferred because it allows subproblems to be terminated in finite steps.

Algorithm 1 TV/L2 Proposed Method (Exact Version)

(Initialization) Input $\mathbf{f}_0$, $\mathbf{u}_0$, $\mathbf{y}_0$, ρ_0, γ and α.
while Not converge **do**
// Solve for $(\mathbf{f}_{k+1}, \mathbf{u}_{k+1})$
$(\mathbf{f}_{k+1}, \mathbf{u}_{k+1}) = \underset{\mathbf{f},\mathbf{u}}{\operatorname{argmin}} \ \frac{\mu}{2}\|\mathbf{H}\mathbf{f} - \mathbf{g}\|^2 + \|\mathbf{u}\|_1 - \mathbf{y}_k{}^T(\mathbf{u} - \mathbf{D}\mathbf{f}) + \frac{\rho_k}{2}\|\mathbf{u} - \mathbf{D}\mathbf{f}\|^2$
// Lagrange multiplier update
$\mathbf{y}_{k+1} = \mathbf{y}_k - \rho_k(\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1})$
// Penalty parameter update
$\rho_{k+1} = \gamma\rho_k$ if $\|\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1}\|_2 \geq \alpha\|\mathbf{u}_k - \mathbf{D}\mathbf{f}_k\|_2$
$k \leftarrow k + 1$
end while

Algorithm 2 TV/L2 Proposed Method (Inexact Version)

(Initialization) Input $\mathbf{f}_0$, $\mathbf{u}_0$, $\mathbf{y}_0$, ρ_0, γ and α
while not converged **do**
// Solve $\mathbf{f}$ subproblems
$\mathbf{f}_{k+1} = \underset{\mathbf{f}}{\text{argmin}} \ \frac{\mu}{2}\|\mathbf{Hf}-\mathbf{g}\|^2 - \mathbf{y}_k^T(\mathbf{u}_k - \mathbf{Df}) + \frac{\rho_k}{2}\|\mathbf{u}_k - \mathbf{Df}\|^2$
// Solve $\mathbf{u}$ subproblems
$\mathbf{u}_{k+1} = \underset{\mathbf{u}}{\text{argmin}} \ \|\mathbf{u}\|_1 - \mathbf{y}_k^T(\mathbf{u} - \mathbf{Df}_{k+1}) + \frac{\rho_k}{2}\|\mathbf{u} - \mathbf{Df}_{k+1}\|^2$
// Lagrange multiplier update
$\mathbf{y}_{k+1} = \mathbf{y}_k - \rho_k(\mathbf{u}_{k+1} - \mathbf{Df}_{k+1})$
// Penalty parameter update
$\rho_{k+1} = \gamma\rho_k$ if $\|\mathbf{u}_{k+1} - \mathbf{Df}_{k+1}\|_2 \geq \alpha\|\mathbf{u}_k - \mathbf{Df}_k\|_2$
$k \leftarrow k+1$
end while

4.2.2 f-subproblem

Minimizing the augmented Lagrangian function (4.13) involves solving two subproblems (4.16) and (4.17). In this subsection, we discuss the methods of solving the $\mathbf{f}$-subproblem.

The $\mathbf{f}$-subproblem is (we drop the iteration number k for simplicity)

$$\begin{aligned}\underset{\mathbf{f}}{\text{argmin}} \ L(\mathbf{f},\mathbf{u},\mathbf{y},\rho) &= \underset{\mathbf{f}}{\text{argmin}} \ \frac{\mu}{2}\|\mathbf{Hf}-\mathbf{g}\|^2 - \mathbf{y}^T(\mathbf{u}-\mathbf{Df}) + \frac{\rho}{2}\|\mathbf{u}-\mathbf{Df}\|^2 \\ &= \underset{\mathbf{f}}{\text{argmin}} \ \frac{1}{2}\left\| \begin{pmatrix} \sqrt{\mu}\mathbf{H} \\ \sqrt{\rho}\mathbf{D} \end{pmatrix} \mathbf{f} - \begin{pmatrix} \sqrt{\mu}\mathbf{g} \\ \sqrt{\rho}\mathbf{u} \end{pmatrix} \right\|_2^2 + \mathbf{y}^T\mathbf{Df}.\end{aligned}$$

The optimality conditions for this minimization problem imply that $\mathbf{f}$ may be found by solving the normal equation

$$\left(\mu\mathbf{H}^T\mathbf{H} + \rho\mathbf{D}^T\mathbf{D}\right)\mathbf{f} = \mu\mathbf{H}^T\mathbf{g} + \rho\mathbf{D}^T\mathbf{u} - \mathbf{D}^T\mathbf{y},$$

which gives the pseudo inverse solution

$$\mathbf{f} = \left(\mu\mathbf{H}^T\mathbf{H} + \rho\mathbf{D}^T\mathbf{D}\right)^{-1}(\mu\mathbf{H}^T\mathbf{g} + \rho\mathbf{D}^T\mathbf{u} - \mathbf{D}^T\mathbf{y}). \tag{4.18}$$

An important observation here is that if $\mathbf{H}$ is a block-circulant-with-circulant-block matrix (BCCB [61]), then $\mathbf{H} = \mathbf{F}^H\mathbf{\Lambda_H}\mathbf{F}$ where $\mathbf{F}$ is the discrete Fourier transform

matrix, and $\mathbf{\Lambda_H}$ is a diagonal matrix [80, Section 3.4]. Therefore, the inverse in (4.18) is

$$\begin{aligned}\left(\mu\mathbf{H}^T\mathbf{H}+\rho\mathbf{D}^T\mathbf{D}\right)^{-1} &= \left(\mu\mathbf{F}^H|\mathbf{\Lambda_H}|^2\mathbf{F}+\rho\mathbf{F}^H|\mathbf{\Lambda_D}|^2\mathbf{F}\right)^{-1}\\ &= \mathbf{F}^H\left(\mu|\mathbf{\Lambda_H}|^2+\rho|\mathbf{\Lambda_D}|^2\right)^{-1}\mathbf{F},\end{aligned}$$

where $\mathbf{\Lambda_D}$ is a diagonal matrix such that $\mathbf{D}=\mathbf{F}^H\mathbf{\Lambda_D}\mathbf{F}$. To summarize, the computation involved in solving (4.18) includes

1. Pre-calculate $\mathbf{\Lambda_H}$ and $\mathbf{\Lambda_D}$ using Fourier Transforms.
2. Apply Fourier Transform to $\mu\mathbf{H}^T\mathbf{g}+\rho\mathbf{D}^T\mathbf{u}-\mathbf{D}^T\mathbf{y}$.
3. Apply element-wise division to the result of step 2 by $\mu|\mathbf{\Lambda_H}|^2+\rho|\mathbf{\Lambda_D}|^2$.
4. Apply inverse Fourier Transform to the result of step 3.

If $\mathbf{H}$ is not a BCCB matrix but has some other structures such as a product of selection matrix and a discrete Fourier transform matrix (which is popularly used in compressive sensing), there are also fast methods to solve the $\mathbf{f}$-subproblem, see, e.g., [2]. If $\mathbf{H}$ is a general matrix, then we use an iterative algorithm to solve the $\mathbf{f}$-subproblem, for example conjugate gradient.

4.2.3 u-subproblem

The $\mathbf{u}$-subproblem involves minimizing $L(\mathbf{f},\mathbf{u},\mathbf{y},\rho)$ with respect to $\mathbf{u}$. It holds that

$$\begin{aligned}\underset{\mathbf{u}}{\operatorname{argmin}}\ \ L(\mathbf{f},\mathbf{u},\mathbf{y},\rho) &= \underset{\mathbf{u}}{\operatorname{argmin}}\ \ \|\mathbf{u}\|_1-\mathbf{y}^T(\mathbf{u}-\mathbf{D}\mathbf{f})+\frac{\rho}{2}\|\mathbf{u}-\mathbf{D}\mathbf{f}\|^2\\ &= \underset{\mathbf{u}}{\operatorname{argmin}}\ \ \|\mathbf{u}\|_1+\frac{\rho}{2}\left\|\mathbf{u}-(\mathbf{D}\mathbf{f}+\frac{1}{\rho}\mathbf{y})\right\|_2^2. \qquad (4.19)\end{aligned}$$

Since

$$\begin{aligned}x^* &= \underset{x}{\operatorname{argmin}}\ \ \frac{1}{2}(x-a)^2+\lambda|x|\\ &= \max\{|a|-\lambda,\ \ 0\}\operatorname{sign}(a).\end{aligned}$$

and (4.19) is a sum of independent sub-problems, the solution of (4.19) is

$$\mathbf{u} = \max\left\{\left|\mathbf{D}\mathbf{f} + \frac{1}{\rho}\mathbf{y}\right| - \frac{1}{\rho}, 0\right\} \cdot \operatorname{sign}\left(\mathbf{D}\mathbf{f} + \frac{1}{\rho}\mathbf{y}\right), \tag{4.20}$$

where "$\cdot$" denotes element-wise multiplication. We can also write the solution as

$$\begin{aligned}
\mathbf{u}_x &= \max\left\{\left|\mathbf{D}_x\mathbf{f} + \frac{1}{\rho}\mathbf{y}_x\right| - \frac{1}{\rho}, 0\right\} \cdot \operatorname{sign}\left(\mathbf{D}_x\mathbf{f} + \frac{1}{\rho}\mathbf{y}_x\right), \\
\mathbf{u}_y &= \max\left\{\left|\mathbf{D}_y\mathbf{f} + \frac{1}{\rho}\mathbf{y}_y\right| - \frac{1}{\rho}, 0\right\} \cdot \operatorname{sign}\left(\mathbf{D}_y\mathbf{f} + \frac{1}{\rho}\mathbf{y}_y\right),
\end{aligned}$$

where $\mathbf{u}$ is partitioned as $\mathbf{u} = [\mathbf{u}_x^T, \mathbf{u}_y^T]^T$ and $\mathbf{y}$ is partitioned as $\mathbf{y} = [\mathbf{y}_x^T, \mathbf{y}_y^T]^T$.

For the case of isotropic TV problems, the $\mathbf{u}$-subproblem is

$$\underset{\mathbf{u}}{\operatorname{argmin}} \;\; L(\mathbf{f}, \mathbf{u}, \mathbf{y}, \rho) \;\; = \;\; \underset{\mathbf{u}}{\operatorname{argmin}} \;\; \|\mathbf{u}\| + \frac{\rho}{2}\left\|\mathbf{u} - (\mathbf{D}\mathbf{f} + \frac{1}{\rho}\mathbf{y})\right\|_2^2,$$

where $\|\mathbf{u}\| = \sum_i \sqrt{[\mathbf{u}_x]_i^2 + [\mathbf{u}_y]_i^2}$. In this case, we let

$$\mathbf{v}_x = \mathbf{D}_x\mathbf{f} + \frac{1}{\rho}\mathbf{y}_x \qquad \text{and} \qquad \mathbf{v}_y = \mathbf{D}_y\mathbf{f} + \frac{1}{\rho}\mathbf{y}_y.$$

Then, by [69, lemma 4], we have

$$\begin{aligned}
\mathbf{u}_x &= \max\left\{\sqrt{|\mathbf{v}_x|^2 + |\mathbf{v}_y|^2} - \tfrac{1}{\rho}, 0\right\} \cdot \tfrac{\mathbf{v}_x}{\sqrt{|\mathbf{v}_x|^2 + |\mathbf{v}_y|^2 + \epsilon}} \\
\mathbf{u}_y &= \max\left\{\sqrt{|\mathbf{v}_x|^2 + |\mathbf{v}_y|^2} - \tfrac{1}{\rho}, 0\right\} \cdot \tfrac{\mathbf{v}_y}{\sqrt{|\mathbf{v}_x|^2 + |\mathbf{v}_y|^2 + \epsilon}},
\end{aligned} \tag{4.21}$$

where $|\cdot|$ is the complex modulus of the argument and $\epsilon = 10^{-8}$ is a constant. The division is an element-wise division.

4.3 Proposed Algorithm: `deconvtv` for TV/L1

The algorithm described in the preceding section is for TV/L2 problems. For TV/L1 problems, two intermediate variables $\mathbf{r}$ and $\mathbf{u}$ should be introduced. Hence,

Problem (4.2) (anisotropic case) is modified as

$$\begin{aligned} \underset{\mathbf{f},\mathbf{r},\mathbf{u}}{\text{minimize}} \quad & \mu \left\|\mathbf{r}\right\|_1 + \left\|\mathbf{u}\right\|_1 \\ \text{subject to} \quad & \mathbf{r} = \mathbf{H}\mathbf{f} - \mathbf{g} \\ & \mathbf{u} = \mathbf{D}\mathbf{f}. \end{aligned} \tag{4.22}$$

Similar to the TV/L2 problems, the augmented Lagrangian of (4.22) is considered:

$$\begin{aligned} L(\mathbf{f},\mathbf{r},\mathbf{u},\mathbf{y},\mathbf{z}) = \mu \left\|\mathbf{r}\right\|_1 + \left\|\mathbf{u}\right\|_1 \\ - \mathbf{z}^T(\mathbf{r} - \mathbf{H}\mathbf{f} + \mathbf{g}) + \frac{\rho_o}{2}\left\|\mathbf{r} - \mathbf{H}\mathbf{f} + \mathbf{g}\right\|^2 \\ - \mathbf{y}^T(\mathbf{u} - \mathbf{D}\mathbf{f}) + \frac{\rho_r}{2}\left\|\mathbf{u} - \mathbf{D}\mathbf{f}\right\|^2. \end{aligned}$$

Here, the variable $\mathbf{y}$ is the Lagrange multiplier associated with constraint $\mathbf{u} = \mathbf{D}\mathbf{f}$ and the variable $\mathbf{z}$ is the Lagrange multiplier associated with the constraint $\mathbf{r} = \mathbf{H}\mathbf{f} - \mathbf{g}$. The parameters ρ_o and ρ_r are two regularization parameters. The subscripts "o" and "r" stand for "**o**bjective", and "**r**egularization", respectively. TV/L1 solving involves three sub-problems, namely **f**-subproblem, **u**-subproblem and **r**-subproblem.

4.3.1 f-subproblem

The **f**-subproblem of TV/L1 is

$$\underset{\mathbf{f}}{\text{minimize}} \quad \frac{\rho_o}{2}\left\|\mathbf{r} - \mathbf{H}\mathbf{f} + \mathbf{g}\right\|^2 + \frac{\rho_r}{2}\left\|\mathbf{u} - \mathbf{D}\mathbf{f}\right\|^2 + \mathbf{z}^T\mathbf{H}\mathbf{f} + \mathbf{y}^T\mathbf{D}\mathbf{f}, \tag{4.23}$$

Using the first order optimality criteria, $\mathbf{f}$ can be found by considering the normal equation

$$(\rho_o\mathbf{H}^T\mathbf{H} + \rho_r\mathbf{D}^T\mathbf{D})\mathbf{f} = \rho_o\mathbf{H}^T\mathbf{g} + \mathbf{H}^T(\rho_o\mathbf{r} - \mathbf{z}) + \mathbf{D}^T(\rho_r\mathbf{u} - \mathbf{y}).$$

Similar to the TV/L2 case, if the matrix $\mathbf{H}$ is BCCB then $\mathbf{H}$ can be diagonalized using the DFT matrix. Therefore,

$$\mathbf{f} = \mathbf{F}^H(\rho_o|\mathbf{\Lambda_H}|^2 + \rho_r|\mathbf{\Lambda_D}|^2)^{-1}\mathbf{F}\left[\rho_o\mathbf{H}^T\mathbf{g} + \mathbf{H}^T(\rho_o\mathbf{r} - \mathbf{z}) + \mathbf{D}^T(\rho_r\mathbf{u} - \mathbf{y})\right].$$

4.3.2 u-subproblem

The **u**-subproblem of TV/L1 is

$$\underset{\mathbf{u}}{\text{minimize}} \quad \|\mathbf{u}\|_1 - \mathbf{y}^T(\mathbf{u} - \mathbf{Df}) + \frac{\rho_r}{2}\|\mathbf{u} - \mathbf{Df}\|^2,$$

which is the same as that of TV/L2. Therefore, the solution is

$$\begin{aligned}\mathbf{u}_x &= \max\left\{\left|\mathbf{D}_x\mathbf{f} + \frac{1}{\rho}\mathbf{y}_x\right| - \frac{1}{\rho}, 0\right\} \cdot \text{sign}\left(\mathbf{D}_x\mathbf{f} + \frac{1}{\rho}\mathbf{y}_x\right),\\ \mathbf{u}_y &= \max\left\{\left|\mathbf{D}_y\mathbf{f} + \frac{1}{\rho}\mathbf{y}_y\right| - \frac{1}{\rho}, 0\right\} \cdot \text{sign}\left(\mathbf{D}_y\mathbf{f} + \frac{1}{\rho}\mathbf{y}_y\right)\end{aligned}$$

for the anisotropic case, and

$$\begin{aligned}\mathbf{u}_x &= \max\left\{\sqrt{|\mathbf{v}_x|^2 + |\mathbf{v}_y|^2} - \tfrac{1}{\rho}, 0\right\} \cdot \tfrac{\mathbf{v}_x}{\sqrt{|\mathbf{v}_x|^2+|\mathbf{v}_y|^2+\epsilon}}\\ \mathbf{u}_y &= \max\left\{\sqrt{|\mathbf{v}_x|^2 + |\mathbf{v}_y|^2} - \tfrac{1}{\rho}, 0\right\} \cdot \tfrac{\mathbf{v}_y}{\sqrt{|\mathbf{v}_x|^2+|\mathbf{v}_y|^2+\epsilon}},\end{aligned} \tag{4.24}$$

for the isotropic case, where

$$\mathbf{v}_x = \mathbf{D}_x\mathbf{f} + \frac{1}{\rho}\mathbf{y}_x \qquad \text{and} \qquad \mathbf{v}_y = \mathbf{D}_y\mathbf{f} + \frac{1}{\rho}\mathbf{y}_y.$$

4.3.3 r-subproblem

Finally, the **r**-subproblem is

$$\underset{\mathbf{r}}{\text{minimize}} \quad \mu\|\mathbf{r}\|_1 - \mathbf{z}^T\mathbf{r} + \frac{\rho_o}{2}\|\mathbf{r} - \mathbf{Hf} + \mathbf{g}\|^2, \tag{4.25}$$

which is in the same form as the anisotropic **u**-subproblem. Therefore, using the shrinkage formula, the solution is

$$\mathbf{r} = \max\left\{\left|\mathbf{Hf} - \mathbf{g} + \frac{1}{\rho_o}\mathbf{z}\right| - \frac{\mu}{\rho_o}, 0\right\} \cdot \text{sign}\left(\mathbf{Hf} - \mathbf{g} + \frac{1}{\rho_o}\mathbf{z}\right). \tag{4.26}$$

4.3.4 Overall algorithm

In TV/L1, the Lagrange multipliers $\mathbf{y}$ and $\mathbf{z}$ are updated as

$$\begin{aligned}\mathbf{y}_{k+1} &= \mathbf{y}_k - \rho_r(\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1}),\\ \mathbf{z}_{k+1} &= \mathbf{z}_k - \rho_o(\mathbf{r}_{k+1} - \mathbf{H}\mathbf{f}_{k+1} + \mathbf{g}).\end{aligned} \tag{4.27}$$

The overall algorithm is shown in Algorithm 3.

Algorithm 3 Proposed Method (TV/L1)

Input $\mathbf{g}$, $\mathbf{H}$ and parameters μ. Let $k = 0$.
Set parameters ρ_r (default = 2), ρ_o (default = 100), α_o and α_r (default = 0.7).
Initialize $\mathbf{f}_0 = \mathbf{g}$, $\mathbf{u}_0 = \mathbf{D}\mathbf{f}_0$, $\mathbf{y}_0 = \mathbf{0}$, $\mathbf{r}_0 = \mathbf{H}\mathbf{f}_0 - \mathbf{g}$, $\mathbf{z}_0 = \mathbf{0}$.
Compute the matrices $\mathcal{F}[\mathbf{D}_x]$, $\mathcal{F}[\mathbf{D}_y]$, $\mathcal{F}[\mathbf{H}]$.
while not converge **do**
  // Solve $\mathbf{f}$ subproblems
  $\mathbf{f}_{k+1} = \underset{\mathbf{f}}{\operatorname{argmin}} \ \frac{\rho_o}{2}\|\mathbf{r}_k - \mathbf{H}\mathbf{f} + \mathbf{g}\|^2 + \frac{\rho_r}{2}\|\mathbf{u}_k - \mathbf{D}\mathbf{f}\|^2 + \mathbf{z}_k^T\mathbf{H}\mathbf{f} + \mathbf{y}_k^T\mathbf{D}\mathbf{f}$
  // Solve $\mathbf{u}$ subproblems
  $\mathbf{u}_{k+1} = \underset{\mathbf{u}}{\operatorname{argmin}} \ \|\mathbf{u}\|_1 - \mathbf{y}_k^T(\mathbf{u} - \mathbf{D}\mathbf{f}_{k+1}) + \frac{\rho_r}{2}\|\mathbf{u} - \mathbf{D}\mathbf{f}_{k+1}\|^2$
  // Solve $\mathbf{r}$ subproblems
  $\mathbf{r}_{k+1} = \underset{\mathbf{r}}{\operatorname{argmin}} \ \mu\|\mathbf{r}\|_1 - \mathbf{z}_k^T(\mathbf{r} - \mathbf{H}\mathbf{f}_{k+1} + \mathbf{g}) + \frac{\rho_o}{2}\|\mathbf{r} - \mathbf{H}\mathbf{f}_{k+1} + \mathbf{g}\|^2$
  // Lagrange multipliers update
  $\mathbf{y}_{k+1} = \mathbf{y}_k - \rho_r(\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1})$
  $\mathbf{z}_{k+1} = \mathbf{z}_k - \rho_o(\mathbf{r}_{k+1} - \mathbf{H}\mathbf{f}_{k+1} + \mathbf{g})$
  // Penalty parameter update
  $\rho_r \leftarrow \gamma_r\rho_r$ if $\|\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1}\|_2 \geq \alpha_r\|\mathbf{u}_k - \mathbf{D}\mathbf{f}_k\|_2$
  $\rho_o \leftarrow \gamma_o\rho_o$ if $\|\mathbf{r}_{k+1} - \mathbf{H}\mathbf{f}_{k+1} + \mathbf{g}\|_2 \geq \alpha_o\|\mathbf{r}_k - \mathbf{H}\mathbf{f}_k + \mathbf{g}\|_2$
  $k \leftarrow k + 1$
end while

4.4 Selecting Parameters

Regularization parameters are crucial in the proposed method. In this section, we discuss how to select parameters. We will focus on the TV/L2 problem.

4.4.1 Choosing μ

The regularization parameter μ trades off the least-squares error and the total variation penalty. Large values of μ tend to give sharper results, but noise will be

amplified. Small values of μ give less noisy results, but the image may be smoothed. The choice of μ is not known prior to solving the minimization. Empirically, a reasonable value of μ for a natural image typically lies in the range $[10^3, 10^5]$. Fig. 4.1 shows the recovery results by using different values of μ.

Input $\mu = 10^2$ $\mu = 10^3$ $\mu^* = 10352$ $\mu = 10^5$

Figure 4.1: Image recovery using different choices of μ. The optimal (in terms of PSNR compared to the reference) is $\mu = 10352$. The image is blurred by a Gaussian blur PSF of size 9×9, $\sigma = 5$. Gaussian noise is added to the image so that the blurred signal to noise ratio (BSNR) is 40dB.

One method to choose μ is to assume a noise model and estimate the noise power σ^2 [70, 71, 88]. Consequently, we can consider the constrained minimization problem

$$\underset{\mathbf{f}}{\text{minimize}} \quad \|\mathbf{Df}\|_1 \quad \text{subject to} \quad \|\mathbf{Hf} - \mathbf{g}\|^2 \leq \sigma^2. \tag{4.28}$$

Let μ^* be the optimal Lagrange multiplier of (4.28) and let $\mathbf{f}(\mu^*)$ be the corresponding solution. By complementarity conditions of problem (4.28), it holds that either $\|\mathbf{Hf}(\mu^*) - \mathbf{g}\|^2 = \sigma^2$, or $\mu^* = 0$. In other words, μ^* is either the root of the function $\|\mathbf{Hf}(\mu^*) - \mathbf{g}\|^2 - \sigma^2$, or $\mu^* = 0$. But if $\mu^* = 0$, then minimizing the Lagrangian of (4.28) is equivalent to minimizing $\|\mathbf{Df}\|_1$ and so the solution is $\mathbf{f}(\mu^*) = \mathbf{0}$. However, $\mathbf{f}(\mu^*) = \mathbf{0}$ is not meaningful in practice. Therefore, μ^* must be a root of $\|\mathbf{Hf}(\mu^*) - \mathbf{g}\|^2 = \sigma^2$ (or equivalently the root of $\|\mathbf{Hf}(\mu^*) - \mathbf{g}\| = \sigma$).

The complementarity condition suggests a method to determine μ^* for a given σ: let $\phi(\mu) = \|\mathbf{Hf}(\mu) - \mathbf{g}\|$, our goal is to find a root of $\phi(\mu) = \sigma$. This root-finding strategy is motivated by the work of Berg and Friedlander [9], where the authors considered an approximated Newton method by exploiting the duality between LASSO and basis pursuit problems. However, due to the presence of the differential operator $\mathbf{D}$ in problem (4.3), the approach that Berg and Friedlander used cannot be applied to our problem. Therefore, we propose to use a bisection method to find a root of $\phi(\mu) = \sigma$.

In Fig. 4.2, the blue solid line represents the trajectory $\phi(\mu)$, the red solid line is

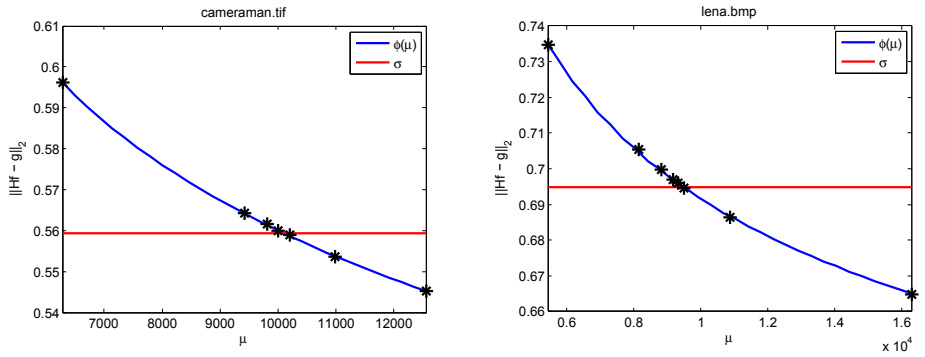


Figure 4.2: Bisection method to find μ. Starting with two values $\mu_a < \mu_b$, we let $\mu = (\mu_a + \mu_b)/2$ and evaluate $\phi(\mu)$. If $\phi(\mu) > \sigma$, then μ_a is replaced by μ. Otherwise μ_b is replaced by μ. The process repeats until the target tolerance level is reached. In this figure, the blue solid line $\phi(\mu)$ is numerically calculated based on a dense grid of μ.

Algorithm 4 Algorithm to determine μ

Given σ, $\mu_{\min}$, $\mu_{\max}$. Let $\mu_a = \mu_{\min}$, $\mu_b = \mu_{\max}$.
while $|\phi(\mu) - \sigma| > \texttt{tol}$ **do**
  Let $\mu = (\mu_a + \mu_b)/2$
  Solve $\mathbf{f}(\mu) = \underset{\mathbf{f}}{\operatorname{argmin}} \ \frac{\mu}{2}\|\mathbf{Hf} - \mathbf{g}\|^2 + \|\mathbf{Df}\|_1$
  Calculate $\phi(\mu) = \|\mathbf{Hf}(\mu) - \mathbf{g}\| - \sigma$
  if $\phi(\mu) > \sigma$ **then**
    $\mu_a \leftarrow \mu$
  else
    $\mu_b \leftarrow \mu$
  end if
end while

the target noise level σ, and the black markers are the bisection steps. At each bisection step, the algorithm calculates $\phi(\mu)$, where $\mu_a < \mu < \mu_b$ is an interval containing the optimal μ^*. If $\phi(\mu) > \sigma$, then we replace μ_a by μ. Otherwise if $\phi(\mu) < \sigma$, then we replace μ_b by μ. The algorithm stops when $|\phi(\mu) - \sigma| < \texttt{tol}$ for some tolerance level.

To increase the speed, we use a continuation scheme [53] by warm starting the algorithm. Typically, the number of bisection steps is between 5 to 10. The initial minimum and maximum μ are $\mu_a = 1$ and $\mu_b = 10^6$. Algorithm 4 summarizes the bisection method to determine the optimal μ^*.

4.4.2 Choosing ρ, γ and α

One of the major differences between the proposed algorithm and FTVd 4.0 [104] is the update of ρ, because ρ is a fixed constant in [104]. However, as mentioned in [87], the method of multipliers shows a faster rate of convergence by adopting the following parameter update scheme:

$$\rho_{k+1} = \begin{cases} \gamma\rho_k, & \text{if } \|\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1}\|_2 \geq \alpha\|\mathbf{u}_k - \mathbf{D}\mathbf{f}_k\|_2, \\ \rho_k, & \text{otherwise.} \end{cases} \tag{4.29}$$

Here, the condition $\|\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1}\|_2 \geq \alpha\|\mathbf{u}_k - \mathbf{D}\mathbf{f}_k\|_2$ specifies the constraint violation with respect to a constant α. The intuition is that the quadratic penalty $\frac{\rho}{2}\|\mathbf{u} - \mathbf{D}\mathbf{f}\|^2$ is a convex surface added to the original objective function $\mu\|\mathbf{H}\mathbf{f} - \mathbf{g}\|^2 + \|\mathbf{u}\|_1$ so that the problem is guaranteed to be strongly convex [91]. Ideally, the residue $\frac{\rho}{2}\|\mathbf{u}_k - \mathbf{D}\mathbf{f}_k\|^2$ should decrease as k increases. However, if $\frac{\rho}{2}\|\mathbf{u}_k - \mathbf{D}\mathbf{f}_k\|^2$ is not decreasing for some reasons, one can increase the weight of the penalty $\frac{\rho}{2}\|\mathbf{u} - \mathbf{D}\mathbf{f}\|^2$ relative to the objective so that $\frac{\rho}{2}\|\mathbf{u} - \mathbf{D}\mathbf{f}\|^2$ is forced to be reduced. Therefore, given α and γ where $0 < \alpha < 1$ and $\gamma > 1$, Equation (4.29) guarantees that the constraint violation is decreasing asymptotically. In the steady state as $k \to \infty$, ρ becomes a constant [10].

The initial value of ρ is chosen to be within the range of [2, 10]. This value cannot be large (in the order of 100), because the role of the quadratic surface $\|\mathbf{u} - \mathbf{D}\mathbf{f}\|^2$ is to perturb the original objective function so that it becomes strongly convex. If the initial value of ρ is too large, the solution of the original problem may not be found. However, ρ cannot be too small either, for otherwise the effect of the quadratic surface $\|\mathbf{u} - \mathbf{D}\mathbf{f}\|^2$ becomes negligible. Empirically, we find that $\rho = 2$ is robust to most restoration problems.

Table 4.1 illustrates the sensitivity of the algorithm to the parameters ρ, γ and α. In this test, twenty images are blurred by a Gaussian blur PSF of size 9×9 with variance $\sigma = 1$. The blurred signal to noise ratio (BSNR) is 30dB. For each image, two of the three parameters (ρ, γ and α) are fixed at their default values $\rho = 2$, $\gamma = 2$, $\alpha = 0.7$, whereas one of them is varying within the range specified in Table 4.1. The stopping criteria of the algorithm is $\|\mathbf{f}_{k+1} - \mathbf{f}_k\|_2/\|\mathbf{f}_k\| \leq 10^{-3}$ and $\mu = 10^4$ for all images. The maximum PSNR, minimum PSNR and the difference are reported in Table 4.1. Referring to the values, it can be calculated that the average maximum-to-minimum PSNR differences

among all twenty images for ρ_r, γ and α are 0.311dB, 0.208dB and 0.357dB respectively. For an average PSNR difference in the order of 0.3dB, the perceivable difference is small[1].

Table 4.1: Sensitivity Analysis of Parameters. Maximum and minimum PSNR (dB) for a range of ρ, γ and α. If a parameter is not the variable, it is fixed at the default values: $\rho = 2$, $\gamma = 2$, $\alpha = 0.7$.

Image no.	$1.5 \leq \rho \leq 10$			$1 \leq \gamma \leq 5$			$0.5 \leq \alpha \leq 0.9$		
	Max	Min	Diff	Max	Min	Diff	Max	Min	Diff
1	28.6468	28.8188	0.1719	28.6271	28.7931	0.1661	28.5860	28.8461	0.2601
2	31.3301	31.4858	0.1556	31.7720	32.0908	0.3188	31.0785	31.5004	0.4219
3	31.7009	31.9253	0.2244	31.9872	32.0847	0.0976	31.7238	31.9833	0.2596
4	33.6080	33.8427	0.2346	33.9994	34.0444	0.0450	34.1944	34.6197	0.4252
5	36.2843	36.5184	0.2341	36.1729	36.3173	0.1444	35.9405	36.7737	0.8332
6	32.0193	32.3859	0.3666	32.2805	32.4795	0.1990	31.9998	32.4207	0.4208
7	29.2861	29.7968	0.5107	29.5890	29.7408	0.1518	29.8872	30.1685	0.2813
8	30.0598	30.4347	0.3749	29.6344	29.9748	0.3404	29.4519	29.7627	0.3108
9	34.4951	34.7675	0.2724	34.5234	34.7378	0.2144	34.3567	34.9726	0.6159
10	29.5555	30.1231	0.5676	29.3502	29.5715	0.2213	29.4009	29.6558	0.2549
11	28.6291	29.1908	0.5617	28.6711	28.9846	0.3135	28.7760	29.0099	0.2340
12	31.6657	31.7473	0.0815	31.2254	31.3172	0.0918	31.3596	31.5423	0.1827
13	35.5306	35.9015	0.3710	35.4584	35.7442	0.2858	36.0163	36.2163	0.2000
14	36.8008	36.9204	0.1196	37.1039	37.1956	0.0917	36.6822	37.1470	0.4648
15	32.0469	32.0969	0.0501	32.4076	32.5918	0.1843	32.0101	32.5421	0.5320
16	31.5836	31.6572	0.0736	31.5975	31.9582	0.3607	31.3778	31.6027	0.2249
17	32.2500	32.6248	0.3748	32.8744	33.0967	0.2223	32.5141	32.8665	0.3524
18	32.6311	33.0377	0.4066	32.2999	32.5472	0.2473	32.9494	33.1908	0.2414
19	28.4927	29.1870	0.6943	28.6654	28.8488	0.1834	28.7902	29.0220	0.2318
20	30.2615	30.6387	0.3771	30.3235	30.6007	0.2772	30.3351	30.7206	0.3855

4.5 Convergence

This section presents the convergence property of the algorithm. We discuss the empirical convergence, followed by the proof.

4.5.1 Empirical Convergence

Fig. 4.3 illustrates the convergence profile of the TV/L2 algorithm in a typical image recovery problem. In this test, the image "cameraman.tif" (size 256×256, gray-scaled) is blurred by a Gaussian blur PSF of size 9×9 and $\sigma = 1$. Gaussian noise is added so that the blurred signal to noise ratio (BSNR) is 40dB. To visualize the effects of the parameter update scheme, we set the initial value of ρ to be $\rho = 2$, and let $\alpha = 0.7$. Referring to (4.29), ρ is increased by a factor of γ if the condition is satisfied. Note that [104](FTVd 4.0) is a special case when $\gamma = 1$, whereas the proposed algorithm

[1]It should be noted that the optimization problem is identical for all parameter settings. Therefore, the correlation between the PSNR and visual quality is high.

allows the user to vary γ.

In Fig. 4.3, the y-axis is the objective value $\frac{\mu}{2}\|\mathbf{H}\mathbf{f}_k - \mathbf{g}\|^2 + \|\mathbf{f}_k\|_{TV}$ for the k-th iteration, and the x-axis is the iteration number k. As shown in the figure, an appropriate choice of γ improves the rate of convergence significantly. However, if γ is too large, the algorithm is not converging to the solution. Empirically, we find that $\gamma = 2$ is robust to most of the image and video problems.

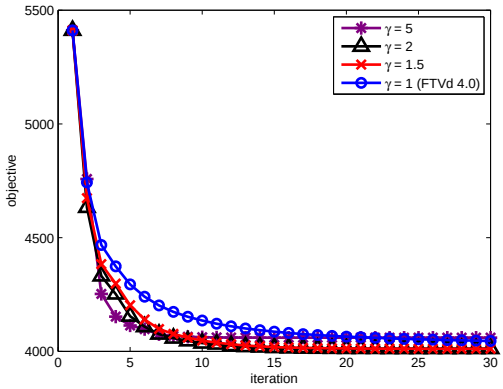


Figure 4.3: Convergence profile of the proposed algorithm for deblurring the image "cameraman.tif". The four colored curves show the rate of convergence using different values of γ, where γ is the multiplication factor for updating ρ_r.

4.5.2 Convergence Proof

The overall idea of the convergence proof is as follows. There are two possible situations about the penalty parameter ρ_k: either (1) $\rho_k \to \infty$, or (2) $\rho_k \to \rho^*$ for some $\rho^* < \infty$. The first half of the proof is dealing with the case where $\rho_k \to \infty$. We show that Algorithm 1 and 2 converge in this case. The second half of the proof is dealing with the case where $\rho_k \to \rho^*$. In this case, the proof follows from Eckstein and Bertsekas [33].

The proof is given in Appendix B.

4.6 Warm Start

The proposed algorithm can be warm started. By warm start we meant to use the solution of one problem as the initial guess to another problem. Specifically, if the

solution of the previous problem is

$$(\mathbf{f}^{(1)}, \mathbf{u}^{(1)}) = \underset{\mathbf{f},\mathbf{u}}{\operatorname{argmin}} \ \frac{\mu}{2} \|\mathbf{Hf} - \mathbf{g}\|^2 + \|\mathbf{u}\|_1 - \mathbf{y}^T(\mathbf{u} - \mathbf{Df}) + \frac{\rho}{2} \|\mathbf{u} - \mathbf{Df}\|^2,$$

and the associated Lagrange multiplier is $\mathbf{y}^{(1)}$ and the penalty parameter is $\rho^{(1)}$, then the initial guess to the next problem is $(\mathbf{f}^{(1)}, \mathbf{u}^{(1)}, \mathbf{y}^{(1)}, \rho^{(1)})$. Note that using $\rho^{(1)}$ is crucial to the convergence. If the initial ρ for the next problem is not $\rho^{(1)}$, but some small values (e.g., $\rho = 2$ in our experiments), then the effect of warm start is not so apparent because for small ρ, the rate of cost reduction relies mainly on the Lagrange multiplier term $\mathbf{y}_k^T(\mathbf{u}_k - \mathbf{Df}_k)$. However, if $\rho^{(1)}$ is too large, then we might lose track in the first few iterations of the next problem. Therefore, a practical strategy is to set the initial guess as $\min\{\rho^{(1)}, c\}$, for some constant $c > 1$. A typical value of c is $c = 32$.

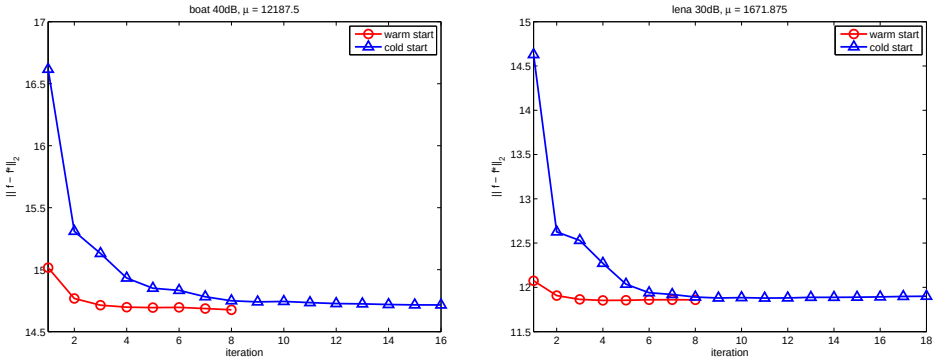


Figure 4.4: Warm and cold starting the algorithm for image restoration problems. The computing time is reduced by using the warm start.

To illustrate the effectiveness of warm start, we show some results in Fig. 4.4. In this experiment, we construct two observations $\mathbf{g}_1$ and $\mathbf{g}_2$ by adding two different noise vectors to the ground truth image $\mathbf{f}$, i.e., $\mathbf{g}_1 = \mathbf{f} + \eta_1$ and $\mathbf{g}_2 = \mathbf{f} + \eta_2$, where $\eta_1 \neq \eta_2$. Our goal is to recover $\mathbf{f}$ from $\mathbf{g}_1$, and $\mathbf{f}$ from $\mathbf{g}_2$. Suppose that $(\mathbf{f}^{(1)}, \mathbf{u}^{(1)}, \mathbf{y}^{(1)}, \rho^{(1)})$ is the solution recovered from $\mathbf{g}_1$. To recover an image from observation $\mathbf{g}_2$, we can either warm start the algorithm by using $\mathbf{f}_0 = \mathbf{f}^{(1)}$, $\mathbf{u}_0 = \mathbf{u}^{(1)}$, $\mathbf{y}_0 = \mathbf{y}^{(1)}$, $\rho_0 = \rho^{(1)}$ as the initial guess, or we can cold start the algorithm with $\mathbf{f}_0 = \mathbf{g}_2$, $\mathbf{u}_0 = \mathbf{0}$, $\mathbf{y}_0 = \mathbf{0}, \rho_0 = 1$ as the initial guess. As the results in Fig. 4.4 indicate, using warm start reduces the computing time significantly.

Warm start is effective only when the perturbation is small, for example, the

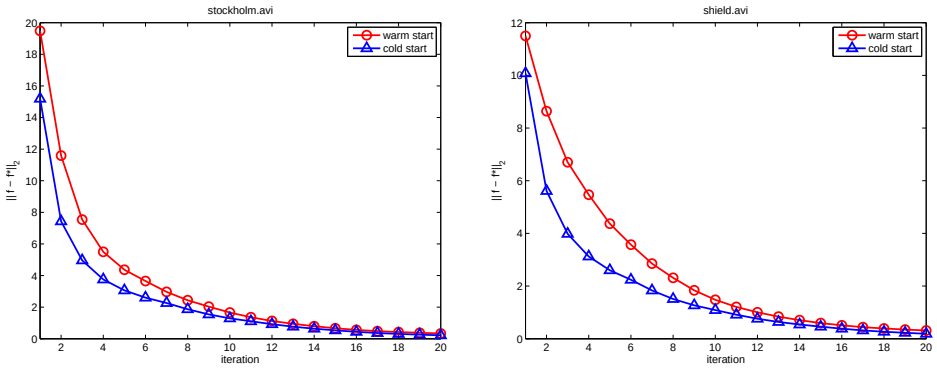


Figure 4.5: Warm and cold starting the algorithm for two video restoration problems (frame 1 and 2 of `stockholm.avi` and `shield.avi`). In these two video problems, we motion compensate the solution of the previous frame and use it as the initial guess to the current frame. The results show that warm start is not suitable for video restoration problems because motion compensation causes error. Note that the two video sequences in this figure are originally blurred and so there is no ground truth $\mathbf{f}^*$. $\mathbf{f}^*$ is obtained by solving the problem with extremely tight tolerance level `tol = 1e-9`.

bisection method mentioned in the previous subsection. However, warm start is not suitable for restoring videos. To use warm start in restoring a video, we need to apply motion compensation to the solution of the previous frame before using it as the initial guess to the current frame. However, motion compensation causes error, especially if there are local motions, occlusions or sub-pixel displacements across two consecutive frames. Although one can use advanced motion compensation algorithms to reduce the motion compensation error, these algorithms are extremely complex and so the time spending on running these algorithms is even longer than starting the proposed algorithm from scratch.

Fig. 4.5 compares the performance of warm starting and cold starting. The motion map is found using [21] and the motion compensation is performed using bicubic interpolation. The block size is 8×8, and the searching accuracy is 0.25 pixel. It can be seen that warm starting the algorithm performs worse than starting the algorithm from scratch.

4.7 Numerical Results

In this section we compare the proposed method with other existing methods. To begin with, we define the peak signal to noise ratio (PSNR) and the blurred signal

to noise ratio (BSNR). PSNR measures the error between the recovered image and the ground truth. A higher PSNR value usually implies better image quality. BSNR is the ratio between the power of the observed image and the power of the noise. A higher BSNR value usually implies lower observation noise. Mathematically, PSNR and BSNR are defined as

$$\text{BSNR} = 10\log_{10}\frac{\|\mathbf{g}\|^2}{\|\eta\|^2}, \quad \text{PSNR} = 10\log_{10}\frac{1}{\text{MSE}},$$

where $\mathbf{g}$ is the observed image and η is the noise. The mean square error (MSE) is given by $\text{MSE} = \frac{1}{N}\left\|\mathbf{f} - \tilde{\mathbf{f}}\right\|^2$, where N is the number of pixels in the image $\mathbf{f}$. $\tilde{\mathbf{f}}$ is the ground truth image.

In most of the experiments below, we mainly test for the deblurring problem where images are blurred by a Gaussian blur PSF of size 9×9, variance 5, and added with noise so that BSNR = 40dB. The Gaussian blur PSF can be implemented in MATLAB as `fspecial{'gaussian', [9 9], 5}`. Other types of PSFs can also be used, but the results are similar.

The default parameters of the proposed algorithm are $\rho_0 = 2$, $\gamma = 2$, $\alpha = 0.7$. These values are found empirically that balances PSNR and run time.

All experiments in this paper were run on a Dell-XPS PC, with Intel Q9550 Qual Core 2.8GHz, 4GB DDR3 RAM, Windows 7 (64 bit), MATLAB 2009a.

4.7.1 Compare with standard deblurring algorithms

First, we compare the proposed method with standard image deblurring algorithms, namely Wiener deconvolution, Lucy-Richardson deconvolution and regularized least-squares deconvolution. In MATLAB, these functions are known as `deconvwnr`, `deconvlucy`, and `deconvreg`.

The experiment setting is as follows. We first construct a blurry image by applying a Gaussian blur PSF (of size 9×9, and variance 5) to the image. We also add noise to the image so that the BSNR is 40dB. Then we recover the image using the proposed algorithm (with $\mu = 5\times10^3$, $\alpha = 0.7$, $\rho_0 = 2$, $\gamma = 2$), `deconvwnr` (with damping constant $\beta = 2 \times 10^{-3}$), `deconvlucy` (with damping constant $\beta = 2 \times 10^{-3}$, 25 iterations), and `deconvreg` (with damping constant $\beta = 0.25$).

The results in Fig. 4.6 show that the proposed algorithm achieves the highest PSNR value compared to other three standard deblurring algorithms.

original image | observed image | proposed method 27.07dB, 0.24 sec

`deconvwnr` 25.73dB, 0.02sec | `deconvlucy` 24.99dB, 0.51 sec | `deconvreg` 26.53dB, 0.12 sec

Figure 4.6: Comparisons with three standard algorithms: `deconvwnr`, `deconvlucy`, `deconvreg`. In each of the sub-figures, we show the PSNR value (dB) and the run time (sec).

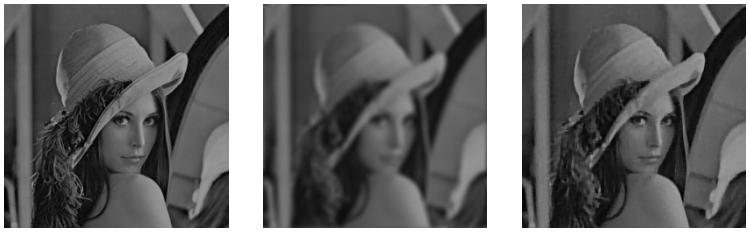
4.7.2 Compare with half-quadratic methods

In this experiment, we compare the proposed method with two half-quadratic penalty methods, namely Fast-TV [57] and FTVd 3.0 [112] [2]. Indirect comparisons can be concluded from the experiments in [57] and [112]: in [57], it has been shown that Fast-TV has higher PSNRs and shorter computing time when compared to TV-Bect [7], Modified-TV [109], two-step iterative shrinkage/ thresholding TwIST [12], and an interior point method [79]; in [112], it has been shown that FTVd 3.0 performs better than ForWaRD [78]. Therefore, comparing with FTVd 3.0 and Fast-TV allows us to conclude the performance of the proposed algorithm compared to these state-of-the-art algorithms.

Fig. 4.8 shows the blurred images [3] and the restored images by using the proposed method, FTVd 3.0 and FastTV. In this experiment, the algorithms are terminated when the relative change satisfies $\|\mathbf{f}_{k+1} - \mathbf{f}_k\| / \|\mathbf{f}_k\| < 10^{-3}$. For FTVd 3.0 and Fast-TV, the $\mathbf{f}$ and $\mathbf{u}$ subproblems are terminated when the relative change is less than 10^{-3}. The

[2] MATLAB code is available at http://www.caam.rice.edu/~optimization/L1/ftvd/v3.0/

[3] There are totally 20 images in this experiment. Dataset can be downloaded at http://videoprocessing.ucsd.edu/~stanleychan



original image | observed image | proposed method 32.76dB, 0.24sec

`deconvwnr` 28.96dB, 0.05sec | `deconvlucy` 30.64dB, 0.82sec | `deconvreg` 31.11dB, 0.12sec

Figure 4.7: Comparisons with three standard algorithms: `deconvwnr`, `deconvlucy`, `deconvreg`. The image is blurred by a Gaussian blur PSF of size 9×9, with variance 5.

regularization parameter μ in Problem (4.3) is the same for all three algorithms: $\mu = \frac{0.1}{\|\eta\|}$, where η is the noise vector. As shown in Fig. 4.8, the images recovered by the proposed algorithm are competitive with those recovered by FTVd 3.0 and FastTV. Table 4.2 lists the PSNRs of the three methods [4].

The run time of the three methods is also listed in Table 4.2. Overall speaking, the proposed algorithm is around twice as fast as FTVd 3.0, and is around 10 to 100 times faster than FastTV. As mentioned, since FTVd 3.0 and FastTV are already more efficient than a number of other existing algorithms, we can conclude that the proposed method has a state-of-the-art speed.

Fig. 4.9(a) shows the PSNR value as a function of the iteration number. In FTVd 3.0, since a continuation scheme is applied, the PSNR shows a "stair-case" characteristic. Each jump corresponds to a major iteration, and between each jump there are a number of minor iterations. Therefore, the total number of iterations is the product of the number of major iterations and the number of minor iterations. In contrast, the proposed method does not require any continuation scheme to control the penalty parameter. The PSNR can reach the peak and maintain in a steady state in only few number of iterations.

[4]Complete list of results can be found in the supplementary document

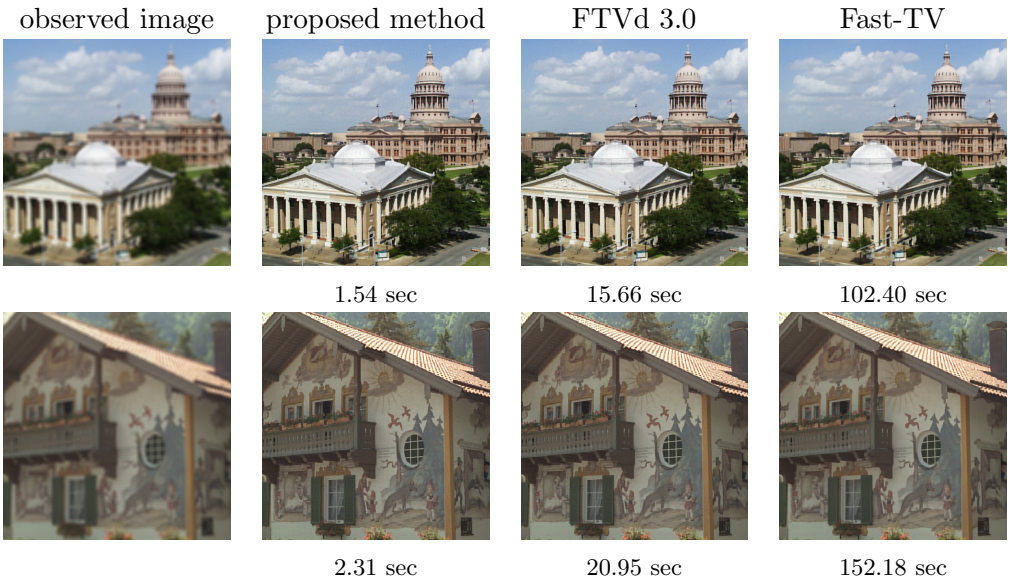


Figure 4.8: Deblurring results of `Building2.bmp` and `House2.bmp`. The images are blurred by a Gaussian blur PSF with size 9×9, and variance 5, BSNR = 40dB. PSNR and run time are listed in Table 4.2. (See supplementary document for more results.)

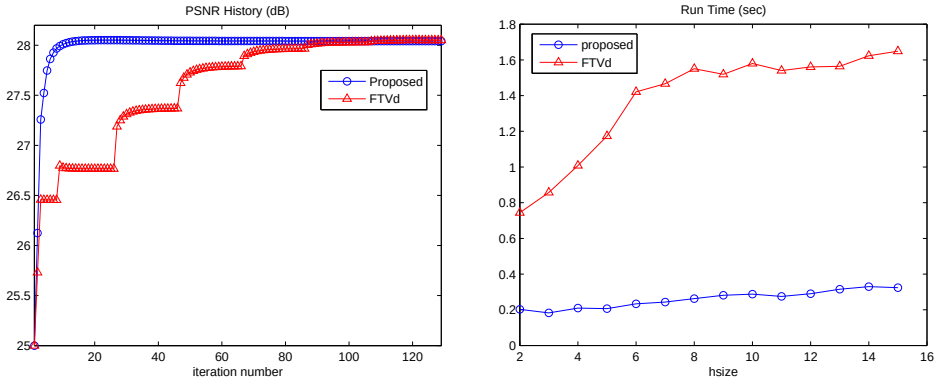
In Fig. 4.9(b), we show the run time as a function of the size of the point spread function. Since increasing the size of the point spread function increases the condition number of the blur operator $\mathbf{H}$, the number of iterations is expected to increase. However, the rate of change of the increment using the proposed method is lower than that of FTVd 3.0. This is because FTVd 3.0 does not only involve the major iterations, but also the minor iterations.

Compare with FTVd4.0

In this experiment, we compare the proposed algorithm with FTVd4.0, in which the code is available online at http://www.caam.rice.edu/~optimization/L1/ftvd/v4.0/. For more details, see, e.g., Tao and Yang [104], and Li [69].

As we discussed in the introduction, the main difference between FTVd4.0 and the proposed method is the update scheme for ρ. For FTVd4.0, we use the default setting of ρ in FTVd 4.0, which is $\rho = 10$. For the proposed method, we set $\rho_0 = 2$, $\gamma = 2$, and $\alpha = 0.7$.

The updating strategy of ρ in the proposed algorithm makes a difference in convergence compared to FTVd4.0. Fig. 4.10 shows the history of PSNR and the relative



(a) PSNR (b) run time

Figure 4.9: Deblurring result of `cameraman.tif`, which is blurred by a Gaussian blur PSF of size 9×9, variance 5, and with BSNR = 40dB. Left: PSNR history using the proposed method and FTVd 3.0. Right: Run time as a function of size of the Gaussian blur PSF (`hsize`).

change $\|\mathbf{f}_{k+1} - \mathbf{f}_k\| / \|\mathbf{f}_k\|$ of a problem. In terms of PSNR, the proposed algorithm converges faster than FTVd4.0. In terms of relative change $\|\mathbf{f}_{k+1} - \mathbf{f}_k\| / \|\mathbf{f}_k\|$, the proposed algorithm achieves faster reduction compared to FTVd4.0, especially when the number of iterations becomes large.

Table 4.3 shows the PSNR, run time and number of iterations of 20 deblurring problems. It can be seen that at a low tolerance level ($\|\mathbf{f}_{k+1} - \mathbf{f}_k\| / \|\mathbf{f}_k\| \leq 10^{-6}$), FTVd 4.0 takes excessive number of iterations to converge.

4.8 Summary

In summary, we discussed the proposed algorithm for image restoration. Convergence properties, warm start properties and parameter selection methods have been discussed. Numerical results showed that the proposed method out-performs existing augmented Lagrangian methods and half-quadratic penalty methods. In next chapter, we extend the proposed method to video problems.

Table 4.2: PSNR values and run time for 20 images. Comparison between the proposed method, FTVd [112] and FastTV [57]. In all the tests, we used a Gaussian blur PSF with size 9×9, variance 5, and BSNR = 40dB. The stopping criteria is $\|\mathbf{f}_{k+1} - \mathbf{f}_k\| / \|\mathbf{f}_k\| < 10^{-3}$.

Image	Size		PSNR (dB)			Time (sec)		
Name	rows	cols	Propose	FTVd	FastTV	Propose	FTVd	FastTV
Barbara	512	512	25.33	25.31	25.00	1.02	9.53	84.50
Bicycles	512	768	26.77	26.75	21.34	1.91	14.42	128.97
Boat	512	512	31.53	31.40	31.01	0.96	8.72	84.52
Boat2	563	844	34.85	34.78	29.91	2.71	19.12	184.70
Boat3	570	856	38.11	38.00	34.04	2.33	16.25	168.11
Building1	588	883	36.34	36.01	32.41	2.25	17.89	172.02
Building2	513	644	28.20	28.18	23.03	1.54	15.66	102.40
Dog1	640	520	34.02	33.88	28.71	1.42	14.06	98.84
Fence	542	705	35.82	35.67	31.47	1.89	13.28	126.21
Fountain	573	717	34.30	34.09	29.25	2.05	23.46	159.64
House1	573	716	27.33	27.26	21.94	2.05	20.57	165.62
House2	576	864	27.63	27.62	22.20	2.31	20.95	152.18
Lady1	720	480	32.86	32.83	27.60	1.58	11.36	99.85
Lady2	775	517	36.34	36.30	31.64	1.97	14.80	129.65
Lena	512	512	35.05	34.99	34.57	0.96	7.86	83.39
Lighthouse1	404	606	28.75	28.71	23.43	1.16	8.92	86.72
Lighthouse2	720	480	31.78	31.71	26.43	1.47	11.13	99.79
Nature1	512	768	30.89	30.84	25.61	1.75	13.87	126.65
Show	507	692	29.69	29.55	24.33	1.51	17.69	122.90
Toy1	517	646	34.74	34.33	29.68	1.39	17.10	107.41

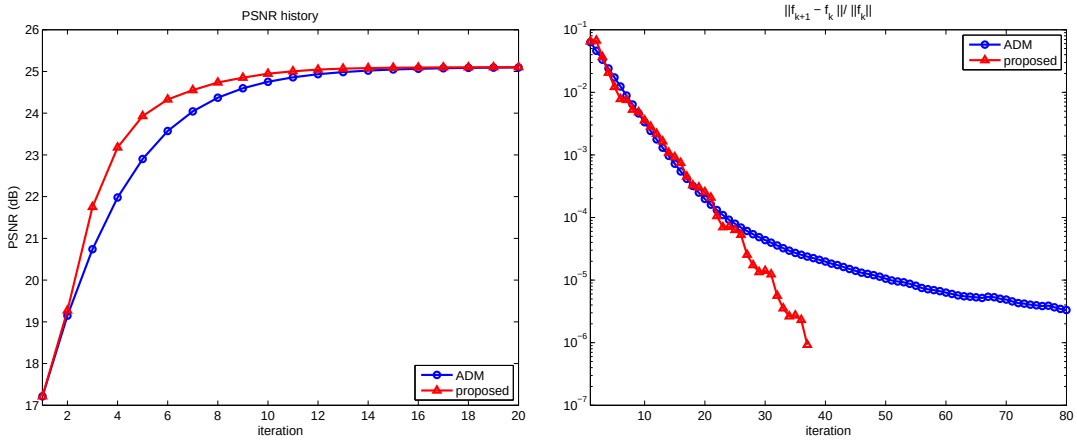


Figure 4.10: Comparison between FTVd4.0 with $\rho = 10$ and the proposed algorithm with $\rho_0 = 2$, $\gamma = 2$, $\alpha = 0.7$. Left: PSNR history of the proposed method and FTVd4.0. Right: the history of relative change.

4.9 Acknowledgment

This chapter, in part, is a reprint of the following paper

Stanley H. Chan, Ramsin Khoshabeh, Kristofor B. Gibson, Philip E. Gill and Truong Q. Nguyen, "An augmented Lagrangian method for total variation video restoration," *IEEE Transactions on Image Processing*, vol. 20, issue 11, pp.3097-3111, Nov 2011.

Table 4.3: Results of deblurring 20 images, comparing the proposed method and FTVd4.0 [104]. The stopping criteria is $\|\mathbf{f}_{k+1} - \mathbf{f}_k\| / \|\mathbf{f}_k\| < 10^{-6}$.

Image	Size		PSNR (dB)		Time (sec)		Iteration	
Name	rows	cols	proposed	FTVd4.0	proposed	FTVd4.0	proposed	FTVd4.0
Barbara	512	512	25.30	25.27	2.86	21.72	35	197
Bicycles	512	768	26.76	26.75	5.39	32.93	36	193
Boat	512	512	31.40	31.34	2.84	19.56	35	178
Boat2	563	844	34.69	34.55	7.64	54.84	36	229
Boat3	570	856	37.88	37.72	6.71	51.19	34	228
Building1	588	883	35.95	35.68	7.15	49.10	35	204
Building2	513	644	28.18	28.10	4.58	29.42	35	201
Dog1	640	520	33.90	33.70	4.33	31.96	36	228
Fence	542	705	35.54	35.39	5.48	31.61	35	180
Fountain	573	717	34.04	33.81	6.06	45.96	35	229
House1	573	716	27.25	27.19	6.19	40.17	36	206
House2	576	864	27.61	27.59	6.35	45.96	35	213
Lady1	720	480	32.80	32.71	4.51	35.42	36	243
Lady2	775	517	36.22	36.09	6.02	43.80	36	237
Lena	512	512	34.95	34.84	3.01	25.64	37	236
Lighthouse1	404	606	28.71	28.69	3.42	21.82	35	201
Lighthouse2	720	480	31.68	31.60	4.58	30.60	37	210
Nature1	512	768	30.83	30.73	5.21	32.64	35	191
Show	507	692	29.59	29.46	4.88	33.36	35	214
Toy1	517	646	34.44	34.08	4.78	34.75	34	221

Chapter 5

`deconvtv` for Video Restoration

In this chapter, we extend the proposed algorithm for video restoration problems. The key difference between image and video is the additional time dimension. Consequently, video restoration has some unique features that do not exist in image restoration:

1. Motion information

   Motion deblurring requires motion vector field, which can be estimated from a video sequence using conventional methods such as block-matching [111] and optical flow [72]. While it is also possible to remove motion blur based on a single image, for example, [27, 30, 60, 67, 97], the performance is limited to global motion or at most one to two objects by using sophisticated object segmentation algorithms.

2. Spatial variance versus spatial invariance

   For a class of spatially variant image restoration problems (in particular motion blur), the convolution matrix $\mathbf{H}$ is not a block-circulant matrix. Therefore, Fourier Transforms cannot be utilized to efficiently find a solution. Videos, in contrast, allow us to transform a sequence of spatially variant problems to a spatially *invariant* problem (See next section for more discussions). Consequently, huge gain in speed can be realized.

3. Temporal consistency

   Temporal consistency concerns about the smoothness of the restored video along the time axis. Although smoothing can be performed spatially (as in the case

of single image restoration), temporal consistency cannot be guaranteed if these methods are applied to a video in a frame-by-frame basis.

Because of these unique features in video, we seek a video restoration algorithm that utilizes motion information, exploits the spatially invariant properties and enforces spatial and temporal consistency.

5.1 Related Work

There are many works on the problem of video restoration, especially in the domain of video super-resolution. In the work of Farsiu, Elad and Milanfar [36,37,39], video super-resolution is formulated in a regularized least-squares minimization framework, in which the bilateral total variation is used as the regularization function. Later, Takeda and Milanfar [102, 103] applied the concept of kernel regression to the video restoration problem. Similar approaches can also be found in [81], where Ng et al. considered isotropic total variation as the regularization function and incorporated the geometric warp caused by motion. In [8], Belekos et al. proposed a novel prior that utilizes the motion vector field in updating the regularization parameters so that the prior is both spatially and temporally adaptive to the data. Recent work by Chan and Nguyen [20] considered a regularization function of the residue between the current solution and the motion compensated version of the previous solution.

It is worth noting that most of the above mentioned methods recover a video in a frame-by-frame basis[1]. Additionally, all of these methods assume that the blur kernel is spatially invariant. While this assumption is valid for many super-resolution scenarios where multiple shots of the same object are used to fuse a higher resolution image, it is invalid when the blur is caused by object motions. As a result, they are unable to handle the spatially variant motion blur kernel.

Our proposed algorithm is inspired by the concept of "space-time volume", which is first introduced in the early 90's by Jähne [59], and rediscovered by Wexler, Shechtman, Caspi and Irani [98, 113]. The idea of space-time volume is to stack the frames of a video to form a three-dimensional data structure known as the space-time volume. This allows one to transform the spatially variant motion blur problem to a spatially invariant

[1] A version of [8] is able to process multiple frames simultaneously, but in practice it only supports 5 frames at once.

problem. By imposing regularization functions along the spatial and temporal directions respectively, both spatial and temporal smoothness can be enforced.

The main drawback of space-time minimization is that the size of a space-time volume is much larger than that of a single image (or 5 frames in the case of [8]). Therefore, the authors of [98] only considered a Tikhonov regularized least-squares minimization (Equation (3) of [98]) in which a closed-form solution exists. More sophisticated regularization functions such as total variation and bilateral total variation do not seem possible under this framework, for these non-differentiable functions are difficult to solve efficiently.

The contribution of our work is summarized as follows:

- We extend the existing augmented Lagrangian method to solve space-time total variation minimization problems. Augmented Lagrangian method was previously used to image restoration only.

- Because of the space-time data structure, our proposed algorithm is able to handle spatially variant motion blur problems (object motion blur). Existing methods such as [8, 20, 36, 37, 39, 81, 102, 103] are unable to do so.

- Compared to [98] which is also a space-time minimization method, our method achieves TV/L1 and TV/L2 minimization quality whereas [98] only achieves least-squares minimization quality.

- In terms of speed, we achieve significantly faster computational speed compared to existing methods. Typical run time to deblur and denoise a 300×400 gray-scaled video is a few second per frame on a personal computer (MATLAB). This implies the possibility of real-time processing on GPU.

- The proposed algorithm supports a wide range of applications: (1). Video deblurring - With the assistance of frame rate up conversion algorithms, the proposed method can remove spatially variant motion blur for *real* video sequences. (2). Video disparity - Occlusion errors and temporal inconsistent estimates in the video disparity can be handled by the proposed algorithm without any modification. (3). Hot-air turbulence - The algorithm can be directly used to deblur and remove hot-air turbulence effects.

5.2 Three-dimensional Operators

A video signal is represented by a three-dimensional function $f(x, y, t)$, where (x, y) denotes the coordinate in space and t denotes the coordinate in time. Suppose that each frame of the video has M rows, N columns, and there are K frames, then the discrete samples of $f(x, y, t)$ for $x = 0, \ldots, M-1$, $y = 0, \ldots, N-1$, and $t = 0, \ldots, K-1$ form a three-dimensional tensor of size $M \times N \times K$.

Same as the image restoration case, we use the bold letter $\mathbf{f}$ to represent the vectorized version of the space-time volume $f(x, y, t)$, i.e., $\mathbf{f} = \mathbf{vec}(f(x, y, t))$.

5.2.1 Three-dimensional Convolution

The three-dimensional convolution is a natural extension of the conventional two-dimensional convolution. Given a space-time volume $f(x, y, t)$ and the blur kernel $h(x, y, t)$, the convolved signal $g(x, y, t)$ is given by $g(x, y, t) = f(x, y, t) * h(x, y, t) \stackrel{def}{=} \sum_{u,v,\tau} h(u, v, \tau) f(x - u, y - v, t - \tau)$. Using matrix-vector notations, we write

$$\mathbf{Hf} = \mathbf{vec}(g(x, y, t)) = \mathbf{vec}(h(x, y, t) * f(x, y, t)). \tag{5.1}$$

The (three-dimensional) convolution matrix $\mathbf{H}$ is a *triple* block-circulant matrix - it has a block circulant structure, and within each block there is a block-circulant-with-circulant block (BCCB) submatrix.

5.2.2 Forward Difference Operators

We define the three dimensional difference operator as

$$\mathbf{D} = [\mathbf{D}_x^T, \mathbf{D}_y^T, \mathbf{D}_t^T]^T,$$

where $\mathbf{D}_x$, $\mathbf{D}_y$ and $\mathbf{D}_t$ are the first-order forward finite difference operators along the horizontal, vertical and temporal directions, respectively. The definitions of each individual

sub-operators are

$$\mathbf{D}_x\mathbf{f} = \mathbf{vec}(f(x+1,y,t) - f(x,y,t)),$$
$$\mathbf{D}_y\mathbf{f} = \mathbf{vec}(f(x,y+1,t) - f(x,y,t)),$$
$$\mathbf{D}_t\mathbf{f} = \mathbf{vec}(f(x,y,t+1) - f(x,y,t)),$$

with periodic boundary conditions.

In order to have greater flexibility in controlling the forward difference along each direction, we introduce three scaling factors as follows. We define the scalars β_x, β_y and β_t and multiply them with $\mathbf{D}_x$, $\mathbf{D}_y$ and $\mathbf{D}_t$, respectively so that $\mathbf{D} = \begin{bmatrix}\beta_x\mathbf{D}_x^T & \beta_y\mathbf{D}_y^T & \beta_t\mathbf{D}_t^T\end{bmatrix}^T$.

With $(\beta_x, \beta_y, \beta_t)$, we define the anisotropic space-time total variation norm as

$$\|\mathbf{f}\|_{TV1} = \sum_i \left(\beta_x|[\mathbf{D}_x\mathbf{f}]_i| + \beta_y|[\mathbf{D}_y\mathbf{f}]_i| + \beta_t|[\mathbf{D}_t\mathbf{f}]_i|\right), \tag{5.2}$$

and the isotropic space-time total variation norm as

$$\|\mathbf{f}\|_{TV2} = \sum_i \sqrt{\beta_x^2[\mathbf{D}_x\mathbf{f}]_i^2 + \beta_y^2[\mathbf{D}_y\mathbf{f}]_i^2 + \beta_t^2[\mathbf{D}_t\mathbf{f}]_i^2}. \tag{5.3}$$

When $\beta_x = \beta_y = 1$ and $\beta_t = 0$, $\|\mathbf{f}\|_{TV2}$ is the two-dimensional total variation of $\mathbf{f}$ (in space). When $\beta_x = \beta_y = 0$ and $\beta_t = 1$, $\|\mathbf{f}\|_{TV2}$ is the one-dimensional total variation of $\mathbf{f}$ (in time). By adjusting β_x, β_y and β_t, we can control the relative emphasis put on individual terms $\mathbf{D}_x\mathbf{f}$, $\mathbf{D}_y\mathbf{f}$ and $\mathbf{D}_t\mathbf{f}$.

5.3 Proposed Algorithm

The proposed video restoration algorithm is a direct extension of the image restoration algorithm. Therefore, instead of repeating the details, we focus on the modifications made to the three-dimensional data structure. Additionally, our discussion is focused on the anisotropic TV/L2 problem. The isotropic TV/L2 problems and TV/L1 problems can be derived similarly.

5.3.1 Algorithm

The core optimization problem that we solve is the following TV/L2 minimization:

$$\underset{\mathbf{f}}{\text{minimize}} \quad \tfrac{\mu}{2} \left\|\mathbf{Hf} - \mathbf{g}\right\|^2 + \left\|\mathbf{Df}\right\|_1, \tag{5.4}$$

where μ is a regularization parameter. Since it has the same form as the image TV/L2 problem, the algorithm is essentially the same as the image case. However, in the video restoration setting, the intermediate variable $\mathbf{u}$ and the Lagrange multiplier $\mathbf{y}$ consist of three terms:

$$\mathbf{u} = \begin{bmatrix} \mathbf{u}_x^T & \mathbf{u}_y^T & \mathbf{u}_t^T \end{bmatrix}^T, \quad \text{and} \quad \mathbf{y} = \begin{bmatrix} \mathbf{y}_x^T & \mathbf{y}_y^T & \mathbf{y}_t^T \end{bmatrix}^T. \tag{5.5}$$

Therefore, the solution of the $\mathbf{f}$-subproblem is

$$\mathbf{f} = \mathcal{F}^{-1}\left[\frac{\mathcal{F}[\mu\mathbf{H}^T\mathbf{g} + \rho\mathbf{D}^T\mathbf{u} - \mathbf{D}^T\mathbf{y}]}{\mu|\mathcal{F}[\mathbf{H}]|^2 + \rho(|\mathcal{F}[\mathbf{D}_x]|^2 + |\mathcal{F}[\mathbf{D}_y]|^2 + |\mathcal{F}[\mathbf{D}_t]|^2)}\right], \tag{5.6}$$

where $\mathcal{F}$ is the three-dimensional Fourier Transform operator. In practice, $\mathcal{F}$ is implemented using three-dimensional Fast Fourier Transform (FFT3).

The solution of the $\mathbf{u}$-subproblem is

$$\begin{aligned} \mathbf{u}_x &= \max\left\{|\mathbf{v}_x| - \frac{1}{\rho_r}, 0\right\} \cdot \operatorname{sign}(\mathbf{v}_x), \\ \mathbf{u}_y &= \max\left\{|\mathbf{v}_y| - \frac{1}{\rho_r}, 0\right\} \cdot \operatorname{sign}(\mathbf{v}_y), \\ \mathbf{u}_t &= \max\left\{|\mathbf{v}_t| - \frac{1}{\rho_r}, 0\right\} \cdot \operatorname{sign}(\mathbf{v}_t). \end{aligned} \tag{5.7}$$

In case of isotropic TV, the $\mathbf{u}$-subproblem solution is

$$\begin{aligned} \mathbf{u}_x &= \max\left\{\mathbf{v} - \frac{1}{\rho_r}, 0\right\} \cdot \frac{\mathbf{v}_x}{\mathbf{v}}, \\ \mathbf{u}_y &= \max\left\{\mathbf{v} - \frac{1}{\rho_r}, 0\right\} \cdot \frac{\mathbf{v}_y}{\mathbf{v}}, \\ \mathbf{u}_t &= \max\left\{\mathbf{v} - \frac{1}{\rho_r}, 0\right\} \cdot \frac{\mathbf{v}_t}{\mathbf{v}}, \end{aligned} \tag{5.8}$$

where $\mathbf{v}_x = \beta_x\mathbf{D}_x\mathbf{f} + \frac{1}{\rho_r}\mathbf{y}_x$ (similar definitions for $\mathbf{v}_y$ and $\mathbf{v}_t$),

$$\mathbf{v} = \sqrt{|\mathbf{v}_x|^2 + |\mathbf{v}_y|^2 + |\mathbf{v}_t|^2 + \epsilon}$$

and ϵ is a small constant ($\epsilon = 10^{-8}$). Here the multiplication and divisions are component-wise operations.

5.3.2 Comparison with Other Methods

The proposed algorithm belongs to the class of operator splitting methods. Table 5.1 summarizes the differences between the proposed video restoration method and some existing methods. The speed comparison is based on deblurring "lena.bmp" (512×512, gray scaled), which is blurred by a Gaussian blur kernel of size 9×9, $\sigma = 5$, BSNR = 40dB. The machine used is Intel Qual Core 2.8GHz, 4GB RAM, Windows 7/ MATLAB 2010. Comparisons between FTVd 4.0 and the proposed method are based on $\rho_r = 2$. If $\rho_r = 10$ (default setting of FTVd 4.0), then the run time are 1.56 sec and 1.28 sec for FTVd 4.0 and the proposed method, respectively.

Table 5.1: Comparisons between proposed and other methods

	Fast-TV [57]	FTVd 3.0 [112]	FTVd 4.0 [104] Split Bregman [48] Constrained TV [2]	Proposed
Principle	Half quadratic	Half quadratic	Operator Splitting	Operator Splitting
Data	Gray-scale image	Gray-scale image Color image	Gray-scale image Color image	Gray-scale image Color image Video
Regularization	Spatial TV	Spatial TV	Spatial TV	Spatial-Temporal TV
Penalty Parameter	$\rho_r \to \infty$	$\rho_r \to \infty$	constant ρ_r	Update ρ_r based on constraint violation
Speed	83.39 sec	7.86 sec	2.94 sec	1.79 sec

5.4 Application 1: Video Deblurring

In the following sections we demonstrate three applications of the proposed algorithm, namely (1) video deblurring, (2) video disparity refinement, and (3) video restoration for videos distorted by hot-air turbulence.

5.4.1 Spatially Invariant Blur

We first consider the class of spatially invariant blur. In this problem, the t-th observed image $g(x, y, t)$ is related to the true image $f(x, y, t)$ as

$$g(x, y, t) = h(x, y) * f(x, y, t) + \eta(x, y, t).$$

Note that the spatially invariant blur kernel $h(x, y)$ is assumed to be identical for all time t.

The typical method to solve a spatially invariant blur is to consider the model

$$\mathbf{g}_k = \mathbf{H}\mathbf{f}_k + \eta,$$

and apply a frame-by-frame approach to recover $\mathbf{f}_k$ individually. In [20], the authors considered the following minimization

$$\underset{\mathbf{f}_k}{\text{minimize}} \|\mathbf{H}\mathbf{f}_k - \mathbf{g}_k\|^2 + \lambda_S \sum_i \|\mathbf{D}_i\mathbf{f}_k\|_1 + \lambda_T \|\mathbf{f}_k - \mathbf{M}_k\hat{\mathbf{f}}_{k-1}\|^2,$$

where $\hat{\mathbf{f}}_{k-1}$ is the solution of the $k-1$-th frame and $\mathbf{M}_k$ is the motion compensation operator that maps the coordinates of $\mathbf{f}_{k-1}$ to the coordinates of $\mathbf{f}_k$. The operators $\mathbf{D}_i$ are the spatial forward finite difference operators oriented at angles 0°, 45°, 90° and 135°. The regularization parameters λ_S and λ_T control the relative emphasis put on the spatial and temporal smoothness.

Another method to solve the spatially invariant blur problem is to apply the multichannel approach by modeling the imaging process as [8, 81]

$$\mathbf{g}_i = \mathbf{H}\mathbf{M}_{i,k}\mathbf{f}_k + \eta,$$

for $i = k-m, \ldots, k, \ldots, k+m$, where m is the size of the temporal window (typically ranged from 1 to 3). $\mathbf{M}_{i,k}$ is the motion compensation operator that maps the coordinates of $\mathbf{f}_k$ to the coordinates of $\mathbf{g}_i$. The k-th frame can be recovered by solving the following minimization [81]

$$\underset{\mathbf{f}_k}{\text{minimize}} \sum_{i=k-m}^{k+m} a_i \|\mathbf{H}\mathbf{M}_{i,k}\mathbf{f}_k - \mathbf{g}_i\|^2 + \lambda \|\mathbf{f}_k\|_{TV2}, \tag{5.9}$$

where a_i is a constant and $\|\mathbf{f}_k\|_{TV2}$ is the isotropic total variation on the k-th frame. The method presented in [8] replaces the objective function by a weighted least-squares and the isotropic total variation regularization function by a weighted two-norm on gradient. The weights are adaptively updated (using residue and motion vector field) in each iteration, and so the regularization function is non-stationary both spatially and

temporally.

A drawback of these methods is that the image recovery result depends heavily on the accuracy of motion estimation and compensation. Especially in occlusion areas, the assumption that $\mathbf{M}_{i,k}$ is a one-to-one mapping [28] fails to hold. Thus, $\mathbf{M}_{i,k}$ is not a full rank matrix and $\mathbf{M}_{i,k}^T\mathbf{M}_{i,k} \neq \mathbf{I}$. As a result, minimizing $\|\mathbf{H}\mathbf{M}_{i,k}\mathbf{f}_k - \mathbf{g}_i\|^2$ can lead to serious error. There are methods to reduce the error caused by rank deficiency of $\mathbf{M}_{i,k}$, for example the concept of *unobservable pixel* introduced in [81], but the restoration result depends on the effectiveness of how the unobservable pixels are selected.

Another drawback of these methods is the computation time. For spatially invariant blur, the blur operator $\mathbf{H}$ is a block circulant matrix. However, in the multichannel model, the operator $\mathbf{H}\mathbf{M}_{i,k}$ is not a block circulant matrix. The block-circulant property is a critical factor to speed as it allows the use of Fourier Transform methods. For methods in [8,81], conjugate gradient (CG) is used to solve the minimization task. While the total number of CG iterations may be few, the per iteration run time can be long.

Table 5.2: Comparisons between video restoration methods

	Belekos 2010 [8]	Ng 2007 [81]	Chan 2011 [20]
Approach	multi-frames to multi-frames	frame-by-frame	frame-by-frame
Spatial Consistency	$\sum_i \sum_{d\in\{x,y\}} (\mathbf{D}_d\mathbf{f})^T\mathbf{A}_i^d(\mathbf{D}_d\mathbf{f})$	$\sum_i \sqrt{[\mathbf{D}_x\mathbf{f}]_i^2 + [\mathbf{D}_y\mathbf{f}]_i^2}$	$\sum_i \|\mathbf{D}_i\mathbf{f}\|_1$
Temporal Consistency	$\sum_{i,j} \|\mathbf{f}_i - \mathbf{M}_{ij}\mathbf{f}_j\|^2_{\mathbf{B}_{ij}}$	$\|\mathbf{H}\mathbf{M}_{ik}\mathbf{f}_k - \mathbf{g}_i\|^2$	$\|\mathbf{f}_k - \mathbf{M}_k\hat{\mathbf{f}}_{k-1}\|^2$
Motion Compensation	Required	Required	Required
Handle of Motion Blur	spatially variant operator	spatially variant operator	spatially variant operator
Objective Function	weighted least-squares	TV/L2	TV/L2 + quadratic penalty
Solver	Conjugate gradient	Conjugate gradient	Sub-gradient Proj.

	Shechtman 2005 [98]	Proposed
Approach	space-time volume	space-time volume
Spatial Consistency	$\|\mathbf{D}_x\mathbf{f}\|^2 + \|\mathbf{D}_y\mathbf{f}\|^2$	$\|\mathbf{f}\|_{TV2}$
Temporal Consistency	$\|\mathbf{D}_t\mathbf{f}\|^2$	$\|\mathbf{f}\|_{TV2}$, Equation (5.3)
Motion Compensation	Not Required	Not Required
Handle of Motion Blur	3D-FFT	3D-FFT
Objective Function	Tikhonov	TV/L2 or TV/L1
Solver	Closed-form	Closed-form + Shrinkage

Our approach to solve spatially invariant blur problem shares the same insight as [98] which does *not* consider motion compensation. The temporal error is handled by the spatio-temporal total variation $\|\mathbf{f}\|_{TV2} = \sum_i \sqrt{|[\mathbf{D}_x\mathbf{f}]_i|^2 + |[\mathbf{D}_y\mathbf{f}]_i|^2 + |[\mathbf{D}_t\mathbf{f}]_i|^2}$. An intuition to this approach is that the temporal difference $\mathbf{f}_k - \mathbf{f}_{k-1}$ can be classified as temporal *edge* and temporal *noise*. The temporal edge is the intensity change

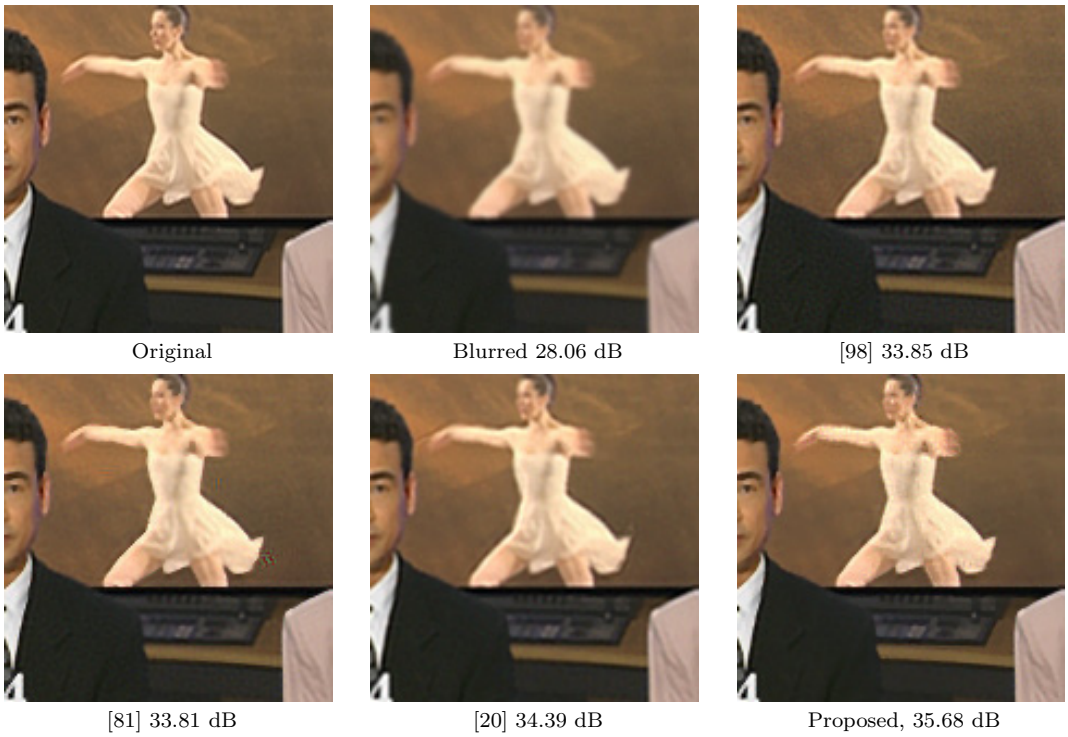


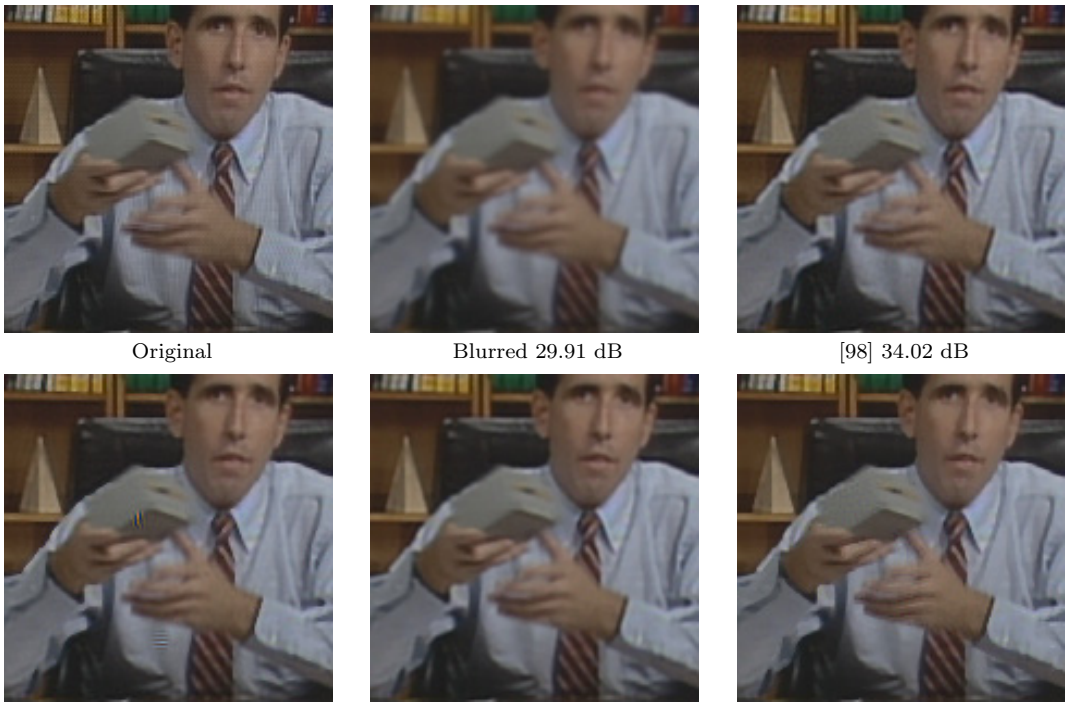
Figure 5.1: "News" sequence, frame no. 100. (a) Original image (cropped for better visualization). (b) Blurred by a Gaussian blur kernel of size 9×9, $\sigma = 1$, BSNR = 30dB. (c)-(f) Results by various methods. (Table 5.3).

caused by object movements, whereas the temporal noise is the artifact generated in the minimization process. Similar to the spatial total variation, the temporal total variation preserves the temporal edges while reducing the temporal noise. Moreover, the space-time volume preserves the block circulant structure of the operator, thus leading to significantly faster computation. Table 5.2 illustrates the differences between various video restoration methods.

Table 5.3 and Fig. 5.1 show the comparisons between [98], [81], [20] and the proposed method on spatially invariant blur. The four testing video sequences are blurred by a Gaussian blur kernel of size 9×9 with $\sigma = 1$. Additive Gaussian noise is added so that the blurred signal to noise ratio (BSNR) is 30dB.

The specific settings of the methods are as follows. For [98], we consider the minimization

$$\underset{\mathbf{f}}{\text{minimize}} \ \mu\|\mathbf{Hf} - \mathbf{g}\|^2 + \beta_x^2\|\mathbf{D}_x\mathbf{f}\|^2 + \beta_y^2\|\mathbf{D}_y\mathbf{f}\|^2 + \beta_t^2\|\mathbf{D}_t\mathbf{f}\|^2$$



Original | Blurred 29.91 dB | [98] 34.02 dB

[81] 33.87 dB | [20] 33.88 dB | Proposed, 34.74 dB

Figure 5.2: "Salesman" sequence, frame no. 10. (a) Original image (cropped for better visualization). (b) Blurred by a Gaussian blur kernel of size 9×9, $\sigma = 1$, BSNR = 30dB. (c)-(f) Results by various methods. (Table 5.3).

and set the parameters empirically for the best recovery quality: $\mu = 200$, $(\beta_x, \beta_y, \beta_t) = (1, 1, 1.25)$. For [81], instead of using the CG presented in the paper, we use a modification of the proposed augmented Lagrangian method to speed up the computation. Specifically, in solving the $\mathbf{f}$-subproblem we used conjugate gradient (LSQR [86]) to accommodate the non-block-circulant operator $\mathbf{HM}_{i,k}$. The motion estimation is performed using the benchmark full search (exhaustive search) with 0.5 pixel accuracy. The block size is 8×8 and the search range is 16×16. Motion compensation is performed by coordinate transform according to the motion vectors (bilinear interpolation for half pixels). The threshold for unobservable pixels [81] is set as 6 (out of 255), and the regularization parameter is $\lambda = 0.001$ (See Equation (5.9)). We use the previous and the next frame for the model, i.e. $m = 1$ and let $(a_{k-1}, a_k, a_{k+1}) = (0.5, 1, 0.5)$ (Using (1, 1, 1) tends to give worse results). For [20], the regularization parameters are also chosen empirically for the best recovery quality: $\lambda_S = 0.001$ and $\lambda_T = 0.05$.

To compare to these methods, we apply TV/L2 (Algorithm 1) with the following

Table 5.3: PSNR, E_S and E_T values for four video sequences blurred by Gaussian blur kernel 9×9, $\sigma = 1$, $BSNR = 30dB$.

		"Foreman"	"Salesman"	"Mother"	"News"
PSNR (dB)	Blurred	28.6197	29.9176	32.5705	28.1106
	[98]	31.6675	33.0171	36.1493	34.0113
	[81]	32.5500	33.8408	38.2164	34.1207
	[20]	33.2154	33.8618	39.6991	34.7133
	Proposed	**33.7864**	**34.7368**	**40.0745**	**35.8813**
E_S ($\times 10^4$)	[98]	1.2067	1.1706	0.82665	1.3764
	[81]	1.1018	1.0743	0.71751	1.2146
	[20]	**1.0076**	**0.9934**	0.61544	1.123
	Proposed	1.0930	1.0105	**0.61412**	**1.1001**
E_T ($\times 10^3$)	[98]	10.954	3.3195	3.7494	4.6484
	[81]	10.827	2.4168	2.9397	3.7503
	[20]	10.202	2.5471	2.7793	3.3623
	Proposed	**9.3400**	**1.9948**	**2.0511**	**2.6165**

parameters (same for all four videos): $\mu = 2000$, $(\beta_x, \beta_y, \beta_t) = (1, 1, 1)$. All other parameters take the default setting: $\alpha = 0.7$, $\gamma = 2$, $\rho_r = 2$. The algorithm terminates if $\|\mathbf{f}_k - \mathbf{f}_{k-1}\|/\|\mathbf{f}_{k-1}\| \leq 10^{-3}$.

In Table 5.3, three quantities are used to evaluate the performance of the algorithms. Peak signal to noise ratio (PSNR) measures the image fidelity. The spatial total variation E_S is defined as $E_S = \sum_i \sqrt{|[\mathbf{D}_x\mathbf{f}]_i|^2 + |[\mathbf{D}_y\mathbf{f}]_i|^2}$ for each frame and the temporal total variation E_T is defined as $E_T = \sum_i |[\mathbf{D}_t\mathbf{f}]_i|$ for each frame [20]. The average (over all frames) PSNR, E_S and E_T are listed in Table 5.3.

Referring to the results, it can be observed that the proposed algorithm produces the highest PSNR values while keeping E_S and E_T at a low level. It is worth noting that [98] is equivalent to the three-dimensional Wiener deconvolution (regularized). Therefore, there exists a closed form solution but the result looks more blurry than the other methods. Among the four methods, both [81] and [20] use motion estimation and compensation. However, [81] is more sensitive to the motion estimation error - motion estimation error in some fast moving areas are amplified in the deblurring step. [20] is more robust to motion estimation error, but the computation time is significantly longer than the proposed method. The run time of [81] and [20] are approximately 100 seconds per frame (per color channel) whereas the proposed algorithm only requires approximately 2 seconds per frame (per color channel). These statistics are based on recovering videos of size 288×352, using a PC with Intel Qual Core 2.8 GHz, 4GB RAM, Windows 7/ MATLAB 2010.

Figure 5.3: "Market Place" sequence, frame no. 146. Top: The original observed video sequences. Middle: Result of [98]. Bottom: Result of the proposed method.

5.4.2 Spatially Variant Motion Blur

The proposed algorithm can be used to remove spatially-variant motion blur. However, since motion blurred videos often have low temporal resolution, frame rate up conversion algorithms are needed to first increase the temporal resolution before applying the proposed method (See [98] for detailed explanations). To this end, we apply [66] to upsample the video by a factor of 8. Consequently, the motion blur kernel can be modeled as

$$h(x,y,t) = \begin{cases} 1/T, & \text{if } x = y = 0, \text{ and } 0 \leq t \leq T, \\ 0, & \text{otherwise,} \end{cases}$$

Figure 5.4: "Super Loop" sequence, frame no. 28. Top: The original observed video sequences. Middle: Result of [98]. Bottom: Result of the proposed method.

where $T = 8$ in this case.

Fig. 5.3 shows frame no. 146 of the video sequence "Market Place", and Fig. 5.4 shows frame no. 28 of the video sequence "Super Loop". The videos are captured by a Panasonic TM-700 video recorder with resolution 1920×1080p at 60 fps. For computational speed we down-sampled the spatial resolution by a factor of 4 (so the resolution is 480×270). The parameters of the proposed algorithm are chosen empirically as $\mu = 1000$, $(\beta_x, \beta_y, \beta_t) = [1, 1, 5]$. There are not many relevant video motion deblurring algorithms for comparison (or unavailable to be tested). Therefore, we are only able to show the results of [98], using parameters $\mu = 1000$, $(\beta_x, \beta_y, \beta_t) = [1, 1, 2.5]$.

As shown in Fig. 5.3 and Fig. 5.4, the proposed algorithm produces result with

much higher quality comparing to the result obtained from [98]. We also tested for a range of parameters μ and β's for [98]. However, we observe that the results are either over-sharpened (serious ringing artifacts), or under-sharpened (not enough deblurring).

5.4.3 Limitations

The proposed algorithm requires considerably less memory than other total variation minimization algorithms such as interior point methods. However, for high definition (HD) videos, the proposed algorithm still has memory issue as the size of the space-time volume is large. While one can use fewer frames to lower the memory demand, trade off in the recovery quality should be expected.

Another issue of the proposed algorithm is the sensitivity to the frame-rate conversion algorithm. At object boundaries where the motion estimation algorithm fails to provide accurate estimates, the estimation error in the deblurring step will be amplified. This typically occurs in areas with non-uniform and rapid motion.

5.5 Application 2: Video Disparity Refinement

5.5.1 Problem Description

Our second example is disparity map refinement. Disparity is proportional to the reciprocal of the distance between the camera and the object (i.e., depth). Disparity maps are useful for many stereo video processing applications, including object detection in three-dimensional space, saliency for stereo videos, stereo coding and view synthesis etc.

There are numerous papers on generating one disparity map based on a pair of stereo images [1]. However, all of these methods cannot be extended to videos because the energy functions are considered in a frame-by-frame basis. Although there are works in enforcing temporal consistency for adjacent frames, such as [84] and [35], the computational complexity is high.

We propose to estimate the video disparity in two steps. In the first step, we combine the locally adaptive support weight [118] and the dual cross bilateral grid [90] to generate an initial disparity estimate. Since this method is a frame-by-frame method, spatial and temporal consistency is poor. In the second step, we consider the initial

video disparity as a space-time volume and solve the TV/L1 minimization problem

$$\underset{\mathbf{f}}{\text{minimize}}\ \mu\|\mathbf{f}-\mathbf{g}\|_1 + \|\mathbf{f}\|_{TV2}.$$

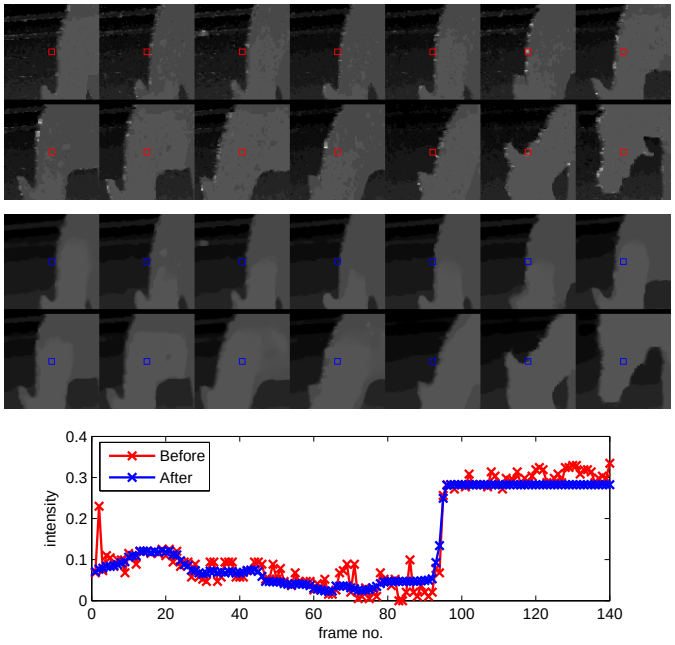


Figure 5.5: Top: Before applying the proposed TV/L1 algorithm; Middle: After applying the proposed TV/L1 algorithm. Bottom: Trace of a pixel along the time axis.

There are two reasons for choosing TV/L1 instead of TV/L2 in refining video disparity. First, disparity is a piece-wise constant function with quantized levels, and across the flat regions there are sharp edges. As shown in Fig. 5.5 (bottom), the estimation error behaves like outliers in a smooth function. Therefore, to reduce the estimation error, one can consider a robust curve fitting as it preserves the shape of the data while suppressing the outliers.

The second reason for using TV/L1 is that the one-norm $\|\mathbf{f}-\mathbf{g}\|_1$ is related to the notion of percentage of bad pixels, a quantity commonly used to evaluate disparity estimation algorithms [1]. Given a ground truth disparity $\mathbf{f}^*$, the number of bad pixels of an estimated disparity $\mathbf{f}$ is the cardinality of the set $\{i|\ |[\mathbf{f}-\mathbf{f}^*]_i| > \tau\}$ for some threshold τ. In the absence of ground truth, the same idea can be used with a reference disparity (e.g., $\mathbf{g}$). In this case, the cardinality of the set $\Omega_\tau = \{i|\ |[\mathbf{f}-\mathbf{g}]_i| > \tau\}$, denoted by $|\Omega_\tau|$, is the number of bad pixels of $\mathbf{f}$ with respect to (w.r.t) $\mathbf{g}$. Therefore, minimizing $|\Omega_\tau|$ is

equivalent to minimizing the number of bad pixels of $\mathbf{f}$ w.r.t. $\mathbf{g}$. However, this problem is non-convex and is NP-hard. In order to alleviate the computational difficulty, we set $\tau = 0$ so that $|\Omega_\tau| = \|\mathbf{f} - \mathbf{g}\|_0$, and convexify $\|\mathbf{f} - \mathbf{g}\|_0$ by $\|\mathbf{f} - \mathbf{g}\|_1$. Therefore, $\|\mathbf{f} - \mathbf{g}\|_1$ can be regarded as the convexification of the notion of percentage bad pixels.

5.5.2 Results - Video

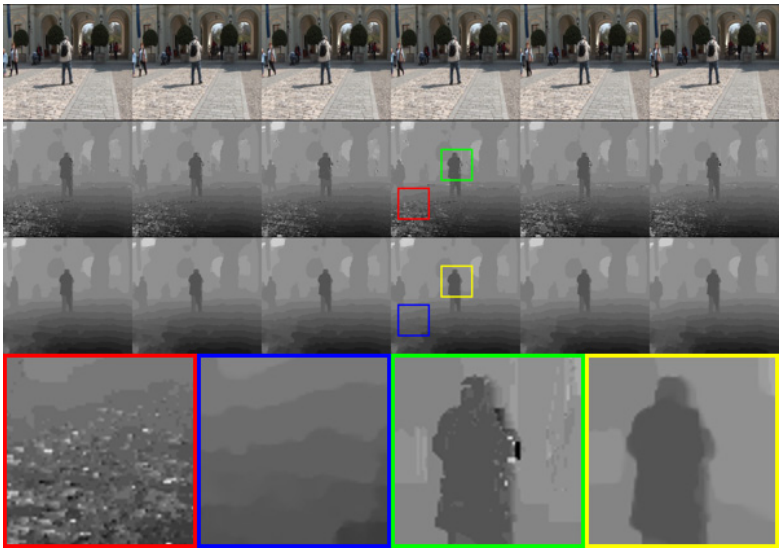


Figure 5.6: Video disparity estimation for "Old Timers" sequence. First row: Left view of the stereo video. Second row: Initial disparity estimate. Third row: Refinement using the proposed method with parameters $\mu = 0.75$, $(\beta_x, \beta_y, \beta_t) = (1, 1, 2.5)$, $\alpha = 0.7$, $\rho_r = 2$, $\rho_o = 100$, $\gamma = 2$. Last row: Zoom-in comparisons.

Two real videos ("Horse" and "Old Timers") are tested for the proposed algorithm. These stereo videos are downloaded from the following website:

http://sp.cs.tut.fi/mobile3dtv/stereo-video/

Fig. 5.6 illustrates the results. The first row of Fig. 5.7 shows the left view of the stereo video. The second row shows the results of applying [90, 118] to the stereo video. Note that we are implementing a spatio-temporal version of [90], which uses adjacent frames to enhance the temporal consistency. However, the estimated disparity is still noisy, especially around the object boundaries. The third row shows the result of applying the proposed TV/L1 minimization to the initial disparity estimated in the second row. It should be noted that the proposed TV/L1 minimization improves not only the flat

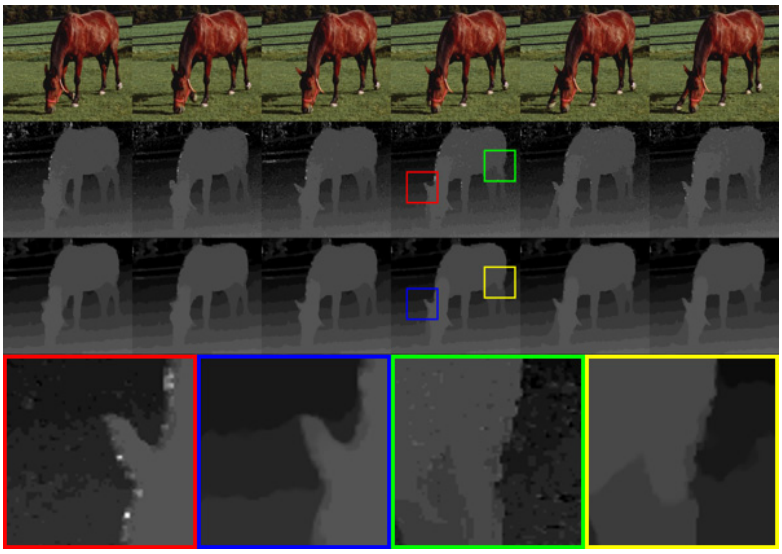
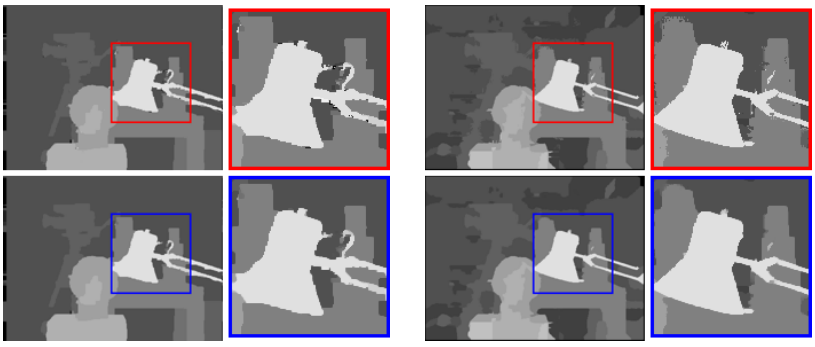


Figure 5.7: Video disparity estimation for "Horse" sequence. First row: Left view of the stereo video. Second row: Initial disparity estimate. Third row: Refinement using the proposed method with parameters $\mu = 0.75$, $(\beta_x, \beta_y, \beta_t) = (1, 1, 2.5)$, $\alpha = 0.7$, $\rho_r = 2$, $\rho_o = 100$, $\gamma = 2$. Last row: Zoom-in comparisons.

interior region, but also the object boundary (e.g. the arm of the man in "Old Timers" sequence), an area that [90, 118] are unable to handle.

5.5.3 Results - Image

The effectiveness of the proposed algorithm can further be elaborated by comparing to the 99 benchmark methods on Middlebury stereo evaluation website [1]. For all 99 methods on Middlebury stereo evaluation website, we download their results and apply the proposed algorithm to improve the spatial smoothness. Note that the proposed algorithm is readily for this test because an image is a single frame video. In this case, we set $(\beta_x, \beta_y, \beta_t) = (1,\ 1,\ 0)$. Fig. 5.8 show the results for two of the 99 methods (randomly chosen) for the dataset "Tsukuba", and Fig. 5.9 shows the percentage of error reduction (in terms of number of bad pixels, with threshold 1) by applying the proposed algorithm to all methods on the Middlebury database. The higher bars in the plots indicate that the proposed algorithm reduces the error by a greater amount. It can be observed that the errors are typically reduced by a large margin of over 10%. While there is less error reduction for some datasets, it is important to note that error reduction is *always non-negative*. In other words, the proposed algorithm always improves the initial disparity



Algorithm no. 8 Algorithm no. 78

Figure 5.8: Image disparity refinement on algorithms no. 8 and 78 (randomly chosen) from Middlebury for "Tsukuba". Red box: Before applying the proposed method; Blue box: After applying the proposed method. $\mu \in [0.1,\ 1]$ is found exhaustively with increment 0.1, $(\beta_x, \beta_y, \beta_t) = (1, 1, 0)$, $\alpha = 0.7$, $\rho_r = 2$, $\rho_o = 100$, $\gamma = 2$.

estimate. Furthermore, for *every* algorithm, we provide improvement in at least one of the image sets.

5.5.4 Limitations

A limitation of the proposed algorithm is that it is unable to handle large and consistent error results from poor initial disparity estimation algorithm. This happens especially in large occlusion areas, repeating texture regions, or frames consisting of rapid motions. We are currently seeking methods to feedback the TV/L1 result to the initial disparity estimation so that the algorithm is more robust to these errors.

5.6 Application 3: Videos Distorted by Hot-Air Turbulence

5.6.1 Problem Description

Our third example is the stabilization of videos distorted by hot-air turbulence effects. In the presence of hot-air turbulence, the refractive index along the transmission path of the light ray is spatially and temporally varying [93]. Consequently, the path differences and hence the phases of the light rays are also spatially and temporally varying. As a result, the observed image is distorted by geometric warping, motion blur and sometimes out-of-focus blur. This type of distortion is generally known as the hot-air

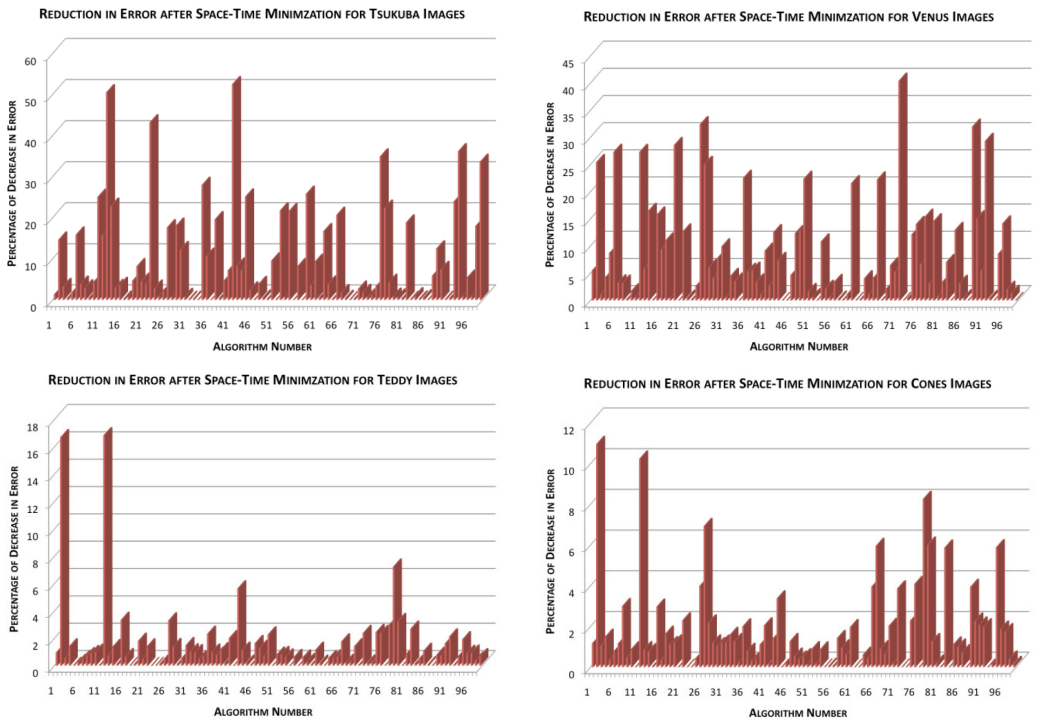


Figure 5.9: Percentage error reduction (in terms of number of bad pixels) by applying the proposed algorithm to all 99 methods on the Middlebury stereo database.

turbulence effect.

There are various methods to overcome imaging through hot-air turbulence. For example, the speckle imaging technique [93] assumes that the refractive index is changing randomly but is also statistically stationary [50, 51]. Consequently, by averaging enough number of frames, the geometric distortion will be smoothed out. Then a deconvolution algorithm can be used to remove the blur.

The drawback of the speckle imaging technique is that the average operation makes the deblurring process challenging. Therefore, Zhu and Milanfar [119], Shimizu et. al. [99] proposed to first compensate the geometric distortion using non-rigid registration [101], and then deblur the images using deconvolution algorithms. The limitation is that non-rigid registration works well only when the geometric distortion can be adjusted by all the control points in the grid [101]. However, imaging through hot-air turbulence contains both large area distortion (perceived as waving) and small disturbance (perceived as jittering). If non-rigid registration has to be used to compensate small disturbance, then the number of control points will be huge, making the computation not

(a) input (b) step 1 [24,25] (c) step 2: our method.

Figure 5.10: Hot-air turbulence removal for the sequence "Acoustic Explorer" - using the proposed method to reduce the effect of hot-air turbulence. (a) A frame of the original video sequence. (b) Step 1: Apply gray level grouping [24,25] to the input. (c) Step 2: Apply the proposed method to the results of Step 1.

practical. There are other methods such as lucky frame/region fusion approach [5,42]. However, these methods cannot handle small disturbance effectively either.

Using the same methodology as we used for video deblurring, we consider the video as a space-time volume and minimize the TV/L2 problem. Our intuition is that the small hot-air turbulence can be regarded as temporal noise whereas the object movement is regarded as temporal edge. Under this framework, spatially invariant blur can also be incorporated. If the input video originally has a low contrast, a preprocessing step using gray level grouping (GLG) [24,25] can be used (See Fig. 5.10).

5.6.2 Results

Fig. 5.11 shows the snapshots (zoom-in) of a video sequence "Acoustic Explorer". In this example, gray level grouping is applied to the input video so that contrast is enhanced. Then the proposed algorithm is used to reduce the hot-air turbulence effect. A Gaussian blur kernel is assumed in both examples, where the variance is determined empirically. Comparing the video quality before and after applying the proposed method, fewer jittering like artifacts are observed in the processed videos. While this may not be apparent by viewing the still images, the improvement is significant in the 24fps videos[2].

Fig. 5.12 shows the comparisons without the contrast enhancement by GLG. Referring to the figures, the proposed algorithm does not only reduce the unstable hot-air turbulence effects, it also improves the blur. The word "Empire State" could not be

[2]Videos are available at http://videoprocessing.ucsd.edu/~stanleychan/deconvtv

Figure 5.11: Zoom-in of "Acoustic Explorer" sequence frame no. 25-28 (object is 2 miles from camera). Top: input video sequence with contrast enhanced by gray level grouping (GLG). Bottom: Processed video by applying the proposed method to the output of GLG.

seen clearly in the input sequence, but becomes sharper in the processed sequence.

5.6.3 Limitations

The experiments above indicate that the proposed algorithm is effective for reducing small hot-air turbulence effects. However, for large area geometric distortions, non-rigid registration is needed. In addition, the general turbulence distortion is spatially and temporally varying, meaning that the point spread function cannot be modeled as one Gaussian function. This issue is an open problem.

5.7 Summary

In this chapter, we showed a video deblurring/denoising algorithm for spatial-temporal data. The algorithm uses an augmented Lagrangian method to solve the optimization problem. With the introduction of spatial and temporal regularization to the spatial-temporal data, the solution of the algorithm is both spatially and temporally consistent.

Applications of the algorithm include video deblurring, disparity refinement and turbulence removal. For video deblurring, the proposed algorithm restores motion-blurred video sequences. The average PSNR is improved, and the spatial and temporal

Figure 5.12: Snapshot of "Empire State" sequence. Left: input video sequence without GLG. Right: Processed video by applying GLG and the proposed method.

total variation are maintained at an appropriate level, meaning that the restored videos are spatially and temporally consistent. For disparity map refinement, the algorithm removes flickering in the disparity map, and preserves the sharp edges in the disparity map. For turbulence removal, the proposed algorithm stabilizes and deblurs videos taken under the influence of hot air turbulence.

5.8 Acknowledgment

This chapter, in part, is a reprint of the following papers

Stanley H. Chan, Ramsin Khoshabeh, Kristofor B. Gibson, Philip E. Gill and Truong Q. Nguyen, "An augmented Lagrangian method for total variation video restoration," *IEEE Transactions on Image Processing*, vol. 20, issue 11, pp.3097-3111, Nov 2011.

Stanley H. Chan, Ramsin Khoshabeh, Kristofor B. Gibson, Philip E. Gill and Truong Q. Nguyen, "An augmented Lagrangian method for video restoration," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '11)*, pp.941-944, May 2011.

Ramsin Khoshabeh, Stanley H. Chan and Truong Q. Nguyen, "Spatio-temporal consistency in video disparity estimation," in *Proceedings of IEEE International Con-*

ference on Acoustics, Speech and Signal Processing (ICASSP '11), pp.885-888, May 2011.

Chapter 6

Blind Deconvolution

In Chapters 4 and 5, the convolution matrix $\mathbf{H}$ is assumed to be known. While this assumption makes the computation easy, in practice $\mathbf{H}$ can only be estimated and is never known exactly.

The objective of this chapter is to address the issue of blind deconvolution, where both $\mathbf{H}$ and $\mathbf{f}$ have to be solved from the equation $\mathbf{g} = \mathbf{H}\mathbf{f} + \eta$, with only $\mathbf{g}$ given. Equivalently in terms of convolution, we must find $\mathbf{h}$ and $\mathbf{f}$ simultaneously from

$$\mathbf{g} = \mathbf{h} * \mathbf{f} + \eta,$$

where "$*$" denotes convolution. Note that solving for $\mathbf{h}$ and $\mathbf{f}$ simultaneously is an ill-posed non-linear minimization problem. Therefore, unless prior knowledge on $\mathbf{h}$ or $\mathbf{f}$ is assumed, the problem is usually intractable.

Classical methods to solve blind deconvolution consider an alternating minimization. At the k-th iteration, the methods solve the following pair of minimization problems iteratively

$$\begin{cases} \mathbf{f}_{k+1} &= \underset{\mathbf{f}}{\operatorname{argmin}} \ \|\mathbf{h}_k * \mathbf{f} - \mathbf{g}\|^2 + \lambda_f \psi_f(\mathbf{f}) \\ \mathbf{h}_{k+1} &= \underset{\mathbf{h}}{\operatorname{argmin}} \ \|\mathbf{h} * \mathbf{f}_{k+1} - \mathbf{g}\|^2 + \lambda_h \psi_h(\mathbf{h}), \end{cases}$$

where $\psi_f(\mathbf{f})$ and $\psi_h(\mathbf{h})$ are two regularization functions. Some common choices [23,64,76] include $\psi(\mathbf{x}) = \|\mathbf{x}\|^2$, $\psi(\mathbf{x}) = \|\mathbf{x}\|_{TV}$ and $\psi(\mathbf{x}) = \|\mathbf{x}\|_1$, where $\mathbf{x}$ may be either $\mathbf{f}$ or $\mathbf{h}$.

The difficulty of blind deconvolution is that the convergence of the alternating minimization depends heavily on the initial guess. This is a dilemma because a good

Figure 6.1: Blind deconvolution result using MATLAB's command `deconvblind`.

initial guess is often a point close to the solution, which can only be found once the problem is solved. As an illustration, we refer to Fig. 6.1 which shows a typical blind deconvolution result using MATLAB's standard command `deconvblind`. The initial guess of this image is the input blurred image.

The objective of this chapter is to review the state-of-art blind deconvolution algorithms [26, 40, 97, 116], and discusses our modifications.

6.1 Estimate h

In classical blind deconvolution, estimation of the point spread function (PSF) relies on solving the problem

$$\underset{\mathbf{h}}{\text{minimize}} \ \|\mathbf{f} * \mathbf{h} - \mathbf{g}\|^2 + \lambda_h \|\mathbf{h}\|^2, \tag{6.1}$$

where we choose the regularization function $\psi_h(\mathbf{h}) = \|\mathbf{h}\|^2$ as an illustration. (6.1) is a least-squares fitting problem. The minimizer of (6.1) is the best fit to $\mathbf{g}$ and $\mathbf{f}$. If $\mathbf{f}$ is the true estimate, then the minimizer $\mathbf{h}$ is the best solution to the original problem in the l_2-norm residue sense. If $\mathbf{f}$ is an incorrect estimate, then $\mathbf{h}$ cannot be the solution of the original problem, even if it is the minimizer of (6.1).

In [40], Fergus *et al.* found that $\mathbf{h}$ can be better estimated if we replace $\mathbf{f}$ by $\nabla\mathbf{f}$, where $\nabla\mathbf{f}$ is the image gradient of $\mathbf{f}$. The idea can be intuitively understood by observing Fig. 6.2, which shows an image blurred by different Gaussian PSFs. When the variance of the Gaussian PSF increases, the texture of the image is washed out, but the edges can still be seen clearly. This result implies that a significant portion of the

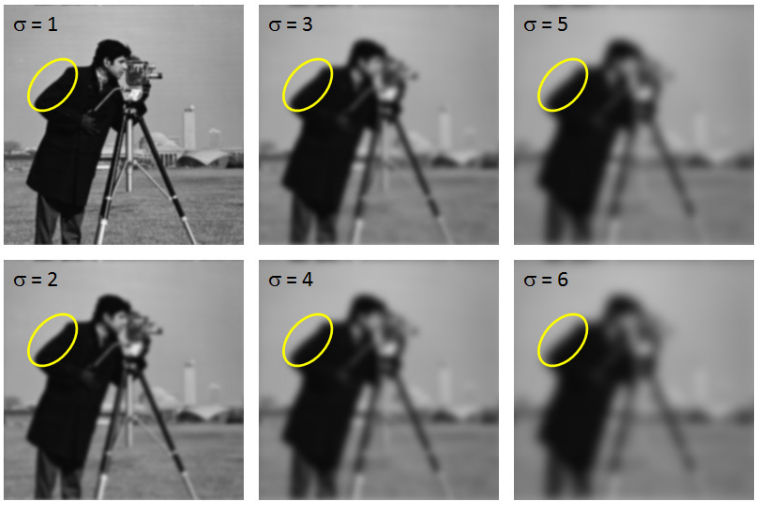


Figure 6.2: An image is blurred using Gaussian PSFs with different variance σ^2. The texture regions are smoothed when σ increases, but strong edges are still clearly seen (although blurred).

information is preserved in $\nabla\mathbf{f}$. Furthermore, we note that if $\mathbf{g} = \mathbf{f} * \mathbf{h} + \eta$, then

$$\nabla\mathbf{g} = \nabla\mathbf{f} * \mathbf{h} + \nabla\eta$$

is also valid because convolution is a linear operation. Therefore, the following minimization problem

$$\underset{\mathbf{h}}{\text{minimize}} \ \|\nabla\mathbf{f} * \mathbf{h} - \nabla\mathbf{g}\|^2 + \lambda_h\|\mathbf{h}\|^2 \tag{6.2}$$

is considered. Solving (6.2) is inexpensive, because closed-form solution exists

$$\mathbf{f} = \mathcal{F}^{-1}\left[\frac{\overline{\mathcal{F}[\nabla\mathbf{f}]}\mathcal{F}[\nabla\mathbf{g}]}{|\mathcal{F}[\nabla\mathbf{f}]|^2 + \lambda_h}\right],$$

where $\overline{(\cdot)}$ denotes complex conjugate, and $\mathcal{F}$ is the Fourier Transform operator.

The drawback of (6.2) is that (6.2) still relies on the initial estimate of $\mathbf{f}$. In the typical setting where initially $\mathbf{f} = \mathbf{g}$, solution of (6.2) converges to the dirac-delta function. In fact, if one wants to estimate $\mathbf{h}$ based on $\nabla\mathbf{f}$, $\nabla\mathbf{f}$ must be *sharp*, for otherwise $\mathbf{h} * \nabla\mathbf{f}$ is not a good prediction of $\nabla\mathbf{g}$. Now, the question is: how do we sharpen $\nabla\mathbf{f}$ without solving for $\mathbf{h}$?

6.1.1 Shock Filter

The question of how to recover edges $\nabla \mathbf{f}$ from $\mathbf{g}$ is answered by Shan *et al.* [97], Cho and Lee [26], and Xu and Jia [116]. Given a blurry image $\mathbf{g}$, they applied a shock filter [85], an iterative algorithm developed for anisotropic diffusion problems, to sharpen the image. In the k-th iteration of the shock filter, the algorithm updates the image as

$$\mathbf{f}^{k+1} = \mathbf{f}^k - \beta \operatorname{sign}(\Delta \mathbf{f}^k) \|\nabla \mathbf{f}^k\|_1.$$

Here $\nabla \mathbf{f} = [\mathbf{f}_x^T,\ \mathbf{f}_y^T]^T$ is the gradient of $\mathbf{f}$ and $\Delta \mathbf{f} = \mathbf{f}_x^2 \mathbf{f}_{xx} + 2\mathbf{f}_x \mathbf{f}_y \mathbf{f}_{xy} + \mathbf{f}_y^2 \mathbf{f}_{yy}$ is the Laplacian of $\mathbf{f}$. $\beta (= 1)$ is the step size.

Algorithm 5 shows the pseudo-code of a shock filter. A Gaussian PSF is applied to the initial guess to reduce noise before other operations. Fig. 6.3 shows the result of a shock filtered image. It can be seen that the edges are recovered, whereas texture regions are smoothed.

Algorithm 5 Shock Filter

Input: $\mathbf{f}$ and β.
Initialize $\mathbf{f}_0 = \mathbf{f} * \mathbf{h}_G$, where $\mathbf{h}_G$ = Gaussian PSF with variance $\sigma = 1$.
while not converge **do**
  $\mathbf{f}^{k+1} = \mathbf{f}^k - \beta \operatorname{sign}(\Delta \mathbf{f}^k) \|\nabla \mathbf{f}^k\|_1$
  $k = k + 1$
end while

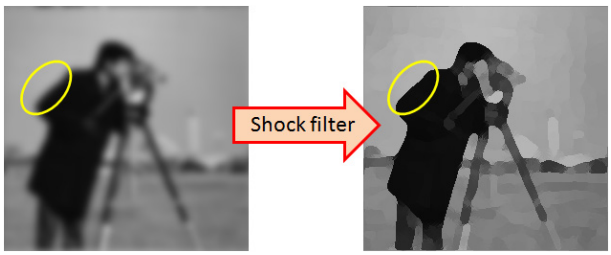


Figure 6.3: Applying shock filter to a blurry image.

6.1.2 Strong Edge Selection

In (6.2), $\nabla \mathbf{f}$ denotes *all* edges of the image. This includes both strong and weak edges. However in [116], Xu and Jia find that only strong edges should be used.

To determine the strong edges of a given image $\mathbf{f}$, Xu and Jia define a metric

$$\mathbf{R} = \frac{\sqrt{|\mathbf{h}_A * \mathbf{f}_x|^2 + |\mathbf{h}_A * \mathbf{f}_y|^2}}{\mathbf{h}_A * \sqrt{|\mathbf{f}_x|^2 + |\mathbf{f}_y|^2} + 0.5}, \tag{6.3}$$

where $\mathbf{h}_A$ is a 5×5 uniform blur PSF with entries being $1/25$. In (6.3), the numerator $\mathbf{h}_A * \mathbf{f}_x$ is the average of the horizontal gradient within a 5×5 window. Therefore, if there are small objects/textures/noise, positive and negative gradients will appear in the 5×5 window. Consequently, the average $\mathbf{h}_A * \mathbf{f}_x$ is small. On the other hand, the denominator $\mathbf{h}_A * \sqrt{|\mathbf{f}_x|^2 + |\mathbf{f}_y|^2}$ denotes the average of the *absolute* gradient within the 5×5 window. Therefore, regardless of the sizes of the object, all the absolute gradients within the window are positive. As a result, $\mathbf{R}$ differentiates the large objects versus small texture in the window.

R **M**

Figure 6.4: Illustrations of $\mathbf{R}$ and $\mathbf{M}$.

To rule out small values of $\mathbf{R}$, we define a mask $\widetilde{\mathbf{R}} = \max\{\mathbf{R} - \tau_r, 0\}$ where τ_r is a threshold. Finally, we define

$$\mathbf{M} = \max\left\{\widetilde{\mathbf{R}} \cdot \sqrt{|\mathbf{f}_x^s|^2 + |\mathbf{f}_y^s|^2} - \tau_s, 0\right\},$$

where τ_s is also a threshold, $\mathbf{f}^s$ is the shock filtered image, $\mathbf{f}_x^s$ and $\mathbf{f}_y^s$ are gradients of $\mathbf{f}^s$. $\mathbf{R}$ and $\mathbf{M}$ are shown in Fig. 6.4.

In the following, we denote the edge selected gradients as

$$\nabla^s \mathbf{f} = \text{edge selected gradient}(\mathbf{f}) = \mathbf{M} \cdot (\nabla \mathbf{f}).$$

Thus,

$$\nabla^s \mathbf{f}^s = \text{edge selected gradient(shock filtering}(\mathbf{f})).$$

6.1.3 Solve for h

With the shock filter and edge selection method, the minimization of finding $\mathbf{h}$ becomes

$$\underset{\mathbf{h}}{\text{minimize}} \ \|\nabla^s \mathbf{f}^s * \mathbf{h} - \nabla^s \mathbf{g}\|^2 + \lambda_h \|\mathbf{h}\|^2, \tag{6.4}$$

where $\mathbf{f}^s$ is the shock filtered version of $\mathbf{f}$, and ∇^s is the gradient operator with mask $\mathbf{M}$. Writing $\nabla^s \mathbf{f}^s = \begin{bmatrix} \partial_x^s \mathbf{f}^s \\ \partial_y^s \mathbf{f}^s \end{bmatrix}$, problem (6.4) can be written as

$$\begin{aligned}
&\underset{\mathbf{h}}{\text{minimize}} \ \|\nabla^s \mathbf{f}^s * \mathbf{h} - \nabla^s \mathbf{g}\|^2 + \lambda_h \|\mathbf{h}\|^2 \\
&= \underset{\mathbf{h}}{\text{minimize}} \ \left\| \begin{bmatrix} \partial_x^s \mathbf{f}^s \\ \partial_y^s \mathbf{f}^s \end{bmatrix} * \mathbf{h} - \begin{bmatrix} \partial_x^s \mathbf{g} \\ \partial_y^s \mathbf{g} \end{bmatrix} \right\|^2 + \lambda_h \|\mathbf{h}\|^2 \\
&= \underset{\mathbf{h}}{\text{minimize}} \ \|\partial_x^s \mathbf{f}^s * \mathbf{h} - \partial_x^s \mathbf{g}\|^2 + \left\|\partial_y^s \mathbf{f}^s * \mathbf{h} - \partial_y^s \mathbf{g}\right\|^2 + \lambda_h \|\mathbf{h}\|^2
\end{aligned}$$

Solution of (6.4) exists in a closed form as

$$\mathbf{h} = \mathcal{F}^{-1} \left\{ \frac{\overline{\mathcal{F}(\partial_x^s \mathbf{f}^s)} \mathcal{F}(\partial_x^s \mathbf{g}) + \overline{\mathcal{F}(\partial_y^s \mathbf{f}^s)} \mathcal{F}(\partial_y^s \mathbf{g})}{|\mathcal{F}(\partial_x^s \mathbf{f}^s)|^2 + |\mathcal{F}(\partial_y^s \mathbf{f}^s)|^2 + \lambda_h} \right\}, \tag{6.5}$$

where $\mathcal{F}$ is the Fourier Transform operator and $\overline{(\cdot)}$ denotes complex conjugate. Here, the multiplication and division equation are element-wise operations. Solving (6.5) is an inexpensive procedure, because $\nabla^s \mathbf{f}^s$ and $\nabla^s \mathbf{g}$ can be pre-computed.

6.1.4 Update f

Finding $\mathbf{h}$ is an iterative process in which $\mathbf{f}$ must be also be updated before a new $\mathbf{h}$ is found. In fact, estimation of $\mathbf{h}$ must be solved via an alternating minimization, for example,

$$\begin{cases} \mathbf{f}_{k+1} & = \underset{\mathbf{f}}{\text{argmin}} \ \|\mathbf{h}_k * \mathbf{f} - \mathbf{g}\|^2 + \lambda_f \|\mathbf{f}\|^2 \\ \mathbf{h}_{k+1} & = \underset{\mathbf{h}}{\text{argmin}} \ \|\mathbf{h} * \nabla^s \mathbf{f}_{k+1}^s - \nabla^s \mathbf{g}\|^2 + \lambda_h \|\mathbf{h}\|^2. \end{cases}$$

The minimization problem associated with $\mathbf{f}$

$$\underset{\mathbf{f}}{\text{minimize}} \ \|\mathbf{h} * \mathbf{f} - \mathbf{g}\|^2 + \lambda_f \|\nabla \mathbf{f}\|^2 \tag{6.6}$$

has a closed form solution

$$\mathbf{f} = \mathcal{F}^{-1}\left\{ \frac{\overline{\mathcal{F}(\mathbf{h})}\mathcal{F}(\mathbf{g})}{|\mathcal{F}(\mathbf{h})|^2 + \lambda_f \left[|\mathcal{F}(\partial_x)|^2 + |\mathcal{F}(\partial_y)|^2\right]} \right\},$$

where $\partial_x = [1, -1]$ and $\partial_y = [1, -1]^T$ are the horizontal and vertical gradient operators, respectively.

There are other variations of the problem, such as the following problem suggested by Xu and Jia [116]:

$$\begin{cases} \mathbf{f}_{k+1} & = \underset{\mathbf{f}}{\text{argmin}} \ \|\mathbf{h}_k * \mathbf{f} - \mathbf{g}\|^2 + \lambda_f \|\nabla \mathbf{f} - \nabla \mathbf{f}_k^s\|^2 \\ \mathbf{h}_{k+1} & = \underset{\mathbf{h}}{\text{argmin}} \ \|\mathbf{h} * \nabla^s \mathbf{f}_{k+1}^s - \nabla^s \mathbf{g}\|^2 + \lambda_h \|\mathbf{h}\|^2, \end{cases}$$

where $\mathbf{f}^s$ is the shock filtered version of $\mathbf{f}$. In this case, the solution of the $\mathbf{f}$-subproblem is (dropping the index k)

$$\mathbf{f} = \mathcal{F}^{-1}\left\{ \frac{\overline{\mathcal{F}(\mathbf{h})}\mathcal{F}(\mathbf{g}) + \lambda_f \left[\overline{\mathcal{F}(\partial_x)}\mathcal{F}(\mathbf{f}_x^s) + \overline{\mathcal{F}(\partial_y)}\mathcal{F}(\mathbf{f}_y^s)\right]}{|\mathcal{F}(\mathbf{h})|^2 + \lambda_f \left[|\mathcal{F}(\partial_x)|^2 + |\mathcal{F}(\partial_y)|^2\right]} \right\}.$$

6.1.5 Overall Algorithm for Estimating h

Algorithm 6 Estimate $\mathbf{h}$

Input: $\mathbf{g}$, λ_h and λ_f.
Initialize $\mathbf{f}_0 = \mathbf{g}$. $k = 0$.
while not converge **do**
  $\nabla^s \mathbf{f}_k^s = \text{edge selection}(\text{shock filter}(\mathbf{f}_k))$.
  $\mathbf{h}_{k+1} = \underset{\mathbf{h}}{\text{argmin}} \|\nabla^s \mathbf{f}_k^s * \mathbf{h} - \nabla^s \mathbf{g}\|^2 + \lambda_h \|\mathbf{h}\|^2$.
  $\mathbf{f}_{k+1} = \underset{\mathbf{f}}{\text{argmin}} \|\mathbf{h}_{k+1} * \mathbf{f} - \mathbf{g}\|^2 + \lambda_f \|\nabla \mathbf{f}\|^2$.
  $k = k + 1$.
end while

Algorithm 6 shows the overall algorithm to estimate $\mathbf{h}$. We used (6.6) for updating $\mathbf{f}$ as an illustration. The algorithm is terminated when $\|\mathbf{h}_{k+1} - \mathbf{h}_k\|^2 / \|\mathbf{h}_k\|^2 \leq 10^{-3}$.

6.2 Hierarchical Image Pyramid

6.2.1 Image Pyramid

Since blind deconvolution is an alternating minimization approach, and the convergence depends heavily on the initial guess, one method to improve the search is by means of building a multi-scale image pyramid. Given an image $\mathbf{g}$, we construct a sequence of down-sampled images $\{\mathbf{g}^{(i)}\}$ with a scaling factor $\sqrt{2}$ in each direction (horizontally and vertically). Starting from the lowest resolution level to the highest resolution level of the pyramid, we estimate the PSF $\mathbf{h}^{(i)}$ at the i-th level by applying Algorithm 6. The solution $(\mathbf{h}^{(i)}, \mathbf{f}^{(i)})$ is then used as the initial guess of the next level (see Algorithm 6.2.1). Fig. 6.5 illustrates an example of multi-scale pyramid.

Algorithm 7 Multi-scale pyramid

Compute $\{\mathbf{g}^{(i)}\}_{i=1}^{L}$.
for $i = 1 : L$ **do**
  Set initial guess $\mathbf{h}_0^{(i)}$ = Bi-cubic interpolation of $\mathbf{h}^{(i-1)}$ by a factor of $\sqrt{2}$.
  Set initial guess $\mathbf{f}_0^{(i)}$ = Bi-cubic interpolation of $\mathbf{f}^{(i-1)}$ by a factor of $\sqrt{2}$.
  Estimate new $(\mathbf{h}^{(i)}, \mathbf{f}^{(i)})$ using Algorithm 6, with initial guesses $\mathbf{h}_0^{(i)}$, $\mathbf{f}_0^{(i)}$.
end for

Figure 6.5: A multi-scale pyramid. The scaling factor between adjacent levels is $\sqrt{2}$.

6.2.2 Overall algorithm

Algorithm 8 Blind deconvolution

Input: $\mathbf{g}$.
Initialize $\mathbf{f}_0 = \mathbf{g}$.
Construct a multi-scale image pyramid of L levels.
for $i = 1 : L-1$ **do**
  // Determine $\mathbf{h}$ by solving the following pair of problems (Low Complexity)
  Set $k = 0$.
  while Not converge **do**
    $\nabla^s \mathbf{f}_k^s$ = edge selection(shock filter($\mathbf{f}_k$))
    $\mathbf{h}_{k+1} = \underset{\mathbf{h}}{\operatorname{argmin}} \ \|\nabla^s \mathbf{f}_k^s * \mathbf{h} - \nabla^s \mathbf{g}\|^2 + \lambda_h \|\mathbf{h}\|^2$
    $\mathbf{f}_{k+1} = \underset{\mathbf{f}}{\operatorname{argmin}} \ \|\mathbf{h}_{k+1} * \mathbf{f} - \mathbf{g}\|^2 + \lambda_f \|\nabla \mathbf{f}\|^2$
    $k = k + 1$
  end while
  $\mathbf{f}$ = Bi-cubic interpolation of $\mathbf{f}$ by a factor of $\sqrt{2}$.
  $\mathbf{h}$ = Bi-cubic interpolation of $\mathbf{h}$ by a factor of $\sqrt{2}$.
end for
for $i = L$ **do**
  // Determine $\mathbf{h}$ with sparsity constraint (Intermediate Complexity)
  Set $k = 0$.
  while Not converge **do**
    $\mathbf{h}_{k+1} = \underset{\mathbf{h}}{\operatorname{argmin}} \ \|\nabla^s \mathbf{f}^s * \mathbf{h} - \nabla^s \mathbf{g}\|^2 + \lambda_h \|\nabla \mathbf{h}\|_1$
    $\mathbf{f}_{k+1} = \underset{\mathbf{f}}{\operatorname{argmin}} \ \|\mathbf{h} * \mathbf{f} - \mathbf{g}\|^2 + \lambda_f \|\nabla \mathbf{f}\|^2$
    $k = k + 1$
  end while
end for
// Determine $\mathbf{f}$ using `deconvtv` (High Complexity)
$\mathbf{f} = \underset{\mathbf{f}}{\operatorname{argmin}} \ \mu\|\mathbf{h} * \mathbf{f} - \mathbf{g}\|^2 + \|\mathbf{f}\|_{TV2}$.

Algorithm 8 illustrates the overall procedures of blind deconvolution. The algorithm basically contains two major steps. The first step is the estimation of $\mathbf{h}$, which is discussed in the previous section. The second step is the estimation of $\mathbf{f}$. Given an estimate of $\mathbf{h}$, we use `deconvtv` to solve the following minimization problem

$$\mathbf{f} = \underset{\mathbf{f}}{\operatorname{argmin}} \ \mu\|\mathbf{h} * \mathbf{f} - \mathbf{g}\|^2 + \|\mathbf{f}\|_{TV2}, \tag{6.7}$$

where $\|\mathbf{f}\|_{TV2}$ is the isotropic TV norm on $\mathbf{f}$. However, solving (6.7) is only performed after a good estimate of $\mathbf{h}$ is found. During the intermediate steps of estimating $\mathbf{h}$, we use the low complexity formulation (6.6).

Note that there are two methods to find $\mathbf{h}$. Starting from level 1 to level $L-1$, $\mathbf{h}$ and $\mathbf{f}$ are estimated via:

$$\begin{aligned} &\underset{\mathbf{h}}{\text{minimize}} \ \|\nabla^s \mathbf{f}^s * \mathbf{h} - \nabla^s \mathbf{g}\|^2 + \lambda_h \|\mathbf{h}\|^2, \\ &\underset{\mathbf{f}}{\text{minimize}} \ \|\mathbf{h} * \mathbf{f} - \mathbf{g}\|^2 + \lambda_f \|\nabla \mathbf{f}\|^2. \end{aligned} \tag{6.8}$$

At level L, the algorithm solves

$$\begin{aligned} &\underset{\mathbf{h}}{\text{minimize}} \ \|\nabla^s \mathbf{f}^s * \mathbf{h} - \nabla^s \mathbf{g}\|^2 + \lambda_h \|\nabla \mathbf{h}\|_1, \\ &\underset{\mathbf{f}}{\text{minimize}} \ \|\mathbf{h} * \mathbf{f} - \mathbf{g}\|^2 + \lambda_f \|\nabla \mathbf{f}\|^2. \end{aligned} \tag{6.9}$$

The difference between (6.8) and (6.9) is the norm of $\mathbf{h}$. We chose $\|\nabla \mathbf{h}\|_1$ in (6.9) because $\|\nabla \mathbf{h}\|_1$ enforces sparsity on the gradients of $\mathbf{h}$, for $\mathbf{h}$ are typically smooth. However, minimizations involving $\|\nabla \mathbf{h}\|_1$ are computationally more expensive. Therefore, the algorithm only solves (6.9) in the final resolution level.

6.3 Results

(a) Input (b) Output (c) Estimated PSF

Figure 6.6: Blind deconvolution result.

Fig. 6.6(a) shows an image blurred by an unknown PSF. The goal is to recover both $\mathbf{f}$ and $\mathbf{h}$. The image size is 800×800, and the PSF size is 33×33. The initial guess of the algorithm is $\mathbf{f} = \mathbf{g}$, where $\mathbf{g}$ is the input blurred image. Shock filter is applied with $\beta = 1$, and maximum iteration is 5. During levels 1 to $L-1$, the $\mathbf{h}$-subproblem has

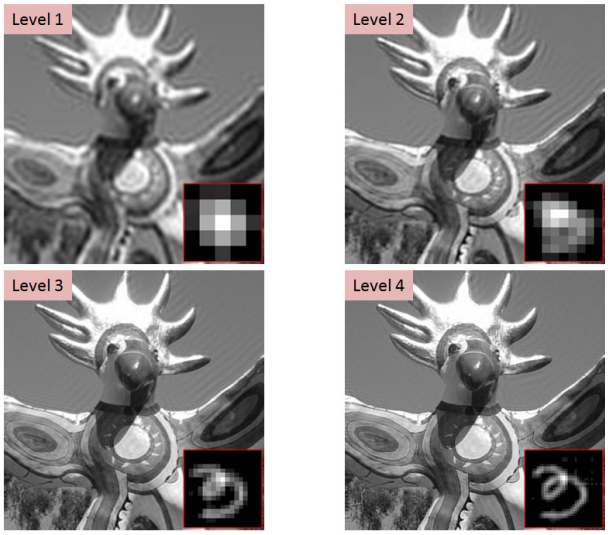


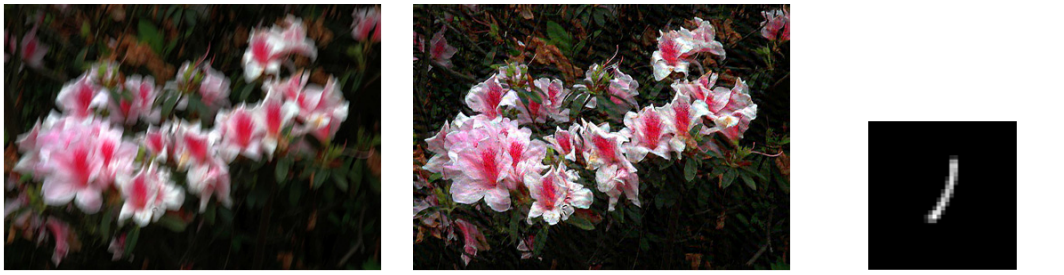
Figure 6.7: Multi-scale pyramid. From low resolution to high resolution, the quality of **f** and **h** increase.

the parameter $\lambda_h = 0.01$, and **f**-subproblem has the parameter $\lambda_f = 2 \times 10^{-3}$. At level L, $\lambda_h = 10$, $\lambda_f = 2 \times 10^{-3}$. The final step using `deconvtv` has a parameter $\mu = 5000$.

The result is shown in Fig. 6.6(b), and the estimated PSF is shown in Fig. 6.6(c). The run-time of the algorithm is 152 seconds on Intel Qual Core Q9550 2.8GHz, 4GB DDR3 RAM, Windows 7/ MATLAB 2010.

Fig. 6.7 shows the intermediate steps of the algorithm. In particular, we show the estimated **h** and **f** at each multi-scale level. It can be seen that the result improves when resolution increases. Note also that during the intermediate steps, only the luminance component of the color image is used. Full color image recovery is performed during the final step (high complexity) to estimate **f**.

More results are shown in Fig. 6.8 and Fig. 6.9. The images were previously used by Xu and Jia [116]. These two images were blurred by unknown motion blurs. The size of the flower image is 700×494 with PSF size 35×35. The size of the wall image is 463×511 with PSF size 35×35. As shown in both images, the recovered result is significantly sharper than the input image.



(a) Input (b) Output (c) Estimated PSF

Figure 6.8: Blind deconvolution on "flower" image. The run time is 100 seconds.

(a) Input (b) Output (c) Estimated PSF

Figure 6.9: Blind deconvolution on "wall" image. The run time is 79 seconds.

6.4 Summary

In this chapter, blind deconvolution is described as a non-linear optimization problem. State-of-art algorithms observe that edges are easy to be sharpened than texture. Also, sharp edges are useful in determining the PSF. Consequently, the blind deconvolution is formulated as an alternating minimization problem in which the image edges are used. For large scaled images, a multi-scale image pyramid is formed so that solution of the current level is propagated to the next level as initial guess. Results show that the blind deconvolution algorithm is able to recover blurred images distorted by complicated blurs.

A limitation of the blind deconvolution algorithm discussed in this chapter is the assumption that the blur is spatially invariant, meaning that all pixels are blurred equally. In practice, spatially invariant assumption is difficult to be satisfied. For an image having two objects at different layers of depth, only one object can be focused.

Therefore, the blur is spatially variant. In the next two chapters, we will address the issue of spatially variant blur.

6.5 Acknowledgment

This chapter, in part, is a reprint of the following papers

Stanley H. Chan, and Truong Q. Nguyen, "Single Image Spatial Variant Out-of-focus Blur Removal," to appear in *Proceedings of IEEE International Conference on Image Processing*, Sep 2011.

Stanley H. Chan, "Single Image Two-layered Out-of-focus Blur Removal," submitted to *IEEE Transactions on Image Processing*, Oct 2011.

Chapter 7

Analysis of Spatially Variant Blur

The second assumption made in Chapter 4 and 5 is that blurs are always spatially invariant. Consequently, the convolution matrix is always a block-circulant with circulant-block (BCCB) matrix, which can be diagonalized using Fourier Transforms. However, in practice, most blurs are spatially variant - pixels at different locations are blurred differently.

This chapter addresses two issues of spatially variant blur. First, we propose an efficient method to construct the convolution matrix. We exploit the submatrix structure of the convolution matrix and systematically assigning values to the nonzero locations. Second, we discuss the spectral properties of spatially variant convolution matrices. We derive the upper and lower bounds of the condition numbers.

7.1 Constructing Convolution Matrices

7.1.1 Motivation

Let us recall the following classical spatially invariant imaging model:

$$\mathbf{g} = \mathbf{H}\mathbf{f} + \eta,$$

where $\mathbf{H}$ is the convolution matrix characterized by the point spread function $h(x, y)$, $\mathbf{f}$ is a vector representing the object (usually as an unknown variable), $\mathbf{g}$ is a vector representing the observed image, and η is a noise vector.

As discussed in chapter 2, when point spread function (PSF) is *spatially invariant*,

i.e., all pixels in the image are blurred uniformly, the forward operation **Hf** can be done via Fourier Transforms. However, if the PSF is *spatially variant* (which is quite common in out-of-focus and motion blur problems), Fourier Transforms cannot be used.

The effectiveness of computing a matrix-vector multiplication **Hf** is essential for iterative methods such as PDCO [61], LSQR [86], RestoreTools [54], projected gradient [17] and many others. In most of these papers, however, there is no discussion on how to construct **H** from the PSF. While it is understandable that invariant convolution can be performed via Fourier Transforms (which makes the construction of **H** not necessary), variant convolution must require **H**.

In literature there are two classes of methods for constructing the convolution matrix. The first class of methods considers spatially variant blur as a set of invariant blurs. Each invariant blur is computed using using Fast Fourier Transform (FFT). Popular image restoration tools such as RestoreTools [65] and HNO [55] belong to this class. However, these methods fail in case of a complicated set of PSFs. The second class of methods constructs the convolution matrix explicitly without partitioning the image into blocks. In [108], Vogel provides codes to construct a small sized invariant convolution matrices. However, he never consider the general variant convolution matrix. Also, Vogel's method is slow and memory demanding. MATLAB's built-in command `convmtx2` is an efficient algorithm to construct a invariant convolution matrix. However, there is no variant version of the program.

This part of the chapter addresses the issue of how to construct a spatially variant convolution matrix efficiently. For small and medium sized images, constructing the convolution matrix explicitly has the following advantages:

1. Having the sparse convolution matrix allows us to apply many powerful linear algebraic tools such as LU/ QR/ Cholesky/ SVD. Consequently, advanced algorithms such as truncated singular value decomposition (TSVD [55]) can be applied.

2. If the PSF is spatially invariant and its support is small, the forward operation **Hf** using the convolution matrix is much more effective than FFT. Consider an image of n pixels and kernel of l entries. Cost for matrix-vector multiplication is $O(nl)$, but cost for FFT is $O(n \log n)$.

3. If the PSF is spatially variant, FFT methods cannot be used. Although one can approximate the situation using locally spatially invariant PSFs, this will fail for

large number of PSFs and small block size. Convolution matrix can resolve this issue because once the matrix is constructed, cost of matrix vector multiplication is $O(nl)$, regardless the number of PSFs.

7.1.2 Algorithm to Construct the Convolution Matrix

Structure of H

The proposed algorithm is a systematic way of allocating PSF values into the convolution matrix. Suppose that an image has size $M \times N$, and the PSF has size $P \times Q$. Not considering the circular boundary conditions, the convolution matrix has size $(M+P-1)(N+Q-1) \times MN$. Partitioning the convolution matrix into submatrices $H_{i,j}$ (each has size $(M+P-1) \times M$) yields

$$\mathbf{H} = \begin{pmatrix} H_{11} & H_{12} & \cdots & H_{1l} \\ H_{21} & H_{22} & \cdots & H_{2l} \\ \vdots & \vdots & \ddots & \vdots \\ H_{k1} & H_{k2} & \cdots & H_{kl} \end{pmatrix} = \begin{pmatrix} \times & & & \\ \times & \times & & \\ & \ddots & \ddots & \\ & & \times & \times \\ & & & \times \end{pmatrix} \tag{7.1}$$

The (i,j)-th submatrix $H_{i,j}$ the operation *from* the j-th column of the input *to* the i-th column of the output. Since the PSF has a small support relative to the image size, most of the off-diagonal submatrices are zero. The nonzero submatrices are located along the diagonal, forming a banded diagonal of width Q submatrices, where Q is the number of rows of the PSF. The right hand side of (7.1) shows an example of $Q = 2$.

Indexing

The proposed method introduces a 4-dimensional cube to allocate PSF values in **H**, regardless of whether the PSF is spatially variant or not. The 4-dimensional cube has size $M \times P \times N \times Q$. The (i,p,j,q)-th entry of the cube is the (p,q)-th kernel value at pixel (i,j). Stacking the entries of the cube into a column vector, we denote it as T_{val}.

Corresponding to T_{val} there is a column index and a row index vector T_{col} and T_{row} respectively. The column and row indices run along the principle diagonal submatrices, then run along the submatrice on one-diagonal off the principal, two-diagonal off

the principal and so on. Within each submatrix, the column and row indices also run in the similar manner: first run along the principal diagonal, then run along one diagonal off the principal and so on.

Note that the cube contains only the *nonzero* elements of the PSF. As compared to [65] where a PSF is first padded with zeros to the size of image and then stored as an entry of a cell structure, the proposed 4-dimensional cube requires much less memory. Besides, it is possible to further reduce the memory of the proposed method by considering repeating PSFs. This is a subject of our future work.

Boundary Conditions

One problem that we have not solved is the boundary issues. To handle the boundary conditions, let us consider the case of periodic boundary condition. For a fixed q-th column of the kernel, and fixed j-th column of the image, the (i,p)-th entry should be reallocated to $(i-p+1,p)$. Pictorially the (i,p)-th entry can be visualized as

$$\begin{pmatrix} & \cdots & & \cdots & \\ & \cdots & & \cdots & \\ & \cdots & & \cdots & \\ \times & \times & \times & \times & \times \\ \triangle & \triangle & \triangle & \triangle & \triangle \\ \circ & \circ & \circ & \circ & \circ \end{pmatrix} \rightarrow \begin{pmatrix} & \cdots & & \cdots & \\ & & \times & \triangle & \circ \\ & \times & \triangle & \circ & \circ \\ \times & \triangle & \circ & \circ & \circ \\ \triangle & \circ & \circ & \circ & \circ \\ \circ & \circ & \circ & \circ & \circ \end{pmatrix}.$$

The reallocation of the (j,q)-th entry is performed similarly so that the (j,q)-th entry is reallocated to $(j-q+1,q)$.

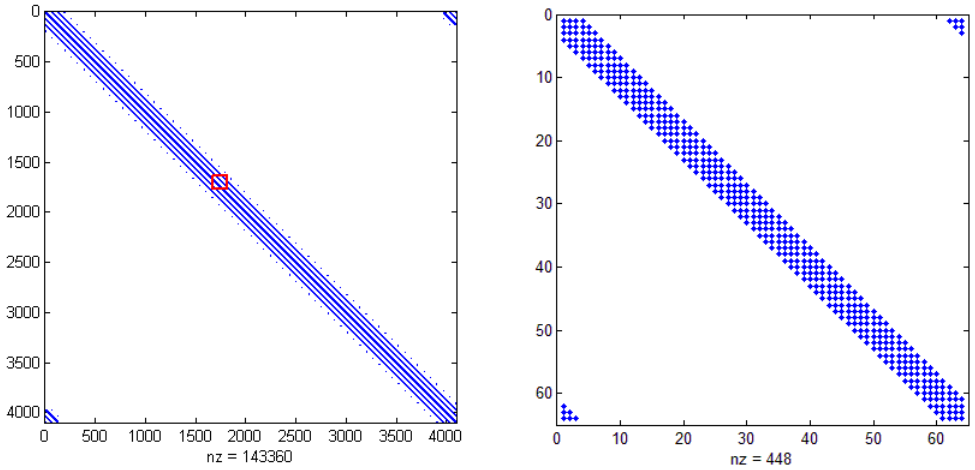
Finally, the nonzero submatrices in the first $Q/2$ rows of $\mathbf{H}$ has to be moved to the last $Q/2$ rows, and vice versa for the non-zero submatrices in the last $Q/2$ rows. This is illustrated in Eq. 7.2. Note that after moving the rows, size of $\mathbf{H}$ becomes

$(M+P-1)N \times MN$.

$$\begin{pmatrix} \circ & & & & \\ \times & \times & & & \\ & \ddots & \ddots & & \\ & & \times & \times & \\ & & & \triangle & \end{pmatrix} \rightarrow \begin{pmatrix} \times & \times & & & \triangle \\ \times & \times & & & \\ & & \ddots & & \\ & & & \times & \times \\ \circ & & & \times & \times \end{pmatrix} \tag{7.2}$$

Since a submatrix has the same structure as that of **H**, applying the above method to each submatrix the boundary conditions are satisfied. Note that size of **H** is reduced to $MN \times MN$.

Fig. 7.1 is an example showing the location of non-zero entries of the sparse matrix. The kernel used here has size 5×7.



(a) sparse matrix (b) zoom in of (a)

Figure 7.1: Structure of the constructed sparse matrix.

7.1.3 Comparisons

In this section we compare the proposed method with state-of-art approaches, namely `bccb` [108], `psfMatrix` [65], [55], and a straight forward for-loop approach. In [108] Vogel did not explicitly mention about the construction of the convolution matrix, but codes of constructing the matrix is available. `psfMatrix` is an FFT based approach, where the convolution is defined through operations. The for-loop approach is a straight

forward implementation of spatially variant PSFs. Suppose every pixel of an $M \times N$ image f has a different PSF $h_{x,y}$. Then the blurred pixel value at position (x, y) is calculated using the following algorithm.

Algorithm 9 For-Loop implementation

for $x = 1 : M$ **do**
  for $y = 1 : N$ **do**
    $g(x, y) = \sum_{i,j} h_{x,y}(x - i, y - j) f(i, j)$
  end for
end for

Motion Blur using Proposed Method

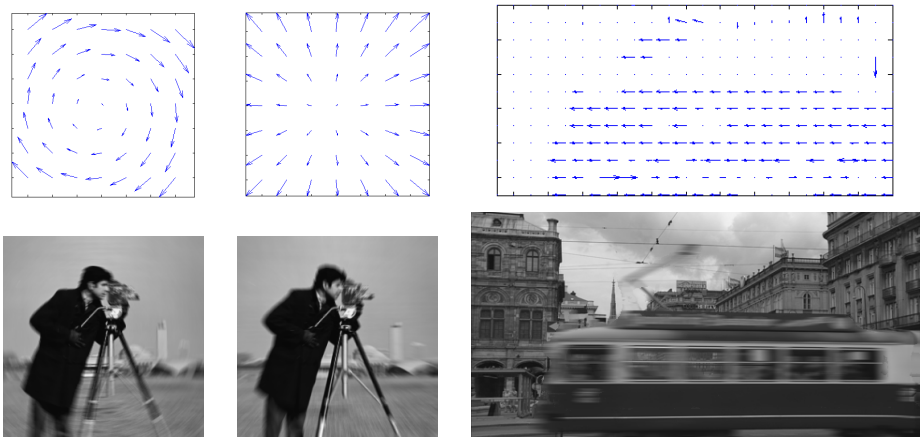


Figure 7.2: Results by the proposed algorithm. The blur is calculated based on motion vectors specified in the top row. There is no partitioning of the image into blocks.

In Fig. 7.2, the images are blurred according to the motion vector fields specified. Although Fig. 7.2 shows a coarse set of motion vectors, the actual motion vectors used for calculation are dense, i.e., every pixel has a different PSF. The effectiveness of the proposed algorithm is that the convolution matrix constructed without requiring a special model of motion, such as rotation and translation.

Comparing Spatially Invariant Blur

The purpose of this comparison is to show the proposed method for spatially invariant cases. The image size ranges from SQCIF (176×144) to CIF (352×288). The PSF is Gaussian with fixed variance $\sigma = 3$. Sizes of the PSF varies from 3×3 to 7×7.

Results are shown in Table 7.1. As one can expect, explicitly constructing a convolution matrix takes time. Therefore, the construction time of `bccb` and the proposed method are longer than that of `psfMatrix` and for-loop, especially when image size increases. However, as far as the matrix-vector multiplication is concerned, the proposed method is faster than other methods.

Table 7.1: Invariant PSF. Time to construct the convolution, and time to perform one matrix-vector multiplication.

Construction Time (sec)					
image size	kernel size	`bccb` [108]	`psfMat -rix` [65]	for-loop	Propose
128x96	5x5	38.6572	0.0423	0.0014	0.5196
	7x7	166.9316	0.0419	0.0014	3.3319
352x288	5x5	fail	0.1072	0.0015	30.742
	7x7	fail	0.1002	0.0015	90.235
Matrix-vector multiplication Time (sec)					
image size	kernel size	`bccb` [108]	`psfMat -rix` [65]	for-loop	Propose
128x96	5x5	0.0021	0.043	0.0754	0.0031
	7x7	0.0032	0.0424	0.0773	0.0045
352x288	5x5	fail	0.1839	0.6357	0.0139
	7x7	fail	0.1767	0.6357	0.0238

Comparing Spatially Variant Blur

This comparison concerns about spatially variant PSFs. First, an image is partitioned into blocks and each block is blurred using a (different) PSF. Therefore, smaller block size implies more blocks and hence more PSFs. The PSFs in this experiments are Gaussian PSFs with size 5×5. Its variance is a function of the position of the image. Pixels around the center of the image have smaller variance, whereas pixels near the edge of the image have larger variance.

As shown in Table 7.2, `bccb` fails to construct the convolution matrix due to lack of memory. `psfMatrix` defines an object easily for large block sizes, but fails for small block sizes. This is because `psfMatrix` has to store a large number of PSFs, and each PSF has to be padded with zeros to make itself the same size as the image.

The time needed to construct a convolution matrix using the proposed method is 10 seconds on average, and the computing time for a matrix-vector multiplication is as short as 0.001 seconds. Compared to `psfMatrix` where the computing time can be

as long as 20 seconds (image size 256×256, block size 16×16), the proposed method is clearly better performed.

Table 7.2: Variant PSF. Time to construct the convolution matrix, and time to perform one matrix-vector multiplication.

Construction Time (sec)					
image size	block size	`bccb` [108]	`psfMat -rix` [65]	for-loop	Proposed
256x256	64x64	fail	0.2227	0.0015	10.3789
	32x32	fail	0.7129	0.0015	10.6477
	16x16	fail	2.6279	0.0014	10.8041
	8x8	fail	fail	0.0014	11.0712
	4x4	fail	fail	0.0014	11.7754
Matrix-vector multiplication Time (sec)					
image size	block size	`bccb` [108]	`psfMat -rix` [65]	for-loop	Proposed
256x256	64x64	fail	1.1633	0.4112	0.0291
	32x32	fail	4.3206	0.4102	0.0287
	16x16	fail	23.439	0.4105	0.0424
	8x8	fail	fail	0.4137	0.0345
	4x4	fail	fail	0.4103	0.0320

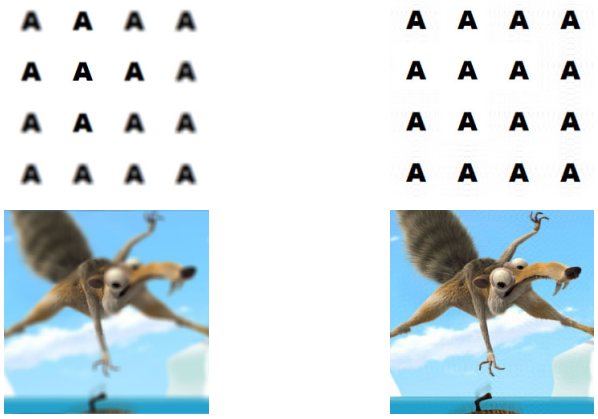
Deblurring Study

The final experiment compares the performance of the methods for solving image restoration problems. Here, our goal is not to test a specific image restoration algorithm but to compare the speed improvement. Therefore, we illustrate the results by solving a Tikhonov regularized least square problem using LSQR [86]: minimize $\|\mathbf{Hf}-\mathbf{g}\|^2+\lambda\|\mathbf{f}\|^2$, with $\mathbf{H}$ being a spatially (in)variant convolution matrix, $\mathbf{f}$ a 256×256 image arranged in lexicographic order, and $\mathbf{g}$ the observed image. Here we set $\lambda = 10^{-2}$, tolerance $\texttt{atol} = \texttt{btol} = 10^{-6}$.

The proposed method explicitly constructs the sparse convolution matrices. Therefore, the computing time for matrix-vector multiplication is significantly less than its counterparts. Especially for spatially varying blurs, the proposed method is the only one that successfully completes the task. Table 7.3 shows the timings, and Fig. 7.3 shows some of the reconstructed images.

Table 7.3: Computing time using LSQR.

A. spatially invariant			
Method	Construction Time (sec)	Computation Time (sec)	Total Time (sec)
`psfMatrix`	0.1254	21.477	21.6024
Proposed	16.418	3.4811	19.8991
B. spatially variant (block size 16×16)			
`psfMatrix`	3.5319	> 1000	> 1000
Proposed	60.092	5.5031	65.5955



(a) Shift varying blurred (b) Reconstructed using LSQR

Figure 7.3: Image reconstruction of shift varying blur using proposed method with LSQR. The average time for LSQR to converge is 60 seconds (gray scale), and 180 seconds (color).

7.2 Eigen-values of Spatially Variant Convolution Matrix

7.2.1 Motivation

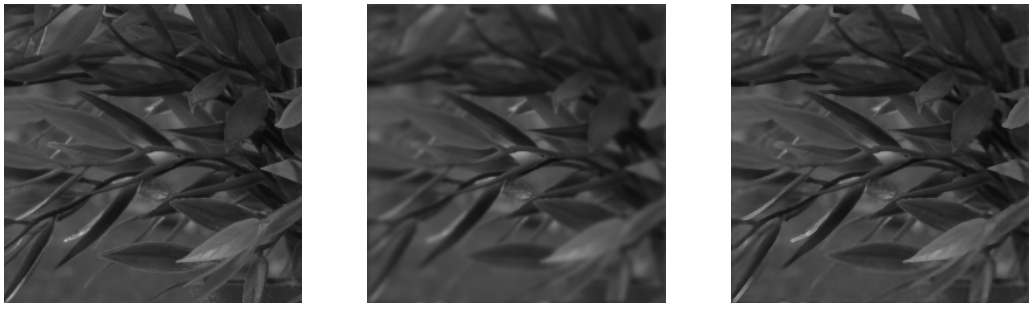
A classical image restoration problem is to solve the following least-squares minimization

$$\underset{\mathbf{f}}{\text{minimize}} \ \ \|\mathbf{Hf} - \mathbf{g}\|^2 + \alpha\|\mathbf{Df}\|^2, \tag{7.3}$$

where α is a regularization parameter and $\mathbf{D}$ is a linear transformation applied to $\mathbf{f}$. Problem (7.3) is also called the Tikhonov regularized least-squares. When $\mathbf{H}$ is a spatially variant convolution matrix, many classical results cannot be applied. In particular, for the case of spatially variant blur, the eigenvalues of $\mathbf{H}$ are not the Fourier coefficients.

In the following sections, we study the eigenvalues of $\mathbf{H}^H\mathbf{H}$, which plays a vital

role in solving (7.3). We estimate the upper and lower bounds on the largest and smallest eigenvalues of $\mathbf{H}^H\mathbf{H}$, and hence derive the bounds of the condition number of $\mathbf{H}^H\mathbf{H}$. We are particularly interested in spatially-variant convolution matrices arising from spherical aberration and defocus with different object depths. As each pixel is approximately blurred by a Gaussian point spread function (PSF), the eigenvalues of the PSFs are *nonnegative*. Fig. 7.4 shows the simulation involving a spherical aberration and its restoration result.



(a) Original image (b) Simulated blurred image (c) Restored image

Figure 7.4: An illustration of spherical aberration and restoration. The spatially variant convolution matrix used in (b) is generated using [18]. The restoration is performed using a modification of the least-squares total variation minimization [19].

7.2.2 Derivation of Upper and Lower Bounds

For simplicity, the derivations are based on one-dimensional signals, but the results can be extended to the two-dimensional case. All matrices are assumed to be of size $n \times n$ and have real entries. Since real matrices can have complex eigenvalues, by the smallest (or largest) eigenvalue we mean the eigenvalue with the smallest (or largest) complex modulus. The smallest and largest eigenvalues of a matrix $\mathbf{H}$ are denoted as $\lambda_{\min}(\mathbf{H})$ and $\lambda_{\max}(\mathbf{H})$, respectively. Also, $|\mathbf{\Lambda}|$ denotes the element-wise complex modulus of $\mathbf{\Lambda}$, and $\mathbf{\Lambda}^*$ denotes the element-wise complex conjugate of $\mathbf{\Lambda}$.

Definition 3. *[32] A matrix* $\mathbf{H}$ *is* circulant *if each row is a circular shift of its preceding row. If* $\mathbf{H}$ *is a column vector, we use* $\texttt{CircMtx}(\mathbf{H}, k)$ *to denote the circulant matrix generated by* $\mathbf{H}$*, with* $\mathbf{H}$ *being put in the* k*-th column.*

For example, if $\mathbf{h} = [1, 2, 3, 2, 1, 0, \ldots, 0]^T$, then a 10×10 circulant matrix

$\texttt{CircMtx}(\mathbf{h}, 3)$ is

$$\texttt{CircMtx}(\mathbf{h}, 3) = \begin{bmatrix} 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 2 \\ 2 & 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 2 & 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 3 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 3 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 3 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 2 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 2 \\ 2 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 \end{bmatrix}$$

Our approach in analyzing a spatially-variant convolution matrix $\mathbf{H}$ is to consider all circulant matrices generated by the columns of $\mathbf{H}$. Thus, we define the *circulant components* of $\mathbf{H}$ as follows.

Definition 4. *Partitioning* $\mathbf{H}$ *into* n *column vectors as* $\mathbf{H} = (\mathbf{H}_1|\mathbf{H}_2|\cdots|\mathbf{H}_n)$*, where* $\mathbf{H}_k$ *denotes the* k*-th column of* $\mathbf{H}$*, we define the* k*-th* circulant component *of* $\mathbf{H}$ *as* $\texttt{CircMtx}(\mathbf{H}_k, k)$*, denoted by* $\mathbf{H}_k$*.*

For example, if

$$\mathbf{H} = \begin{bmatrix} 4 & 3 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 5 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 3 & 3 & 3 & 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 3 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 & 3 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 4 & 0 & 2 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 5 \end{bmatrix},$$

$\mathbf{H}_5 = \texttt{CircMtx}([0,\ 0,\ 4,\ 2,\ 1,\ 2,\ 4,\ 0,\ 0,\ 0]^T, 5)$.

The following main result states that the smallest eigenvalue of $\mathbf{H}^H\mathbf{H}$ is lower-bounded by the smallest eigenvalue among all of the circulant components of $\mathbf{H}$.

Theorem 2. *Let* $\mathbf{H}$ *be a spatially-variant convolution matrix, and let* $\mathbf{H}_1$, $\mathbf{H}_2, \ldots, \mathbf{H}_n$ *be the circulant components of* $\mathbf{H}$*. If all eigenvalues of* $\mathbf{H}_k$ *are nonnegative, the smallest eigenvalue of* $\mathbf{H}^H\mathbf{H}$ *is bounded from below by*

$$|\lambda_{\min}(\mathbf{H}^H\mathbf{H})| \geq \min_k \left\{|\lambda_{\min}(\mathbf{H}_k)|^2\right\},$$

where $|\lambda_{\min}(\mathbf{H}_k)|$ *is the smallest eigenvalue of* $\mathbf{H}_k$*.*

Proof. See Appendix C. □

By flipping the inequality signs, we have a similar result for the maximum eigenvalue of $\mathbf{H}^H\mathbf{H}$.

Theorem 3. *Let* $\mathbf{H}$ *be a spatially-variant convolution matrix, and let* $\mathbf{H}_1$, $\mathbf{H}_2, \ldots, \mathbf{H}_n$ *be the circulant components of* $\mathbf{H}$*. If all eigenvalues of* $\mathbf{H}_k$ *are nonnegative, the largest eigenvalue of* $\mathbf{H}^H\mathbf{H}$ *is upper-bounded by*

$$|\lambda_{\max}(\mathbf{H}^H\mathbf{H})| \leq \max_k \left\{|\lambda_{\max}(\mathbf{H}_k)|^2\right\},$$

where $|\lambda_{\max}(\mathbf{H}_k)|$ *is the largest eigenvalue of* $\mathbf{H}_k$*.*

Using Theorems 2 and 3, we derive a corollary for the condition number of $\mathbf{H}^H\mathbf{H}$.

Corollary 1. *Suppose* $\mathbf{H}$ *is a spatially-variant matrix, and let* $\mathbf{H}_1$, ..., $\mathbf{H}_n$ *be the circulant components of* $\mathbf{H}$*. The condition number of* $\mathbf{H}^H\mathbf{H}$ *is bounded from above by*

$$\text{cond}\left(\mathbf{H}^H\mathbf{H}\right) \leq \frac{\max_k \left\{|\lambda_{\max}(\mathbf{H}_k)|^2\right\}}{\min_k \left\{|\lambda_{\min}(\mathbf{H}_k)|^2\right\}}, \tag{7.4}$$

where $\lambda_{\min}(\mathbf{H}_k)$ *and* $\lambda_{\max}(\mathbf{H}_k)$ *are the minimum and maximum eigenvalues of* $\mathbf{H}_k$*, respectively.*

Corollary 1 implies that a spatially-variant blur (e.g., spherical aberration) can be interpreted as a set of spatially-invariant blurs. The condition number of $\mathbf{H}^H\mathbf{H}$ is never larger than the upper bound given in (7.4). Moreover, in the spherical aberration case where the blur consists of a collection of Gaussian PSFs, the bound in (7.4) can be

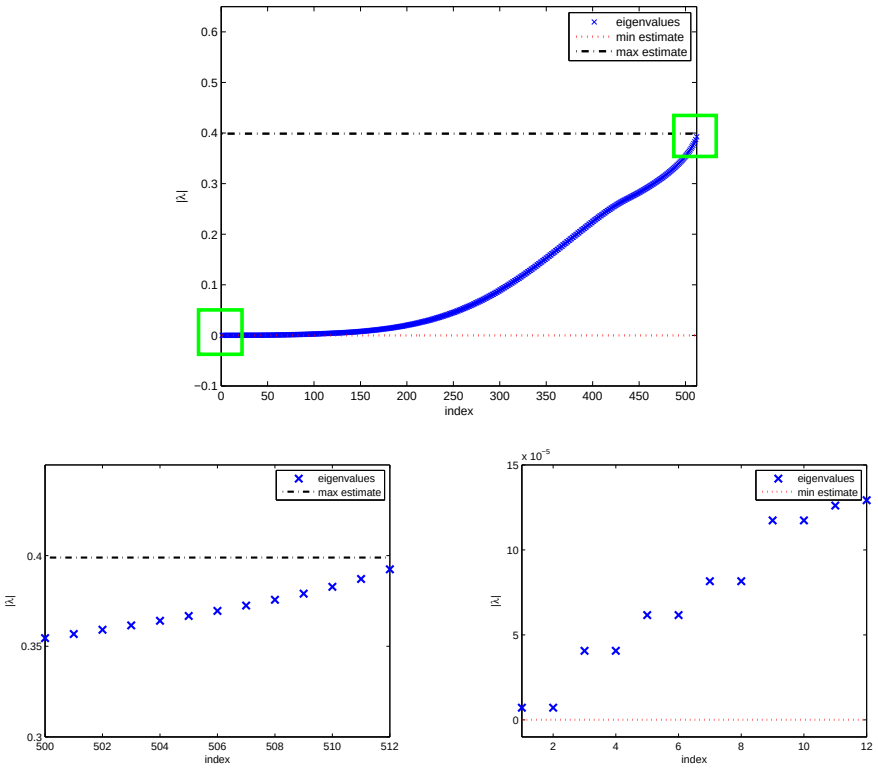


Figure 7.5: Example: Eigenvalues of $\mathbf{H}$. The red and black dotted lines indicate the (estimated) lower and upper bounds respectively.

computed using the PSFs with the largest and smallest variance. The following corollary is useful in analyzing the solution to the regularized least-squares problem expressed in (7.3).

Fig. 7.2.2 shows the distribution of the eigenvalues of a one-dimensional convolution matrix

$$\mathbf{H} = \sum_{k=1}^{N} \mathbf{E}_k \mathbf{H}_k,$$

where $\mathbf{H}_k = \text{Gaussian}(9 \times 9,\ \sigma = 1 + \frac{k}{2N})$, $N = 512$. As seen, the bounds are close to the true maximum and minimum eigenvalues.

Corollary 2. *The smallest eigenvalue of* $\mathbf{H}^H\mathbf{H} + \alpha\mathbf{D}^H\mathbf{D}$ *is bounded by*

$$\left|\lambda_{\min}\left(\mathbf{H}^H\mathbf{H} + \alpha\mathbf{D}^H\mathbf{D}\right)\right| \geq \min_k \left\{ \min_j \left\{ |\lambda_j^{\mathbf{H}_k}|^2 + \alpha|\lambda_j^{\mathbf{D}}|^2 \right\} \right\},$$

where $\lambda_j^{\mathbf{H}_k}$ *is the* j*-th eigenvalue of* $\mathbf{H}_K$ *and* $\lambda_j^{\mathbf{D}}$ *is the* j*-th eigenvalue of* $\mathbf{D}$.

Proof. See Appendix C. □

7.3 Summary

Two issues are discussed in this chapter. The first issue is the question of how to construct a convolution matrix. We proposed an efficient algorithm to construct a sparse convolution matrix for spatially variant blur. Experimental results show that if the blur consists of a large number of invariant PSFs, the proposed method is the only method that works. Consequently, when the proposed method is embedded in other standard image restoration algorithms, speed gain is significant.

The second issue is the eigenvalue property of spatially-variant convolution matrices. If the eigenvalues of the circulant components of a convolution matrix are nonnegative, we show that the smallest eigenvalue of the convolution matrix is lower bounded by the minimum eigenvalue among all circulant components. Consequently, bounds on condition numbers can be derived. We also derive the bounds on the eigenvalues of the normal equation matrix arises from least-squares image restoration formulation.

7.4 Acknowledgment

This chapter, in part, is a reprint of the following papers

Stanley H. Chan, Ankit K. Jain, Truong Q. Nguyen and Edmund Y. Lam, "Bounds for the condition numbers of spatially-variant convolution matrices in image restoration problems," in *OSA Topical Meeting in Signal Recovery and Synthesis*, paper SMA4, July 2011.

Stanley H. Chan, "Constructing a sparse convolution matrix for shift varying image restoration problems," in *Proceedings of IEEE International Conference on Image Processing (ICIP '10)*, pp.3601-3604, Hong Kong, Sept 2010.

Chapter 8

Spatially Variant Out-of-Focus Blur Removal

This chapter addresses the problem of two-layer out-of-focus blur removal from a single image, in which either the foreground or the background is in focus while the other is out of focus. To recover details from the blurry parts, the existing blind deconvolution algorithms are insufficient as the problem is spatially variant. The proposed method exploits the invariant structure of the problem by first predicting the occluded background. Then a blind deconvolution algorithm is applied to estimate the blur kernel and a coarse estimate of the image is found as a side product. Finally, the blurred region is recovered using total variation minimization, and fused with the sharp region to produce the final deblurred image.

8.1 Introduction

For an image consisting of multiple layers of depth – objects at different depth levels – only one of them can be in focus using a standard camera. Those not being focused are blurred, and this type of blur is referred to as the out-of-focus blur. In terms of wave optics, out-of-focus blur is the result of additional (or insufficient) phase propagation from the desired image plane to the actual image plane [51, Ch. 6.4].

Recovering an out-of-focus blurred image using post-processing computational techniques is an ill-posed non-linear problem in general, and seeking a universal solution

is almost impossible. However, in a simplified scenario where there is only one foreground object and a background scene, recovering the image becomes tractable. The goal of this chapter is to present a method that restores both the foreground and background, using a single image at a low computational cost.

Although the two-layered problem is a special case of the general out-of-focus blur problem, some challenges remain. The first challenge is the *spatially variant* property due to different blurs occuring in the foreground and background. Spatially variant problems are computationally intensive to solve, because the Fourier-based methods cannot be used. The second challenge is the need for *blind* deconvolution as the blur kernel is unknown. Blind deconvolution is difficult because simultaneous recovery of image and kernel is an ill-posed nonlinear problem.

In this chapter, we present a single image spatially variant blind deconvolution method. The method takes a single image as the input and separates the foreground and background using alpha matting methods. To handle the spatially variant issue, we propose a photometric model that allows us to transform the variant problem into an invariant problem. We show that by inpainting (filling in) the occluded region in the background image, not only does the variant problem becomes invariant, but also ringing artifacts resulting from the classical approach are suppressed. Additionally, we present an efficient blur kernel estimation algorithm that combines the concepts of blur from image gradient, strong edge selection, joint deblurring and kernel estimation.

8.1.1 Related Works

In a single image deblurring problem, if the blur is spatially invariant, the observed image is related to the input image as

$$\mathbf{g} = \mathbf{h} * \mathbf{f} + \eta,$$

where $\mathbf{g}$ is the observed blur image, $\mathbf{f}$ is the unknown sharp image, $\mathbf{h}$ is the blur kernel, η is the noise and $*$ denotes convolution.

If $\mathbf{h}$ is known, recovering $\mathbf{f}$ from $\mathbf{g}$ can be done using classical methods such as Wiener deconvolution [49], Lucy-Richardson deconvolution [73], or regularized least-squares deconvolution [76]. Better approaches such as total variation minimization [95] and its variations [19] can also be used. If high quality recovery results are required, one

can consider computationally intensive algorithms such as [97].

If $\mathbf{h}$ is unknown, blind deconvolution methods are needed to repeatedly estimate the blur kernel and predict the underlying image in an alternating minimization procedure. In [26], Cho and Lee proposed a fast and reliable blind deconvolution algorithm using image gradients. Later, Xu and Jia [116] improved this method by selecting strong gradients.

When $\mathbf{h}$ is spatially variant, Nagy and O'Leary [77] suggested the following model:

$$\mathbf{g} = \sum_{i=1}^{p} \alpha_i \cdot (\mathbf{h}_i * \mathbf{f}), \tag{8.1}$$

where p is the number of blur kernels, $\mathbf{h}_i$ is the i-th blur kernel, α_i is a binary mask indicating the contribution of the i-th kernel, and "$\cdot$" denotes element-wise multiplication.

The problem of (8.1) is that it is inadequate to model a two-layered blur. In [60], Jia reported that ringing artifact is generated by a moderately advanced deconvolution algorithm even if the true blur kernel is known. Similar observations are found in [30], suggesting that some fundamental issues are present.

8.1.2 Contributions

The goal of this chapter is to achieve the following objectives.

- Two-layered out-of-focus blur is a specific subset of the general spatially variant deconvolution problem. Dai and Wu [31] proposed a global minimization and used iterative reweighted least squares (IRLS) algorithm to solve the problem. While their approach gives satisfactory results, the computation time is long. Our first objective is to reduce the computation time by exploiting the invariant structures of the problem. Specifically, we show that by inpainting the background, the variant problem can be transformed to an invariant problem.

- Two-layered out-of-focus blur is also a blind deconvolution problem. Existing deconvolution methods [26, 40, 60, 116] are insufficient to estimate the blur kernel in this problem. Our second objective is to propose a new blur kernel estimation method that uses information from both the image content and alpha-matte edges.

8.1.3 Organization

To present our solutions to the two objectives, we organize the chapter as follows. In Section II, we discuss the imaging model for two-layered blur problems. We point out the limitations of (8.1). Then in Section III, we discuss a method to transform the variant blur to invariant blur. In Section IV, we discuss an improved blur kernel estimation method. Experimental results are shown in Section V. Limitations and conclusions are discussed in Section VI.

To clarify the notations used in this chapter, a list of symbols is given in Table 8.1.

Table 8.1: List of Symbols used in this chapter

$\mathbf{g}$	observed blurred image	α	unknown alpha-matte
$\mathbf{g}_I$, $\mathbf{f}_I$	cropped interior region of $\mathbf{g}$, $\mathbf{f}$	α_0	initial estimate of α
$\mathbf{g}_F$, $\mathbf{f}_F$	foreground component of $\mathbf{g}$, $\mathbf{f}$	$\mathbf{f}$	unknown image
$\mathbf{g}_B$, $\mathbf{f}_B$	background component of $\mathbf{g}$, $\mathbf{f}$	$\nabla\mathbf{f}$, $\nabla\mathbf{g}$	image gradient of $\mathbf{f}$, $\mathbf{g}$
$\widetilde{\mathbf{g}}$	invariant image for deconvolution	$\mathbf{f}_S$, $\mathbf{g}_S$, α_{0S}	shock filter of $\mathbf{f}$, $\mathbf{g}$, α_0
$\mathbf{h}$	unknown blur kernel	$\mathbf{f}_{IS}$, $\mathbf{g}_{IS}$	interior region of $\mathbf{f}_S$, $\mathbf{g}_S$
λ, γ, μ	regularization parameters	$\nabla^s\mathbf{f}$, $\nabla^s\mathbf{g}$	strong gradients of $\mathbf{f}$, $\mathbf{g}$
$\widehat{\mathbf{f}}_B$	estimated background	$\Delta\widehat{\mathbf{f}}_B$	estimation error $\mathbf{f}_B - \widehat{\mathbf{f}}_B$
η_g, η_α	noise	σ_g^2, σ_α^2	noise variance

8.2 Imaging Model

The imaging model of a two-layered blur is the foundation of all subsequent analysis discussed in this chapter. Therefore, in this section, we provide justifications to our model and point out insufficiency of the classical model. Our model has been previously used in [4, 31, 75]. A rigorous treatment of the subject is given in [4].

8.2.1 Limitation of Classical Model

To understand the limitation of (8.1), we consider the formation of a two-layered image consisting of a sharp foreground and blurred background. The image formed according to (8.1) is

$$\mathbf{g} = \alpha \cdot (\mathbf{h}_F * \mathbf{f}) + (1 - \alpha) \cdot (\mathbf{h}_B * \mathbf{f}), \tag{8.2}$$

where $\mathbf{h}_F$ is the δ-function, $\mathbf{h}_B$ is the blur for the background, and α is the alpha-matte that indicates the location of foreground pixels [110]. (8.2) suggests that the image $\mathbf{f}$ is first blurred using $\mathbf{h}_F$ and $\mathbf{h}_B$ individually, and linearly combined using α.

If the classical model *were* valid for the formation of a sharp foreground and blurred background, then one should be able to recover the image (reasonably well) by using methods such as [19], [60] or [30]. However, even with a good estimate of the blur kernel and a fine-tuned algorithm, ringing artifacts still appear at the foreground object boundary as shown in the left image of Fig. 8.1.

To further illustrate the problem of the classical model, we synthesize a blurred image using (8.2). The right image of Fig. 8.1 is a simulation of (8.2) in an extreme situation where $\mathbf{h}_F$ is the δ-function and $\mathbf{h}_B$ is a "disk" function with large radius. Unwanted color bleeding is observed around the object boundary, which is wrong because the foreground color should not contribute to the background blur.

Figure 8.1: Limitation of (8.1). Left: Result of spatially variant TV minimization [19]. Right: Simulation of (8.2) with $\mathbf{h}_F$ being a delta function and $\mathbf{h}_B$ being a "disk" function with large radius.

8.2.2 Our Model

Suggested by McGuire *et al.* [75], the image $\mathbf{f}$ can be expressed as $\mathbf{f} = \alpha\mathbf{f}_F + (1-\alpha)\mathbf{f}_B$, where $\mathbf{f}_F$ denotes the foreground and $\mathbf{f}_B$ denotes the background. Consequently, the observed image is

$$\mathbf{g} = \mathbf{h}_F * (\alpha \cdot \mathbf{f}_F) + (1 - \alpha * \mathbf{h}_F) \cdot (\mathbf{h}_B * \mathbf{f}_B). \tag{8.3}$$

Note that the order of convolution "$*$" and the element-wise multiplication "$\cdot$" cannot be switched. Also, it is assumed that $\mathbf{h}_F$ has a small support[1].

Observing (8.3), we find that there is no cross convolution terms such as $\mathbf{h}_F * \mathbf{f}_B$ or $\mathbf{h}_B * \mathbf{f}_F$, which would appear in (8.2) if we substitute $\mathbf{f} = \alpha \cdot \mathbf{f}_F + (1 - \alpha) \cdot \mathbf{f}_B$. For

[1] See [4] for the assumptions made about the solid angle.

example, for the case where $\mathbf{h}_F$ is the δ-function, our model (8.3) implies

$$\mathbf{g} = (\alpha \cdot \mathbf{f}_F) + (1 - \alpha) \cdot (\mathbf{h}_B * \mathbf{f}_B), \tag{8.4}$$

whereas Nagy and O'leary's model (8.2) implies

$$\begin{aligned} \mathbf{g} &= \alpha \cdot \mathbf{f} + (1 - \alpha) \cdot (\mathbf{h}_B * \mathbf{f}) \\ &= \alpha \cdot (\alpha \cdot \mathbf{f}_F + (1 - \alpha) \cdot \mathbf{f}_B) \\ &\quad + (1 - \alpha) \cdot (\mathbf{h}_B * (\alpha \cdot \mathbf{f}_F + (1 - \alpha) \cdot \mathbf{f}_B)) \\ &= \alpha \cdot \mathbf{f}_F + (1 - \alpha) \cdot (\mathbf{h}_B * (\alpha \cdot \underline{\mathbf{f}_F} + (1 - \alpha) \cdot \mathbf{f}_B)), \end{aligned} \tag{8.5}$$

where we used the facts $\alpha \cdot \alpha = \alpha$ and $\alpha \cdot (1 - \alpha) = 0$. It can be seen that (8.4) and (8.5) coincide if the underlined term $\mathbf{f}_F$ in (8.5) is replaced by $\mathbf{f}_B$, which suggests that the cross convolution term $\mathbf{h}_B * \alpha\mathbf{f}_F$ causes the color bleeding shown in Fig. 8.1.

8.3 Exploiting Invariant Structures

By definition of a two-layered out-of-focus blur problem, the observed image consists of two regions in which each region is homogeneously blurred by a blur kernel. Assuming that the blur kernels are known (estimation of kernels are discussed in Section IV), recovering for each region is a classical invariant deconvolution problem. However, the question is how to partition a variant problem into two invariant sub-problems.

8.3.1 Background Blur / Foreground Sharp Case

A background blur is characterized by setting $\mathbf{h}_F = \delta$-function in (8.3) so that the observed image is given in (8.4). With the assumption that the ground truth alpha-matte α is available, the foreground component $\alpha \cdot \mathbf{f}_F$ equals to $\alpha \cdot \mathbf{g}$, and hence $\mathbf{g} - \alpha \cdot \mathbf{f}_F = (1 - \alpha) \cdot \mathbf{g}$. Since it is also true that $\mathbf{g} - \alpha \cdot \mathbf{f}_F = (1 - \alpha) \cdot (\mathbf{h}_B * \mathbf{f}_B)$, we have

$$(1 - \alpha) \cdot \mathbf{g} = (1 - \alpha) \cdot (\mathbf{h}_B * \mathbf{f}_B). \tag{8.6}$$

Solving (8.6) would be a standard deconvolution problem if the term $(1 - \alpha)$ were invertible, which is not possible due to the binary nature of α. However, the singularity

of $(1-\alpha)$ implies the existence of many $\widetilde{\mathbf{g}}$ such that $\widetilde{\mathbf{g}} \neq \mathbf{g}$ but $(1-\alpha)\cdot\widetilde{\mathbf{g}} = (1-\alpha)\cdot\mathbf{g}$. In fact, any $\widetilde{\mathbf{g}}$ in the following form would satisfy (8.6):

$$\widetilde{\mathbf{g}}(i,j) = \begin{cases} (\mathbf{h}_B * \mathbf{f}_B)(i,j), & \text{if } (i,j) \in \Omega_B, \\ \text{any function } \psi(i,j), & \text{if } (i,j) \in \Omega_F, \end{cases}$$

where $\widetilde{\mathbf{g}}(i,j)$ denotes the (i,j)-th pixel of $\widetilde{\mathbf{g}}$, $\Omega_B = \{(i,j)\,|\,\alpha(i,j) = 0\}$ is the set of background pixel coordinates, and $\Omega_F = \{(i,j)\,|\,\alpha(i,j) = 1\}$ is the set of foreground pixel coordinates. The arbitrary function $\psi(i,j)$ is chosen such that the deconvolution problem $\widetilde{\mathbf{g}} = \mathbf{h}_B * \mathbf{f}_B$ yields good results. Choosing a meaningful $\psi(i,j)$ is equivalent to *inpainting* the pixels for Ω_F.

A naive choice of $\psi(i,j)$ is that $\psi(i,j) = 0$ for any $(i,j) \in \Omega_F$, which means no inpainting is performed. As one can expect, ringing artifacts will appear. Shown in Fig. 8.2 is an example where we use $\mathbf{h}_B =$ Gaussian blur kernel ($\sigma = 2$). The deconvolution is performed using a regularized Wiener filter.

Figure 8.2: [Left] The background component with occluded region unfilled (i.e., $\psi(i,j) = 0$). [Right] Deblurring result of the left image.

Clearly, to reduce oscillation, we must fill Ω_F carefully so that the transient between Ω_F and Ω_B is smooth. Thinking in the one-dimension case, filling Ω_F is equivalent to extrapolating a discrete-time signal $g[n]$ for $n \geq 0$, with known values of $g[n]$ for $n < 0$. The smoothness criteria can be translated to requiring $g'[n] = g'[n-1]$, where g' is the derivative. The condition $g'[n] = g'[n-1]$ means that the slope at $g[n]$ should be the same as the slope at $g[n-1]$.

The condition $g'[n] = g'[n-1]$ has a problem that it leads to unbounded prediction, because if $g'[n-1] > 0$, then $g[n] \to \infty$ as $n \to \infty$. To ensure boundedness, instead of using $g'[n] = g'[n-1]$, we require $g'[n] = \frac{1}{n}g'[n-1]$ for $n > 0$, and $g'[n] = g'[n-1]$

for $n = 0$. Consequently, the recursion is defined as

$$g[n] = \left(1 + \frac{1}{n}\right) g[n-1] - \frac{1}{n} g[n-2] \quad \text{for } n > 0, \tag{8.7}$$

with the initial condition $g[0] = 2g[-1] - g[-2]$. Intuitively, this recursion forces the slope at every extrapolation location to be reduce by a factor depending on the physical distance from the object boundary. The derivation of (8.7) and a proof of the boundedness are given in Appendix D.

Extending the idea to the two-dimensional setting, we want the gradient of $\widetilde{\mathbf{g}}$ at pixel (i, j) to be similar to the gradients in its neighborhood. Since the two-dimensional gradient is directional, there are multiple equations for predicting $\widetilde{\mathbf{g}}(i, j)$:

$$\widetilde{\mathbf{g}}(i, j) - \widetilde{\mathbf{g}}(i + p, j + q) = \widetilde{\mathbf{g}}(i + p, j + q) - \widetilde{\mathbf{g}}(i + 2p, j + 2q), \tag{8.8}$$

where $p = q = \{-1, 0, 1\}$. Therefore, to maintain the smoothness condition we must find a $\widetilde{\mathbf{g}}(i, j)$ such that it is the best fit to all of its neighboring gradients. Denote by $\mathcal{A}$ the set of non-zero pixels of neighborhood of $\widetilde{\mathbf{g}}(i, j)$:

$$\mathcal{A} = \{(p, q) \,|\, \widetilde{\mathbf{g}}(i + p, i + q) \neq 0, |p| \leq 1, |q| \leq 1\},$$

we solve the following minimization problem

$$\underset{\widetilde{\mathbf{g}}(i,j)}{\text{minimize}} \sum_{(p,q)\in\mathcal{A}} (\widetilde{\mathbf{g}}(i, j) - 2\widetilde{\mathbf{g}}(i + p, i + q) + \widetilde{\mathbf{g}}(i + 2p, i + 2q))^2,$$

of which the solution can be found by considering the first order optimality, yielding

$$\widetilde{\mathbf{g}}(i, j) = \frac{1}{|\mathcal{A}|} \sum_{(p,q)\in\mathcal{A}} 2\widetilde{\mathbf{g}}(i + p, i + q) - \widetilde{\mathbf{g}}(i + 2p, i + 2q).$$

Incorporating the idea of diminishing gradient so that $\widetilde{\mathbf{g}}(i, j)$ is bounded, we have

$$\widetilde{\mathbf{g}}(i, j) = \frac{1}{|\mathcal{A}|} \sum_{(p,q)\in\mathcal{A}} \left(1 + \frac{1}{k}\right) \widetilde{\mathbf{g}}(i + p, i + q) - \frac{1}{k}\widetilde{\mathbf{g}}(i + 2p, i + 2q),$$

where k is the shortest distance from the unknown pixel (i, j) to the known set Ω_B. The proposed algorithm for filling Ω_F is shown in Algorithm 10(Inward Version). The overall

steps for removing background blur is given in Algorithm 11.

Algorithm 10 Proposed Inpainting Algorithm

Given $\widetilde{\mathbf{g}}$, Ω_F and Ω_B.
// Outward Version
Partition Ω_F into K rings from the outermost to the innermost, with each ring one pixel in width.
// Inward Version
Partition Ω_B into K rings from the innermost to the outermost, with each ring one pixel in width.
for $k = 1 : K$ **do**
  for each (i, j) on the k-th ring **do**
    Determine $\mathcal{A} = \{(p, q) \,|\, \widetilde{\mathbf{g}}(i + p, i + q) \neq 0, |p| \leq 1, |q| \leq 1\}$.
    Find

$$\widetilde{\mathbf{g}}(i, j) = \frac{1}{|\mathcal{A}|} \sum_{(p,q)\in\mathcal{A}} \left(1 + \frac{1}{k}\right) \widetilde{\mathbf{g}}(i + p, i + q) - \frac{1}{k}\widetilde{\mathbf{g}}(i + 2p, i + 2q),$$

  end for
end for

Algorithm 11 Recovering from background blur

Given $\mathbf{g}$. Estimate $\mathbf{h}_B$ and α (see Section IV).
Estimate $\widetilde{\mathbf{g}}$ using Algorithm 10.
Solve the deconvolution $\widetilde{\mathbf{g}} = \mathbf{h}_B * \mathbf{f}_B$.
Form the solution $\mathbf{f} = \alpha \cdot \mathbf{g} + (1 - \alpha) \cdot \mathbf{f}_B$.

Algorithm 10 can be modified to take into account of more neighboring pixels, e.g, 5×5 or 7×7. In this case, the footstep for the forward difference operation is increased, and (8.8) will consist of more equations.

It is also interesting to note that asymptotically Algorithm 10 becomes a moving average when $k \to \infty$, because $\widetilde{\mathbf{g}}(i, j) \to \frac{1}{|\mathcal{A}|} \sum_{(p,q)\in\mathcal{A}} \widetilde{\mathbf{g}}(i + p, i + q)$. In fact, except for small k, experimentally we find the difference between Algorithm 10 and the simple moving average is almost undetectable visually. Fig. 8.3 shows the result of applying Algorithm 10 to fill Ω_F. Here, the neighborhood size is 5×5, and we used moving average to approximate Algorithm 10.

As shown in Fig. 8.3, the central region of $\widetilde{\mathbf{g}}$ does not seem visually pleasing. However, we argue that the goal is to fill Ω_F so that there are less ringing artifacts for the deconvolution step. As a comparison, we applied a state-of-the-art exemplar-based inpainting algorithm by Criminisi *et at.* [29]. Shown in Fig. 8.4 are the inpainting

Figure 8.3: Filling Ω_F for Image No.5 . From Left to Right: The intermediate result of inpainting at iteration 0, 20, 40 and final respectively. Left: Ω_F. When inpainting starts, the algorithm fills the occluded region from outside to inside. Right: the inpainted $\widetilde{\mathbf{g}}$.

(a) Proposed inpainting method (b) Inpainting method by Criminisi et al. [29]

Figure 8.4: Comparisons between the proposed inpainting method and the exemplar-based inpainting method by Criminisi et al. [29]. Top: inpainting results and zoom-in. Bottom: deblurring results and zoom-in.

results of Algorithm 10 and the method by Criminisi *et al.*. In terms of visual quality, it is clear that the method by Criminisi *et al.* is significantly better than Algorithm 10. However, since [29] does not impose smoothness criteria at the boundary, ringing artifacts are present. In contrast, the smoothness condition at the boundary of Ω_F is explicitly enforced by Algorithm 10. There are artifacts occurring in the center part of Ω_F, but these can be covered by $\mathbf{g}_F$ as they are far from the boundary. Note also that the computational complexity of Algorithm 10 is significantly lower than the method by Criminisi *et al.*.

8.3.2 Foreground Blur / Background Sharp Case

In the case of foreground blur, we set $\mathbf{h}_B = \delta$-function in (8.3). Thus, the observed image is

$$\mathbf{g} = \mathbf{h}_F * (\alpha \cdot \mathbf{f}_F) + (1 - \alpha * \mathbf{h}_F) \cdot \mathbf{f}_B. \tag{8.9}$$

Rearranging the terms we have

$$\begin{aligned} \mathbf{g}_F &= \mathbf{g} - (1 - \alpha * \mathbf{h}_F) \cdot \mathbf{f}_B \\ &= \mathbf{h}_F * (\alpha \cdot \mathbf{f}_F). \end{aligned} \tag{8.10}$$

Therefore, assuming that α, $\mathbf{h}_F$ are correctly estimated, solving for $\alpha \cdot \mathbf{f}_F$ from $\mathbf{g}_F$ in (8.10) is a standard deconvolution once $\mathbf{f}_B$ is known. However, $\mathbf{f}_B$ is never known exactly because part of $\mathbf{f}_B$ is occluded by the blurring edge of $\mathbf{f}_F$ [2]. Thus, given an estimate $\widehat{\mathbf{f}}_B$, for example using Algorithm 10, there is an error term $\Delta\mathbf{f}_B$ so that

$$\widehat{\mathbf{f}}_B = \mathbf{f}_B + \Delta\widehat{\mathbf{f}}_B. \tag{8.11}$$

Substituting (8.11) into (8.10) yields

$$\begin{aligned} \mathbf{g}_F &= \mathbf{g} - (1 - \alpha * \mathbf{h}_F) \cdot (\widehat{\mathbf{f}}_B - \Delta\widehat{\mathbf{f}}_B) \\ &= \mathbf{g} - (1 - \alpha * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B + (1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B. \end{aligned} \tag{8.12}$$

In (8.12), only $\mathbf{g}$ and $(1 - \alpha * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B$ can be calculated.

(a) (b) (c)

Figure 8.5: (a) Input blurred image $\mathbf{g}$. (b) Inpainted result $\widetilde{\mathbf{g}}$ using inpainting method 1. (c) Inpainted result $\widetilde{\mathbf{g}}$ using inpainting method 2

[2] The central interior region of $\mathbf{f}_F$ is not important because it is the same as the central interior region of $\mathbf{g}$.

(a) (b) (c) (d)

Figure 8.6: Illustration of (8.16) using inpainting method 2. (a) The observed image extracted by alpha-matte $(\alpha * \mathbf{h}_F) \cdot \mathbf{g}$. (b) Subtract the result of (a) with the estimated background $(\alpha * \mathbf{h}_F) \cdot \left[\mathbf{g} - (1 - \alpha * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B\right]$. (c) Add the result of (b) with the newly inpainted background $(\alpha * \mathbf{h}_F) \cdot \left[\mathbf{g} - (1 - \alpha * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B + (1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B\right]$. (d) Add the result of (c) with an approximation from the object boundary $(\alpha * \mathbf{h}_F) \cdot \left[\mathbf{g} - (1 - \alpha * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B + (1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B\right] + (1 - \alpha * \mathbf{h}_F) \cdot \overline{\mathbf{g}}_B$.

Multiplying $\alpha * \mathbf{h}_F$ to both sides of (8.12) yields

$$(\alpha * \mathbf{h}_F) \cdot [\mathbf{h}_F * (\alpha \cdot \mathbf{f}_F)] = (\alpha * \mathbf{h}_F) \cdot \left[\mathbf{g} - (1 - \alpha * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B + (1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B\right]. \tag{8.13}$$

It is not difficult to show that (See Appendix D)

$$\|(\alpha * \mathbf{h}_F) \cdot (1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B\| \leq \|(1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B\|,$$

implying that the effect of $\Delta\widehat{\mathbf{f}}_B$ is reduced by multiplying $(\alpha * \mathbf{h}_F)$. In fact, approximating $(\alpha * \mathbf{h}_F) \cdot (1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B$ is significantly easier than approximating $(1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B$, because the non-zero entries of $(\alpha * \mathbf{h}_F) \cdot (1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B$ are confined to the ring $(\alpha * \mathbf{h}_F) \cdot (1 - \alpha * \mathbf{h}_F)$ around the object boundary, whereas the non-zero entries of $(1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B$ compose the entire foreground object.

Now, two issues remain: (i) We need an approximation for $(\alpha * \mathbf{h}_F) \cdot (1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B$ because $\Delta\widehat{\mathbf{f}}_B$ is not known; (ii) Given the approximation and hence the right hand side of (8.13), we need to solve for $\alpha \cdot \mathbf{f}_F$.

Let us consider the second question first. Given $\mathbf{g}_F$, we want to solve $\alpha \cdot \mathbf{f}_F$ from the equation

$$(\alpha * \mathbf{h}_F) \cdot \mathbf{g}_F = (\alpha * \mathbf{h}_F) \cdot [\mathbf{h}_F * (\alpha \cdot \mathbf{f}_F)]. \tag{8.14}$$

Solving (8.14) is similar to solving (8.6), which both requires inpainting and deconvo-

lution. However, (8.14) is more difficult than solving (8.6) because $\alpha * \mathbf{h}_F$ is not a binary mask. Here we propose two inpainting strategies. The first strategy inpaints the background using the background color, which can be accomplished using Algorithm 10 (inpainting *inwards*). The second strategy inpaints the background using the foreground color, which can also be accomplished using Algorithm 10 (inpainting *outwards*). In both methods, since there is no sharp cut off between foreground and background, the algorithm starts from some definite background (or foreground) pixels. In our method, we start from K pixels from the expected object boundary (estimated from α), where K is the one-sided width of the blur kernel.

Denoting $\overline{\mathbf{g}}_B$ the inpainted background, $\Omega_M = \{(i,j)\,|\,0 < (\alpha * \mathbf{h}_F)(i,j) < 1\}$, $\Omega_F = \{(i,j)\,|\,(\alpha * \mathbf{h}_F)(i,j) = 1\}$, and $\Omega_B = \{(i,j)\,|\,(\alpha * \mathbf{h}_F)(i,j) = 0\}$, we construct an approximately invariant blur image

$$\widetilde{\mathbf{g}}(i,j) = \begin{cases} \mathbf{g}_F(i,j), & (i,j) \in \Omega_F, \\ \big[(\alpha * \mathbf{h}_F) \cdot \mathbf{g}_F + (1 - \alpha * \mathbf{h}_F) \cdot \overline{\mathbf{g}}_B\big](i,j), & (i,j) \in \Omega_M, \\ \overline{\mathbf{g}}_B(i,j), & (i,j) \in \Omega_B. \end{cases} \tag{8.15}$$

Based on (8.15), we propose an approximation for $(\alpha * \mathbf{h}_F) \cdot (1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B$. Substituting (8.12) to the Ω_M case of (8.15), (8.15) becomes

$$\begin{aligned} \widetilde{\mathbf{g}} = & \underbrace{(\alpha * \mathbf{h}_F) \cdot \mathbf{g}}_{\text{foreground with edge residue}} - \underbrace{(\alpha * \mathbf{h}_F) \cdot (1 - \alpha * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B}_{\approx\text{edge residue}} \\ & + \underbrace{(1 - \alpha * \mathbf{h}_F) \cdot \overline{\mathbf{g}}_B}_{\text{inpainted background}} + (\alpha * \mathbf{h}_F) \cdot (1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B. \end{aligned} \tag{8.16}$$

The four terms in (8.16) have individual meaning: $(\alpha * \mathbf{h}_F) \cdot \mathbf{g}$ is the foreground component; $(\alpha * \mathbf{h}_F) \cdot (1 - \alpha * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B$ is the foreground edge residue remaining in $(\alpha * \mathbf{h}_F) \cdot \mathbf{g}$; $(1 - \alpha * \mathbf{h}_F) \cdot \overline{\mathbf{g}}_B$ is the background component to be added. The sum of the first three terms is an image with a dark ring around the object boundary, because excessive boundary intensity is subtracted by $(\alpha * \mathbf{h}_F) \cdot (1 - \alpha * \mathbf{h}_F) \cdot \widehat{\mathbf{f}}_B$. Therefore, the fourth term must compensate for the presence of the dark ring. Hence, we *choose*

$$(\alpha * \mathbf{h}_F) \cdot (1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B = (\alpha * \mathbf{h}_F) \cdot (1 - \alpha * \mathbf{h}_F) \cdot \overline{\mathbf{g}}_B.$$

The results are shown in Figs. 8.5 and 8.6.

Finally, given the right hand side of (8.16), we solve the deconvolution problem

$$\widetilde{\mathbf{g}} = \mathbf{h}_F * \widetilde{\mathbf{f}}.$$

The solution $\widetilde{\mathbf{f}}$ is then combined with the estimated background $\widehat{\mathbf{f}}_B$ to give the final solution

$$\mathbf{f} = \alpha \cdot \widetilde{\mathbf{f}} + (1 - \alpha) \cdot \widehat{\mathbf{f}}_B.$$

Algorithm 12 Recovering from background blur

Given $\mathbf{g}$. Estimate $\mathbf{h}_F$ and α (see Section IV).
Estimate $\widehat{\mathbf{f}}_B$ using Algorithm 10.
Estimate $\widetilde{\mathbf{g}}$ by (8.15).
Solve the deconvolution $\widetilde{\mathbf{g}} = \mathbf{h}_F * \widetilde{\mathbf{f}}$.
Form the solution $\mathbf{f} = \alpha \cdot \widetilde{\mathbf{f}} + (1 - \alpha) \cdot \widehat{\mathbf{f}}_B$.

8.3.3 Performance Limit

The proposed spatially invariant method is a trade-off between quality and speed. The limiting factor is the amount of blur and complexity of the occluded region, which is also a fundamental limit of the blurring equation (8.3) by McGuire *et al.* [75]. For the background blur case, the inpainting error is small because the object boundary is sharp. For the foreground blur case, the approximation error in $(\alpha * \mathbf{h}_F) \cdot (1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B$ increases when the image is complicated or when the blur is severe.

8.4 Blur Kernel Estimation

In this section we discuss the proposed kernel estimation. Our method is an improved version of Xu and Jia [116], Fergus *et al.* [40] and Cho and Lee [26]. Therefore, the focus of this section is on the unique modifications that we make.

The overall deblurring method consists of four parts, namely the initial alpha-matte estimation, blur kernel estimation, transformation to invariant problems, and finally deblurring. Initial alpha-matte estimation is performed using shared matting [44], which is currently the top-ranked algorithm in [89]. A thorough survey on alpha-matting methods can be found in [110]. The transformation of spatially variant problems to

invariant problems is discussed in Section III. In the following subsections, we discuss the blur kernel estimation and the deblurring steps.

8.4.1 Kernel Estimation by Strong Edges

Discussed in [40], blur kernels can be efficiently estimated using edges. The motivation is that a blurred image is the result of convolving a sharp image and the blur kernel. Thus, if one wants to estimate the blur kernel from a blurred image, a rough estimate of the sharp image must be used.

To obtain such a sharp image estimate, Cho and Lee [26] use the shock filter [95]. A shock is an iterative algorithm that sharpens the strong edges (see Appendix). We consider the shock filter as a module that takes an image $\mathbf{f}$ and produces an output image $\mathbf{f}_S$:

$$\mathbf{f}_S = \text{shock filter}(\mathbf{f}).$$

The estimation considered in [26] is the following minimization problem

$$\underset{\mathbf{h}}{\text{minimize}} \ \|\nabla \mathbf{f}_S * \mathbf{h} - \nabla \mathbf{g}\|^2 + \lambda \|\mathbf{h}\|^2, \tag{8.17}$$

where ∇ is the gradient operator, λ is a regularization parameter.

Later, Xu and Jia [116] observe that not all gradients of $\mathbf{f}_S$ are useful for blur kernel estimation. They propose an edge selection method to decide what edges shall be used. Essentially, the idea is to multiply a mask $\mathbf{M}$ to the image gradient $\nabla \mathbf{f}_S$ (see Appendix). We consider this edge selection process as a module that takes a shock filtered image $\mathbf{f}_S$ to produce an output image $\nabla^s \mathbf{f}_S$:

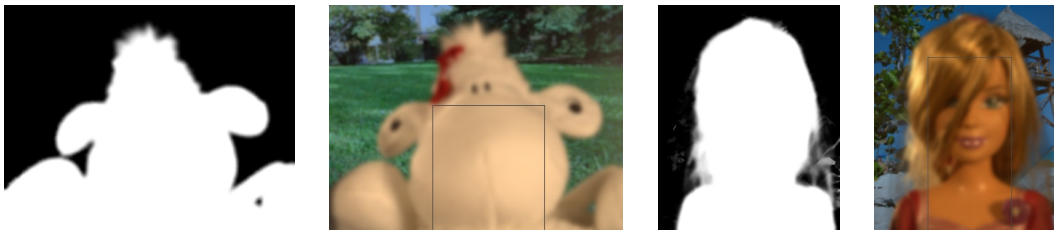
$$\nabla^s \mathbf{f}_S = \text{edge selection}(\mathbf{f}_S) = \mathbf{M} \cdot \nabla \mathbf{f}_S.$$

Therefore, the estimation becomes

$$\underset{\mathbf{h}}{\text{minimize}} \ \|\nabla^s \mathbf{f}_S * \mathbf{h} - \nabla^s \mathbf{g}\|^2 + \lambda \|\mathbf{h}\|^2. \tag{8.18}$$

8.4.2 Blur Information from Boundary and Interior of an Object

A limitation of (8.18) is that it depends on the availability of strong edges in an image. In a two-layered blur image, it is possible that the foreground or background



(a) (b) (c) (d)

Figure 8.7: (a)-(b): An example where there is no strong edge in the cropped area, but the alpha-matte estimate is good. In this case, large γ can be used. (c)-(d): An example where the alpha-matte is not well estimated, but there is strong edge in the cropped area. In this case, large μ shall be used.

component does not have strong edges. For example, Fig. 8.7(b) shows an image where the cropped area of the foreground component does not have significant color variation.

In [60], Jia shows that for a two-layered blur problem, the alpha-matte provides useful information for blur kernel estimation because the alpha-matte is also blurred by the same blur kernel. Therefore, by studying the transient characteristics of the alpha-matte, blur kernel can be estimated. For example in Fig. 8.7(a), while the cropped region in (b) does not have strong edges, the alpha-matte does. The estimation using alpha-matte is similar to (8.18):

$$\underset{\mathbf{h}}{\text{minimize}} \ \|\nabla\alpha_{0S} * \mathbf{h} - \nabla\alpha_0\|^2 + \lambda\|\mathbf{h}\|^2, \tag{8.19}$$

where α_0 is the initial estimate of the alpha-matte, and $\alpha_{0S} = \text{shock filter}(\alpha_0)$.

The observation that alpha-matte can be used when edges in the foreground region are weak suggests that reversely the strong edges can be used when alpha-mattes are inaccurate. Show in Fig. 8.7(c)-(d) is an example where the alpha-matte of the hair regions cannot be estimated correctly, but there are strong edges in the cropped region. This idea illustrates a unique feature of the proposed method, which encapsulates the strengths of both strong edge and alpha-matte based approaches. In the following subsection, we discuss more precisely the mathematical formulation.

8.4.3 Estimation of h using g and α (Foreground Blur)

We first focus on the foreground blur situation. The proposed kernel estimation is formulated in the maximum-a-posteriori (MAP) framework. Denoting by $\mathbf{g}_I$ the cropped

interior region of the input image and α_0 the initial estimated alpha-matte, respectively, the goal is to maximize the conditional probability $P(\mathbf{h}|\mathbf{g}_I, \alpha_0)$, which according to Bayes' theorem (with the assumption that $P(\mathbf{g}_I|\ \mathbf{h})$ and $P(\alpha_0|\ \mathbf{h})$ are independent) is equivalent to

$$\underset{\mathbf{h}}{\operatorname{argmax}}\, P(\mathbf{h}|\ \mathbf{g}_I, \alpha_0) = \underset{\mathbf{h}}{\operatorname{argmax}}\, P(\mathbf{g}_I|\mathbf{h})P(\alpha_0|\mathbf{h})P(\mathbf{h}). \tag{8.20}$$

The likelihood $P(\mathbf{g}_I|\mathbf{h})$ defines the probability of getting $\mathbf{g}_I$ given an estimate of $\mathbf{h}$. Following the idea of Fergus *et al.* [40] that $\mathbf{g}_I \approx \mathbf{g}_{IS} * \mathbf{h}$, it holds that $\nabla^s \mathbf{g}_I = \nabla^s \mathbf{g}_{IS} * \mathbf{h} + \eta_g$. The noise term η_g is assumed to be an i.i.d. Gaussian distributed random variable, with zero mean and variance σ_g^2. Thus,

$$P(\mathbf{g}_I|\mathbf{h}) \propto \prod_{i \in \Omega_{\mathbf{g}}} \exp\left\{-\frac{1}{2\sigma_g^2}[\nabla^s \mathbf{g}_{IS} * \mathbf{h} - \nabla^s \mathbf{g}_I]_i^2\right\}, \tag{8.21}$$

where $[\cdot]_i$ denotes the i-th component of the argument and $\Omega_{\mathbf{g}}$ is the set of pixels in $\mathbf{g}_I$.

The likelihood $P(\alpha_0|\mathbf{h})$ is defined similarly, where we assume that $\nabla\alpha_0 = \nabla\alpha_{0S} * \mathbf{h} + \eta_\alpha$ with η_α being an i.i.d. Gaussian distributed random variable with zero mean and variance σ_α^2. This definition has been previously used by Jia [60]. Expressing the likelihood in terms of these terms yields

$$P(\alpha_0|\mathbf{h}) \propto \prod_{i \in \Omega_\alpha} \exp\left\{-\frac{1}{2\sigma_\alpha^2}[\nabla\alpha_{0S} * \mathbf{h} - \nabla\alpha_0]_i^2\right\}, \tag{8.22}$$

where Ω_α denotes the set of pixels in α_0.

The prior $P(\mathbf{h})$ is given by

$$P(\mathbf{h}) \propto \prod_{i \in \Omega_h} \exp\left\{-\lambda[\mathbf{h}]_i^2\right\}, \tag{8.23}$$

where Ω_h denotes the set of pixels in the blur kernel $\mathbf{h}$. $P(\mathbf{h})$ is a relaxation of the exponential prior given by Shan *et al.* [97], as the one-norm results from the exponential prior is computationally intensive to solve.

Substituting (8.21), (8.22) and (8.23) into (8.20), and simplifying the terms yields

$$\underset{\mathbf{h}}{\text{minimize}}\ \ \mu\|\nabla^s \mathbf{g}_{IS} * \mathbf{h} - \nabla^s \mathbf{g}_I\|^2 + \gamma\|\nabla\alpha_{0S} * \mathbf{h} - \nabla\alpha_0\|^2 + \lambda\|\mathbf{h}\|^2, \tag{8.24}$$

where μ, γ and λ are regularization parameters (to be discussed). Analytical solution

for (8.24) exists, and is given by

$$\mathbf{h} = \mathcal{F}^{-1}\left\{\frac{\mu\overline{\mathcal{F}(\nabla^s \mathbf{g}_{IS})}\cdot\mathcal{F}(\nabla^s \mathbf{g}_I) + \gamma\overline{\mathcal{F}(\nabla\alpha_{0S})}\cdot\mathcal{F}(\nabla\alpha_0)}{\mu|\mathcal{F}(\nabla^s \mathbf{g}_{IS})|^2 + \gamma|\mathcal{F}(\nabla\alpha_{0S})|^2 + \lambda}\right\}, \tag{8.25}$$

where $\mathcal{F}$ is the Fourier Transform operator, $\overline{(\cdot)}$ denotes the complex conjugate over the argument and "$\cdot$" is the element-wise multiplication. Though not written explicitly in (8.24), the gradients $\nabla^s \mathbf{g}_{IS}$ (and similarly for $\nabla\alpha_{0S}$) are assumed to contain both horizontal and vertical directions, i.e., $\nabla^s \mathbf{g}_{IS} = [\partial_x^s \mathbf{g}_{IS},\ \partial_y^s \mathbf{g}_{IS}]$. Thus, $\overline{\mathcal{F}(\nabla^s \mathbf{g}_{IS})}\cdot\mathcal{F}(\nabla^s \mathbf{g}_I) = \overline{\mathcal{F}(\partial_x^s \mathbf{g}_{IS})}\cdot\mathcal{F}(\partial_x^s \mathbf{g}_I) + \overline{\mathcal{F}(\partial_y^s \mathbf{g}_{IS})}\cdot\mathcal{F}(\partial_y^s \mathbf{g}_I)$, and $|\mathcal{F}(\nabla^s \mathbf{g}_{IS})|^2 = |\mathcal{F}(\partial_x^s \mathbf{g}_{IS})|^2 + |\mathcal{F}(\partial_y^s \mathbf{g}_{IS})|^2$.

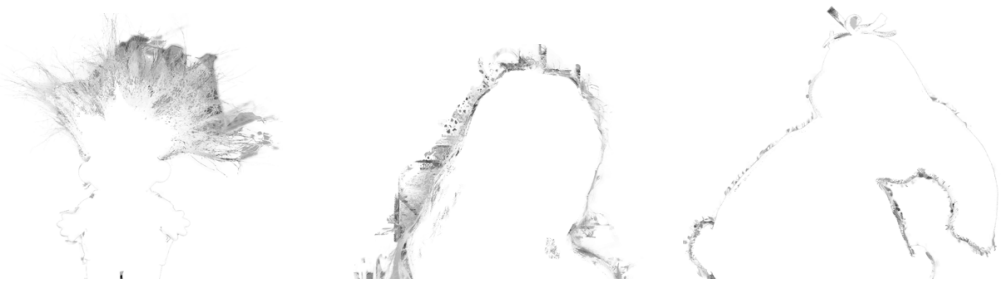
The proposed minimization (8.24) is a generaliation of [40, 60, 97, 116], where these special cases can be obtained by adjusting μ, γ and λ. The advantage of (8.24) is that it eliminates the risk of having weak edges in the cropped foreground area $\mathbf{g}_I$ (which makes [116] and [40] fail), as in this case we can increase γ to make $\|\nabla\alpha_{0S} * \mathbf{h} - \nabla\alpha_0\|^2$ dominant in (8.24). In situations where α_0 is a poor estimate (so that [60] fails), μ can be increased to make $\|\nabla^s \mathbf{g}_{IS} * \mathbf{h} - \nabla^s \mathbf{g}_I\|^2$ dominant.

8.4.4 Choosing Parameters

The next question to ask is how to choose the parameters μ, γ and λ. Without loss of generality we set $\mu = 1$, as the minimizer of (8.24) is unchanged if we scale the objective function in (8.24) by $\frac{1}{\mu}$. Thus it remains to determine γ and λ. λ is the parameter associated with the prior $P(\mathbf{h})$. Typically, meaningful results are found using λ within the range $10^{-3} \leq \lambda \leq 10^{-2}$. In our experiments, λ is fixed at $\lambda = 10^{-2}$.

Choice of γ is a critical one as it sets relative emphasis between $\|\nabla\mathbf{g}_{IS} * \mathbf{h} - \nabla\mathbf{g}_I\|^2$ and $\|\nabla\alpha_{0S} * \mathbf{h} - \nabla\alpha_0\|^2$. Our proposed method is based on the *confidence of* α_0. The confidence measure indicates the chance of getting an accurate alpha-matte. If it is likely that α_0 is a reliable estimate, then the weighing factor γ for $\|\nabla\alpha_{0S} * \mathbf{h} - \nabla\alpha_0\|^2$ should be large. Otherwise, γ should be small.

γ can be evaluated from the performance across different methods. The intuition is that if an image is easy to be alpha-matted, then the results of different alpha-matting methods should be similar. To verify our claim, we conducted an experiment by studying 15 alpha-matting methods available in [89]. Alpha-matting results (small tri-map case,



(a) avg SAD = 18, $\bar{v} = 0.0074$ (b) avg SAD = 10, $\bar{v} = 0.0054$ (c) avg SAD = 9, $\bar{v} = 0.0014$

Figure 8.8: Variance maps **V** for images "troll", "doll", and "elephant". White indicates variance = 0, and black indicates variance = maximized

.

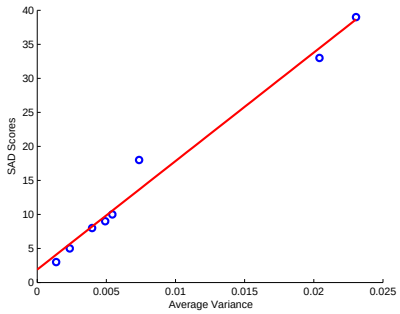


Figure 8.9: Relation between the average SAD score and average variance.

8 images per method), and the corresponding SAD scores are recorded. Discarding the worst performing method, we compute, for each pixel, the variance of alpha-matte values across the methods. That is, given an alpha-matte α^k produced by the k-th method, we compute

$$\mathbf{V}(i,j) = \text{Var}\{\alpha^1(i,j),\, \alpha^2(i,j), \ldots, \alpha^k(i,j)\}.$$

$V(i,j)$ is a map in which each pixel is the variance of the alpha-matting methods (see Fig. 8.8). It is a measure of agreement among the methods, for lower variance means the methods give similar alpha-matte values. Average of the variance over the entire image is then computed as

$$\bar{v} = \frac{1}{n} \sum_{i,j} \mathbf{V}(i,j),$$

where n is the number of pixels in **V**. In Fig. 8.9, we show the relation between the average SAD score and the average variance for the 8 images provided in [89]. It can be

seen that the variance and the SAD score shows a linear relation. This result implies that in the absence of ground-truth, the performance variance between methods is a good indicator of whether the image is easy to be alpha-matted.

The above experiment suggests that for each pixel of α_0, we can use a number of alpha-matting algorithms to measure the confidence. In our algorithm, we use shared matting, grow-cut [107] and closed-form matting [68]. More methods can be considered, but computing time will increase.

Finally, we define γ as

$$\gamma = C \exp\{-a\bar{v}\}, \tag{8.26}$$

where the constants $a = 1000$ and $C = 0.65$ are determined empirically.

8.4.5 Estimation of h using g (Background Blur)

We now discuss the estimation of $\mathbf{h}$ in the case of background blur. In fact, for a background blur/ foreground sharp image, the alpha-matte does not contain information about the blur kernel because the alpha-matte is sharp. Therefore, the MAP framework (8.20) is simplified by removing the term $P(\alpha_0|\mathbf{h})$, which yields

$$\underset{\mathbf{h}}{\operatorname{argmax}}\, P(\mathbf{h}|\ \mathbf{g}_I) = \underset{\mathbf{h}}{\operatorname{argmax}}\, P(\mathbf{g}_I|\mathbf{h})P(\mathbf{h}). \tag{8.27}$$

Consequently, the minimization problem that we consider is

$$\underset{\mathbf{h}}{\text{minimize}}\ \ \mu\|\nabla^s \mathbf{g}_{IS} * \mathbf{h} - \nabla^s \mathbf{g}_I\|^2 + \lambda\|\mathbf{h}\|^2. \tag{8.28}$$

Closed-form solution exists, and is given by (8.25) with $\gamma = 0$.

8.4.6 Iterative Update of f and h

Our problem is a blind deconvolution problem. Therefore, intermediate update of the solution $\mathbf{f}_I$ is required for the iterative update of $\mathbf{h}$. Given the current estimate $\mathbf{h}$, $\mathbf{f}_I$ is updated by solving the following minimization problem

$$\mathbf{f}_I = \underset{\mathbf{f}_I}{\operatorname{argmin}}\ \ \|\mathbf{f}_I * \mathbf{h} - \mathbf{g}_I\|^2 + \kappa\|\nabla\mathbf{f}_I - \nabla^s \mathbf{g}_{IS}\|^2. \tag{8.29}$$

(8.29) is derived from an MAP framework consisting of a Gaussian fidelity term and a Gaussian prior. The Gaussian prior $\|\nabla \mathbf{f}_I - \nabla^s \mathbf{g}_{IS}\|^2$ measures the goodness of fit between the gradient of the unknown image $\mathbf{f}_I$ and the gradient of the shock filtered image $\mathbf{g}_{IS}$. The Gaussian prior $\|\nabla \mathbf{f}_I - \nabla^s \mathbf{g}_{IS}\|^2$ performs better than $\|\nabla \mathbf{f}_I\|^2$ in preserving edges and suppressing ringing artifacts [116]. The parameter κ is fixed at $\kappa = 10^{-2}$.

Closed-form solution of (8.29) exists and is given by

$$\mathbf{f}_I = \mathcal{F}^{-1}\left\{ \frac{\overline{\mathcal{F}(\mathbf{h})}\mathcal{F}(\mathbf{g}_I) + \kappa\left[\overline{\mathcal{F}(\partial_x)}\mathcal{F}(\nabla_x^s \mathbf{g}_{IS}) + \overline{\mathcal{F}(\partial_y)}\mathcal{F}(\nabla_y^s \mathbf{g}_{IS})\right]}{|\mathcal{F}(\mathbf{h})|^2 + \kappa(|\mathcal{F}(\partial_x)|^2 + |\mathcal{F}(\partial_y)|^2)} \right\}, \tag{8.30}$$

where $\partial_x = [1,\ -1]$ and $\partial_y = [1,\ -1]^T$ are two-tap filters.

The updated solution $\mathbf{f}_I$ is then feedback to (8.24) by applying shock filter $\mathbf{f}_{IS} =$ Shock Filter($\mathbf{f}$) and replaces $\mathbf{g}_{IS}$ using $\mathbf{f}_{IS}$ in (8.24). Therefore, the iterative update of $\mathbf{f}_I$ and $\mathbf{h}$ is equivalent to solving

$$\begin{aligned}
\mathbf{f}_I &= \underset{\mathbf{f}_I}{\operatorname{argmin}} \ \ \|\mathbf{f}_I * \mathbf{h} - \mathbf{g}_I\|^2 + \kappa\|\nabla \mathbf{f}_I - \nabla^s \mathbf{f}_{IS}\|^2, \qquad \mathbf{f}_{IS} = \text{Shock Filter}(\mathbf{f}_I), \\
\mathbf{h} &= \underset{\mathbf{h}}{\operatorname{argmin}} \ \ \mu\|\nabla^s \mathbf{f}_{IS} * \mathbf{h} - \nabla^s \mathbf{g}_I\|^2 + \gamma\|\nabla \alpha_{0S} * \mathbf{h} - \nabla \alpha_0\|^2 + \lambda\|\mathbf{h}\|^2.
\end{aligned}$$

The iteration repeats, until the relative change $\|\mathbf{f}_I^{(k+1)} - \mathbf{f}_I^{(k)}\|_2 / \|\mathbf{f}_I^{(k)}\|_2 \leq 10^{-3}$, where $\mathbf{f}_I^{(k)}$ is the solution in (8.30) at the k-th iteration.

8.4.7 Updating α

In case of foreground blur, α_0 is a blurred alpha-matte which needs to be deblurred. To deblur α_0, we consider the posterior probability $P(\alpha|\alpha_0)$, where the unknown (potentially sharp) α is conditional on the blurry observation α_0. Maximizing $P(\alpha|\alpha_0)$ is equivalent to

$$\underset{\alpha}{\operatorname{argmax}}\, P(\alpha|\alpha_0) = \underset{\alpha}{\operatorname{argmax}}\, P(\alpha_0|\alpha)P(\alpha). \tag{8.31}$$

The likelihood $P(\alpha_0|\alpha)$ and the prior $P(\alpha)$ are defined as follows.

We assume that the residue $\mathbf{h} * \alpha - \alpha_0$ is Gaussian distributed with zero mean and variance σ_α^2. Thus, the likelihood $P(\alpha_0|\alpha)$ is

$$P(\alpha_0|\alpha) \propto \prod_{i \in \Omega_\alpha} \exp\left\{ \frac{1}{2\sigma_\alpha^2} [\mathbf{h} * \alpha - \alpha_0]_i^2 \right\}, \tag{8.32}$$

Algorithm 13 Blur Kernel Estimation (Foreground Blur)

Given $\mathbf{g}$ and α_0. Crop an interior region $\mathbf{g}_I$ from the foreground of $\mathbf{g}$.
Initially set $\mathbf{f}_I = \mathbf{g}_I$. Compute $\alpha_{0S} = \text{shock filters}(\alpha_0)$.
while Not converge **do**
Compute $\mathbf{f}_{IS} = \text{shock filter}(\mathbf{f}_I)$.
Compute $\nabla^s \mathbf{f}_{IS} = \text{edge selection}(\mathbf{f}_{IS})$.
Estimate

$$\mathbf{h} = \mathcal{F}^{-1}\left\{\frac{\mu\overline{\mathcal{F}(\nabla^s \mathbf{f}_{IS})}\cdot\mathcal{F}(\nabla^s \mathbf{f}_I) + \gamma\overline{\mathcal{F}(\nabla\alpha_{0S})}\cdot\mathcal{F}(\nabla\alpha_0)}{\mu|\mathcal{F}(\nabla^s \mathbf{f}_{IS})|^2 + \gamma|\mathcal{F}(\nabla\alpha_{0S})|^2 + \lambda}\right\}.$$

Estimate

$$\mathbf{f}_I = \mathcal{F}^{-1}\left\{\frac{\overline{\mathcal{F}(\mathbf{h})}\mathcal{F}(\mathbf{g}_I) + \kappa\left[\overline{\mathcal{F}(\partial_x)}\mathcal{F}(\nabla^s_x \mathbf{g}_{IS}) + \overline{\mathcal{F}(\partial_y)}\mathcal{F}(\nabla^s_y \mathbf{g}_{IS})\right]}{|\mathcal{F}(\mathbf{h})|^2 + \kappa(|\mathcal{F}(\partial_x)|^2 + |\mathcal{F}(\partial_y)|^2)}\right\}.$$

end while
(Remark: For background blur, set $\gamma = 0$.)

where Ω_α is the set of pixels in α.

We also assume that the isotropic gradient at any pixel location is exponentially distributed. Thus,

$$P(\alpha) \propto \prod_{i\in\Omega_\alpha} \exp\left\{-\frac{1}{\tau}\sqrt{[\nabla_x\alpha]_i^2 + [\nabla_y\alpha]_i^2}\right\}, \tag{8.33}$$

where τ is a constant.

Substituting (8.32) and (8.33) into (8.31) yields the following minimization

$$\underset{\alpha}{\text{minimize}} \quad \|\alpha_I - \mathbf{h} * \alpha\|^2 + \lambda\|\alpha\|_{TV}, \tag{8.34}$$

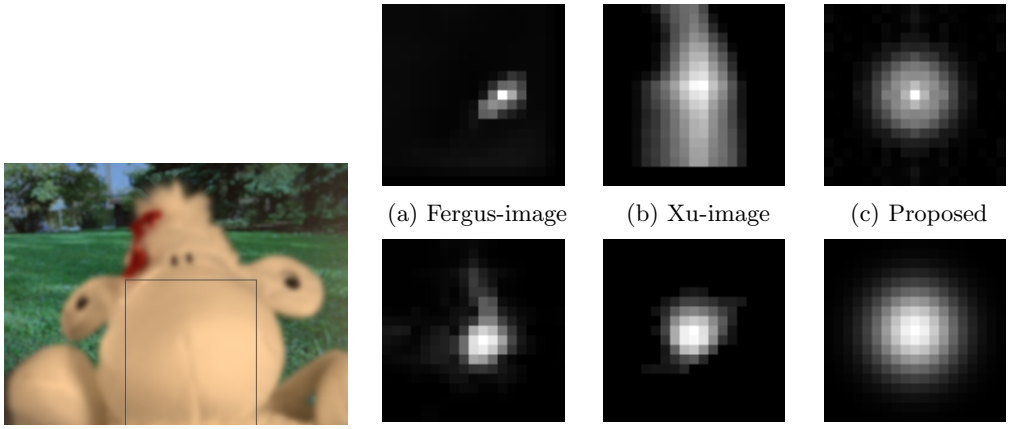
where $\|\alpha\|_{TV} = \sum_i \sqrt{[\nabla_x\alpha]_i^2 + [\nabla_y\alpha]_i^2}$ is the isotropic total variation norm, and λ is a regularization parameter derived from τ and σ_α^2. (8.34) does not have analytic solution, but can be solved efficiently using [19].

8.4.8 Overall Algorithm for Estimating h

The blur kernel estimation step is outlined in Algorithm 13. Note that Algorithm 13 is used for foreground blur cases. In case of background blur, we set $\gamma = 0$.

8.5 Results

A data set of 27 training images from http://www.alphamatting.com are downloaded for the comparisons. These images are all composed of a sharp foreground object and an out-of-focus blurred background scene. For images no.1-23, the object is placed in front of a high-definition (HD) monitor showing some background scenes, whereas for images no.24-27, the object is placed in front of real 3D scenes. Ground-truth alpha-mattes are available in this data set, but we will also test the proposed algorithm with estimated alpha-mattes later.



(a) Fergus-image (b) Xu-image (c) Proposed

Image No.7 (d) Fergus-α (e) Xu-α (g) ground-truth

Figure 8.10: Kernel estimation for Image no. 7. (a) Fergus et al. [40] on cropped region. (b) Xu and Jia [116] on cropped region. (c) Proposed method. (d) Fergus et al. [40] on alpha-matte. (e) Xu and Jia [116] on alpha-matte. (f) Ground-truth kernel.

8.5.1 Blur Kernel Estimation

First, we compare the proposed kernel estimation method with the method by Fergus *et al.* [40] and the method by Xu and Jia [116]. Note that Xu and Jia supersedes Cho and Lee [26], and Fergus *et al.* [40] is used in [31].

We synthesize two foreground blur images using a Gaussian blur kernel of size 19×19 and variance $\sigma = 3$ (See Fig. 8.10 and Fig. 8.11). Shared matting [44] is applied to the blurred images so that alpha-mattes are estimated. Interior regions were cropped manually.

In Fig. 8.10, the cropped interior region does not have strong edges. Thus, applying [40] and [116] to the interior region does not produce good estimates (Fig.

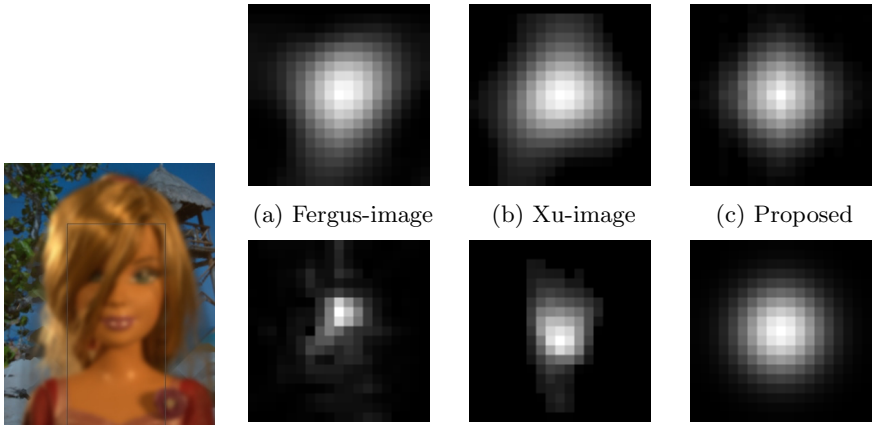


Figure 8.11: Kernel estimation for Image no. 8. (a) Fergus et al. [40] on cropped region. (b) Xu and Jia [116] on cropped region. (c) Proposed method. (d) Fergus et al. [40] on alpha-matte. (e) Xu and Jia [116] on alpha-matte. (f) Ground-truth kernel.

8.10(a)-(b)). On the other hand, when the alpha-matte is poorly estimated (Fig. 8.11), applying [40] and [116] to the alpha-matte does not give good estimates (Fig. 8.11(d)-(e)). The proposed method automatically weights the emphasis on the alpha-matte and the cropped region. Therefore, the kernel estimation result is better than the other two methods.

8.5.2 Real Background Blur

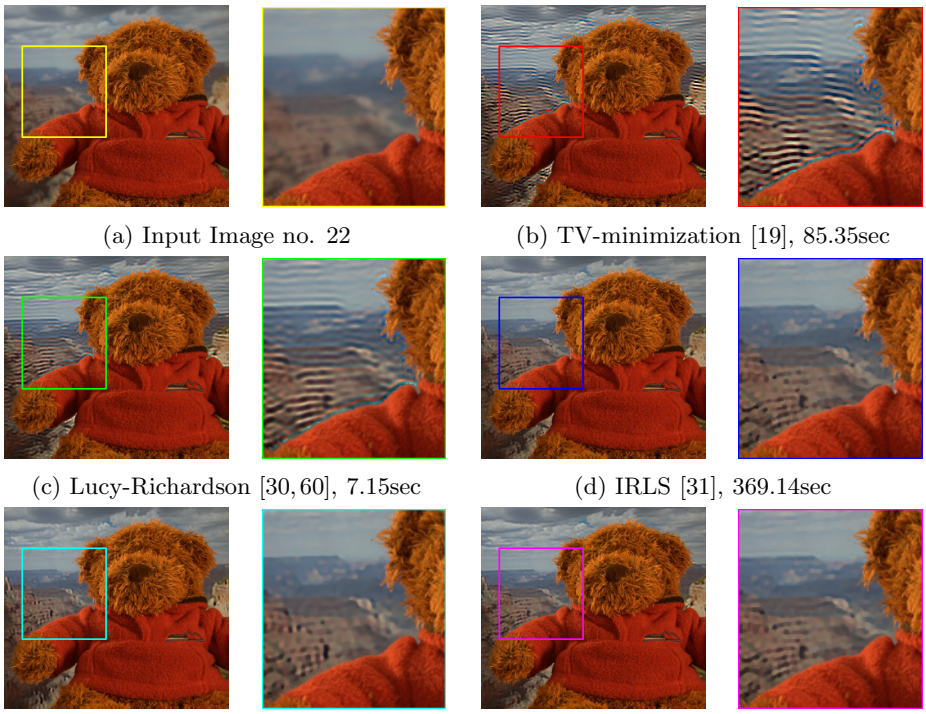
Next we compare the overall performance of the proposed method with three existing spatially variant deconvolution algorithms.

The first method to be compared is the spatially variant Lucy-Richardson (LR) algorithm used in [60] and [30]. In this method, the spatially variant blur $\mathbf{h}$ is expressed as a linear combination of invariant blurs. The deblurring step is performed via an iterative approach as

$$\mathbf{f}^{(k+1)} = \mathbf{f}^{(k)} \cdot \left[\mathbf{h}' * \left(\frac{\mathbf{g}}{\mathbf{h} * \mathbf{f}^{(k)}} \right) \right],$$

where $\mathbf{f}^{(k)}$ is the solution of the k-th iteration, $\mathbf{h}'$ is the flipped version of $\mathbf{h}$, i.e., $\mathbf{h}'(m,n) = \mathbf{h}(-m,-n)$. The multiplication "$\cdot$" and the division in the parenthesis are element-wise operations. The algorithm terminates when $\|\mathbf{f}^{(k+1)} - \mathbf{f}^{(k)}\|^2 / \|\mathbf{f}^{(k)}\|^2 \leq 10^{-3}$.

The second method is a modified version of the total variation (TV) minimization using augmented Lagrangian method [19]. We express the spatially variant operator $\mathbf{h}$



(a) Input Image no. 22 (b) TV-minimization [19], 85.35sec

(c) Lucy-Richardson [30, 60], 7.15sec (d) IRLS [31], 369.14sec

(e) Proposed (Ground-Truth α), 9.45sec (f) Proposed (Estimated α), 9.62sec

Figure 8.12: Real image background deblurring for Image No. 22. In methods shown here, TV-minimization, Lucy-Richardson, IRLS and Proposed (Ground-Truth α) use the ground-truth alpha-matte for blur kernel estimation and deblurring. However, Proposed (Estimated α) uses the shared matting method for the same tasks.

as a non circulant matrix $\mathbf{H}$. Then the $\mathbf{f}$-subproblem (Equation (14) of [19])

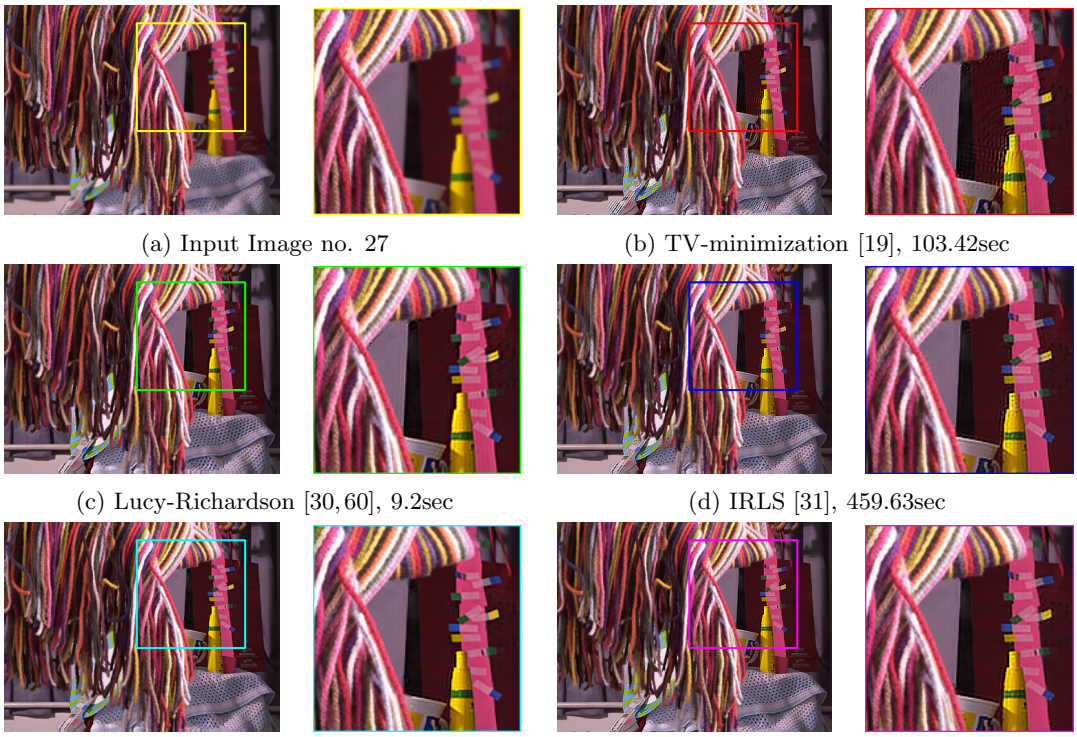
$$(\mu\mathbf{H}^T\mathbf{H} + \rho\mathbf{D}^T\mathbf{D})\mathbf{f} = \mu\mathbf{H}^T\mathbf{g} + \rho\mathbf{D}^T\mathbf{u} - \mathbf{D}\mathbf{y}$$

is solved using conjugate gradient iterations.

The third method is the one by Dai and Wu [31], which is the most relevant method to our approach. [31] solves the minimization problem

$$\underset{\mathbf{f}_F,\,\mathbf{f}_B}{\text{minimize}} \quad \|\mathbf{g} - \alpha\mathbf{f}_F - (1-\alpha)(\mathbf{h}_B * \mathbf{f}_B)\|^2 + \lambda_1\|\mathbf{f}_F\|_{TV} + \lambda_2\|\mathbf{f}_B\|_{TV}, \tag{8.35}$$

using an iterative reweighted least-squares (IRLS) method. Here, we use the standard isotropic TV norm $\|\cdot\|_{TV}$ instead of the l_p-norm ($p = 0.8$) in [31], because the goal of this chapter is not to compare different TV norms. In solving (8.35), $\mathbf{f}_F$ and $\mathbf{f}_B$ are determined simultaneously. Since the linear operators in (8.35) are not block-circulant,



(a) Input Image no. 27

(b) TV-minimization [19], 103.42sec

(c) Lucy-Richardson [30, 60], 9.2sec

(d) IRLS [31], 459.63sec

(e) Proposed (Ground-Truth α), 7.26sec

(f) Proposed (Estimated α), 7.14sec

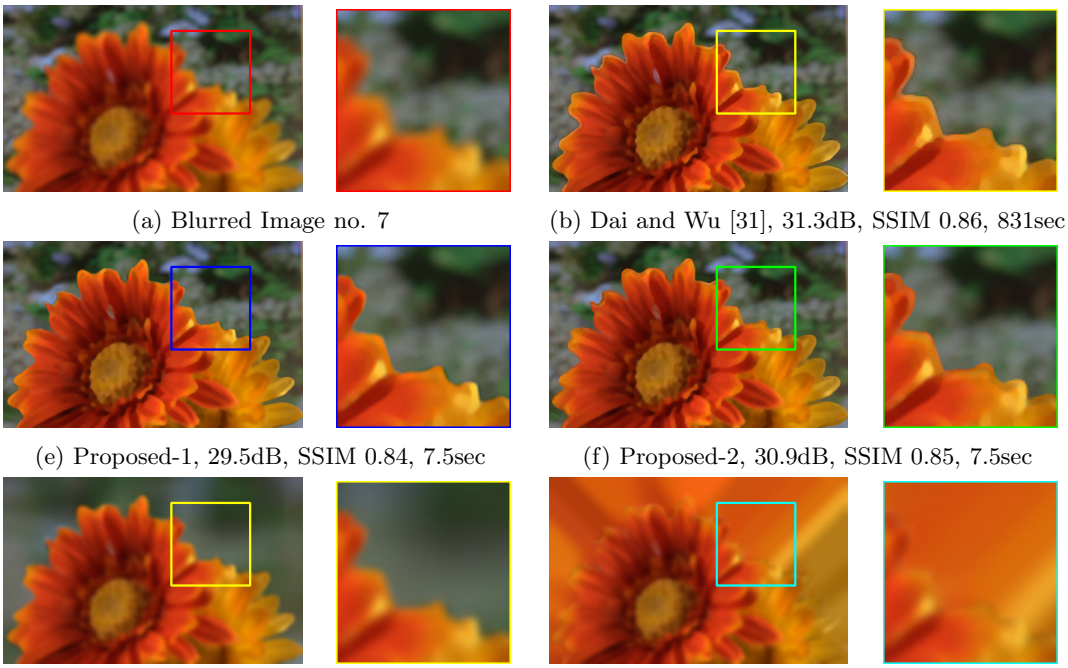
Figure 8.13: Real image background deblurring for Image No. 27. In methods shown here, TV-minimization, Lucy-Richardson, IRLS and Proposed (Ground-Truth α) use the ground-truth alpha-matte for blur kernel estimation and deblurring. However, Proposed (Estimated α) uses the shared matting method for the same tasks.

Fourier Transform cannot be used. Therefore, the speed of solving (8.35) is expected to be slow.

Fig. 8.12 and Fig. 8.13 are two of the 27 images being tested. Referring to the images, the foreground is sharp and the background is blurred. Since the blur kernels are unknown, we applied the proposed algorithm to estimate the blur kernel. The estimated blur kernel is then applied to the three existing methods listed in Fig. 8.12 and Fig. 8.13.

Two versions of the proposed method are also tested. Proposed (Ground-Truth α) uses the ground-truth alpha-matte for kernel estimation, background inpainting and deblurring, whereas Proposed (Estimated α) uses the shared matting results for kernel estimation, background inpainting and deblurring.

The run-time of these methods are recorded based on a desktop computer with Intel Quadcore Q9550 2.8GHz, 4GB DDR3, MATLAB/ Windows 7. It can be seen



(a) Blurred Image no. 7

(b) Dai and Wu [31], 31.3dB, SSIM 0.86, 831sec

(e) Proposed-1, 29.5dB, SSIM 0.84, 7.5sec

(f) Proposed-2, 30.9dB, SSIM 0.85, 7.5sec

(c) $\widetilde{\mathbf{g}}$ using Proposed-1

(d) $\widetilde{\mathbf{g}}$ using Proposed-2

Figure 8.14: Synthetic image foreground deblurring for Image No. 1. PSNR, SSIM, and Run-time can be referred to Table II.

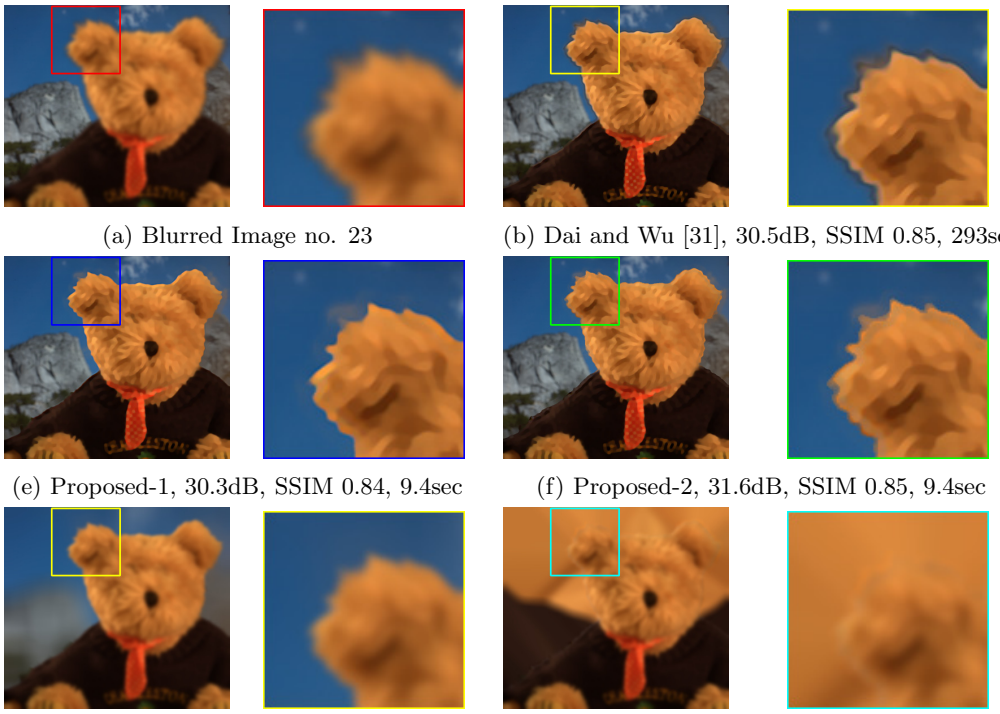
that the proposed method shows significantly faster speed than [31], and better recovery results than [30] and [19].

8.5.3 Synthetic Foreground Blur

We now show the PSNR and SSIM comparisons between existing methods. To do so, the foreground of the 27 testing images are *synthetically* blurred. The blur kernel in this experiment is a Gaussian blur kernel with size 19×19 and variance $\sigma = 3$. Since [19], [30] and [60] are evidently not able to handle the two-layer blur, comparing to [31] is sufficient.

For [31], we first apply the proposed kernel estimation algorithm to estimate the blur kernel. Once the kernel is estimated, it is fixed in the iteration of the IRLS algorithm. For fairness alpha-matting is performed using shared matting [44], same as the proposed method.

Two sets of the results are shown in Fig. 8.14 and Fig. 8.15. It can be observed that the proposed method generally produces similar image quality. However, the run



(a) Blurred Image no. 23 (b) Dai and Wu [31], 30.5dB, SSIM 0.85, 293sec

(e) Proposed-1, 30.3dB, SSIM 0.84, 9.4sec (f) Proposed-2, 31.6dB, SSIM 0.85, 9.4sec

(c) $\widetilde{\mathbf{g}}$ using Proposed-1 (d) $\widetilde{\mathbf{g}}$ using Proposed-2

Figure 8.15: Synthetic image foreground deblurring for Image No. 23. PSNR, SSIM, and Run-time can be referred to Table II.

time is significantly shorter. PSNR, SSIM and the run-times are listed in Table 8.2.

8.5.4 Real Foreground Blur

Finally, we applied the proposed algorithm to deblur real images with blurred foreground. The images were captured using a Canon ESO REBEL T2i camera. The focal length is 24mm, and ISO is 400. The images consist of a toy placed in front of posters with different background contents. The distance between the object and the background is approximately 30cm.

To recover the foreground image, we applied the shared matting algorithm to first extract the foreground object (with blurred edges). Then, the proposed blur kernel estimation method is run to determine the blur kernel, and consequently the deblurring step could be performed.

Fig. 8.16 shows one of the results. Similar to the case of synthetic blur, the proposed method is able to recover the image giving comparable quality with Dai and Wu's

Table 8.2: PSNR, SSIM and run-time comparisons among [31], Proposed method 1 and Proposed method 2 on synthetic foreground blurred images

Image No.	Size		PSNR (dB)			SSIM			Run-time (sec)		
	rows	cols	[31]	Proposed-1	Proposed-2	[31]	Proposed-1	Proposed-2	[31]	Proposed-1	Proposed-2
1	249	400	31.3	29.5	30.9	0.86	0.84	0.85	831.44	7.53	7.53
2	262	400	27.5	26.8	26.9	0.83	0.82	0.82	783.42	7.25	7.28
3	400	320	31.2	33.7	33.9	0.86	0.88	0.88	955.35	8.15	8.21
4	282	400	25.2	28.6	28.9	0.76	0.81	0.82	109.18	7.59	7.52
5	276	400	29.4	30.0	32.7	0.91	0.92	0.93	816.58	7.62	7.61
6	339	400	31.6	31.5	33.7	0.92	0.92	0.93	1230.53	9.58	9.57
7	309	400	31.9	29.3	32.6	0.92	0.90	0.92	92.34	8.54	8.55
8	400	324	26.8	28.9	29.9	0.80	0.84	0.84	217.53	8.61	8.61
9	400	322	29.9	30.7	32.1	0.83	0.86	0.86	293.05	8.75	8.76
10	286	400	31.7	31.2	32.2	0.85	0.85	0.86	805.46	7.54	7.54
11	311	400	27.4	27.3	27.5	0.76	0.75	0.76	148.23	8.24	8.25
12	264	400	30.6	31.0	32.3	0.86	0.87	0.87	102.81	6.84	6.84
13	298	400	27.4	27.9	28.5	0.74	0.76	0.76	1561.58	7.81	7.81
14	265	400	31.8	30.3	32.3	0.93	0.93	0.94	109.45	7.26	7.28
15	245	400	31.3	32.2	34.9	0.93	0.94	0.94	78.68	6.89	6.87
16	268	400	30.5	28.8	31.1	0.88	0.87	0.88	113.42	7.19	7.20
17	283	400	32.1	31.8	33.4	0.89	0.90	0.90	296.12	8.05	8.08
18	323	400	30.5	29.9	31.4	0.90	0.90	0.91	1173.86	8.42	8.42
19	290	400	27.5	27.3	27.4	0.82	0.81	0.81	831.24	7.72	7.71
20	288	400	32.3	33.1	35.8	0.90	0.90	0.92	175.64	7.81	7.80
21	332	400	29.6	29.5	30.3	0.83	0.84	0.85	2188.35	8.39	8.40
22	360	400	29.7	30.2	30.2	0.78	0.78	0.78	1308.15	9.14	9.15
23	358	400	30.5	30.3	31.6	0.85	0.84	0.85	293.24	9.41	9.41
24	272	400	28.9	28.5	29.3	0.85	0.86	0.86	85.31	7.45	7.45
25	266	400	23.8	23.2	23.3	0.83	0.82	0.82	250.61	7.16	7.18
26	302	400	21.4	21.1	21.6	0.75	0.76	0.76	2598.53	7.98	7.92
27	304	400	22.7	22.1	22.7	0.74	0.74	0.74	201.42	8.12	8.15

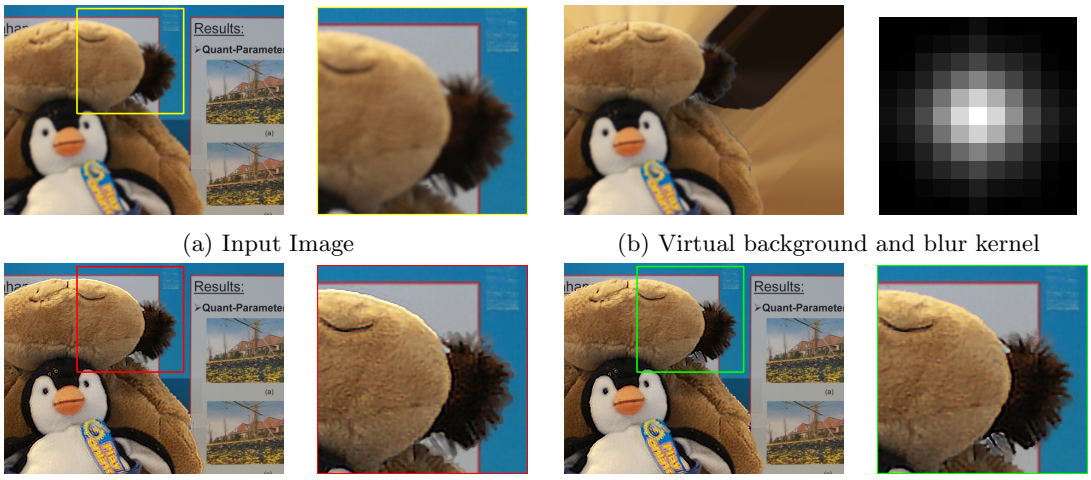
method [31]. However, the computation time of the proposed method is significantly shorter.

More results are available at http://videoprocessing.ucsd.edu/~stanleychan.

8.6 Conclusion

This chapter has two main contributions. First, we proposed a new blur kernel estimation algorithm for the two-layer out-of-focus blur problem. The new algorithm encapsulates the strength of two existing classes of methods by utilizing both the alpha-matte transient and image gradient. Experimental results showed that the new algorithm is more robust than existing blur kernel estimation methods. Second, we proposed a new method to transform the spatially variant blur problem to two spatially invariant blur problems so that fast deconvolution algorithms can be used. The new method predicts the occluded background for the background blur case, and creates virtual background for the foreground blur case. Experimental results showed that the proposed method produces better recovery results than existing methods while at a significantly faster speed.

The proposed method is limited by a number of factors: accuracy of the initial alpha-matte estimation, signal-to-noise ratio of the image, degree of blurriness of the



(a) Input Image (b) Virtual background and blur kernel

(c) Dai and Wu [31] (d) Proposed

Figure 8.16: Real image foreground deblurring (1). (a) The input image is captured using Canon ESO T2i camera. The background is about 30cm from the foreground. (b) The virtual background created by the proposed algorithm and the blur kernel estimated using the proposed algorithm. (c) Deblurring results by Dai and Wu [31]. (d) Deblurring results by the proposed method. In this image, alpha-matte is estimated using shared-matting.

image, and presence of object motion. Future research shall be focused on overcoming these issues.

8.7 Acknowledgment

This chapter, in part, is submitted and presented as

Stanley H. Chan, "Single Image Two-layered Out-of-focus Blur Removal," submitted to *IEEE Transactions on Image Processing*, Oct 2011.

Stanley H. Chan, and Truong Q. Nguyen, "Single Image Spatial Variant Out-of-focus Blur Removal," to appear in *Proceedings of IEEE International Conference on Image Processing*, Sep 2011.

Chapter 9

Conclusion and Future Work

9.1 Conclusion

This dissertation investigates solutions for the problem of image and video restoration. In particular, we develop algorithms to estimate $\mathbf{h}$ and $\mathbf{f}$ from the observation $\mathbf{g} = \mathbf{h} * \mathbf{f} + \eta$. The tools that we used is a set of numerical optimization techniques.

The dissertation begins with the discussion of how to determine $\mathbf{f}$, with the assumption that $\mathbf{h}$ is known. This problem is known as the non-blind deconvolution, which can be formulated as a least-squares minimization problem. The difficulty of deconvolution is caused by the fact that the convolution matrix associated with $\mathbf{h}$ is often rank deficient. Therefore, regularization must be added so that meaningful solutions can be determined. In Chapter 3, common regularization functions such as Tikhonov regularization, isotropic and anisotropic total variation (TV) regularization, and bilateral total variation regularization are studied. Among these regularization functions, the TV regularization gives the most promising results. However, total variation minimization is computationally intensive.

To develop a fast algorithm for solving total variation minimization problems, we studied the augmented Lagrangian method. Traditionally, the augmented Lagrangian method is developed for twice differentiable functions, which does not fit our TV problems as the TV norm is non-differentiable. However, we showed that while TV norm is non-differentiable, the augmented Lagrangian method is still valid because TV norm is convex. This observation is also verified previously in the work of operator split-

ting methods. Our proposed method transforms the unconstrained TV minimization to an equivalent constrained minimization problem. Then using the properties of the augmented Lagrangian function, we split the original problem into three relatively easy sub-problems. Each subproblem is then solved by either closed-form solution, or a system of linear equations. By iteratively searching the subproblem solutions, the solution of the original problem can be approximated. In order to improve the convergence, an automatic parameter update scheme is also proposed. Thorough comparisons show that the proposed algorithm is more superior than existing arts.

Migrating from image restoration is the problem of video restoration. In Chapter 5 we extend the idea of the augmented Lagrangian method to handle the three-dimensional space-time data. We show that the proposed three-dimensional augmented Lagrangian method not only inherits the merits of the two-dimensional method, but it also introduces additional benefits that previously cannot be found in image restoration. First, by introducing the space-time total variation norm, both spatial and temporal consistency of the video are improved. This outperforms most existing video restoration algorithms which only solve for a sequence of images independently. Second, by embedding motion blurred images into a space-time volume, we transform the spatially variant motion blur problem into an invariant blur problem. With the aid of motion estimation and compensation algorithms, motion blurred objects in a scene can be recovered.

Next, the dissertation discusses the issue of blind deconvolution, where $\mathbf{h}$ is no longer assumed known a priori. In Chapter 6 we give an overview of state-of-art blind deconvolution algorithms, with our interpretation and implementation. Our method first seeks sharp edges of the image by means of shock filter. The sharp edges are then used for estimating $\mathbf{h}$. Once an accurate $\mathbf{h}$ is found, the image restoration proposed in Chapter 4 is used to recover the image.

The assumption that the blur is spatially invariant is a restrictive one, and must be handled in practice. Among all the works the most important step is to construct a spatially variant convolution matrix. Otherwise, it is not possible to analyze the spatially variant blur. In Chapter 7 we propose a systematic algorithm in allocating non-zero entries of the convolution matrix. The convolution matrix allows one to solve spatially variant blur removal efficiently, as compared to not using the convolution matrix. The convolution matrix also allows one to analyze the spectral properties. In particular, the upper and lower bounds on the eigenvalues of the convolution matrix is estimated.

Consequently, the condition number of the convolution can be found. This, we believe, would allow us to determine the performance limit of spatially variant image restoration.

Finally, with all the techniques developed, we discuss the issue of spatially variant blind deconvolution problem caused by out-of-focus blur. The general out-of-focus blur is intractable, but simplified cases such as a two layer blur are solvable. In Chapter 8, we develop an algorithm for removing a two layer blur. The algorithm is a combination of image restoration developed in Chapter 4, the blind deconvolution algorithm discussed in Chapter 6 and alpha-matting techniques. Compared to existing methods, the proposed method is computationally inexpensive, yet produces high quality results.

9.2 Future Work

Envisioning the future research directions, we have the following suggestions.

- Operator splitting method is proved to be valid for splitting two functions only. There is no evident that splitting into three or more functions are still valid (as in the case of TV/L1), although in TV/L1 the algorithm seems performing well.

- Total variation may not be the best prior. In fact, recent researches find that the statistics of natural images are better explained by an l_p-norm with $p < 1$. Therefore, approximating l_p norm problems into a sequences of inexpensive procedures becomes a new challenge.

- Among all video restoration applications, we find that the proposed TV/L1 algorithm is a good method for disparity refinement. However, in order to perform disparity refinement in real-time, one must consider the problems of occlusion and fast motion. Additionally, angular consistency across the disparity maps should also be taken care of. Moreover, since $\mathbf{h}$ is a delta function in disparity estimation, there must be room for speed improvement.

- The blind deconvolution proposed in this dissertation has a big room for improvement. The algorithm currently fails when the blur is complicated.

- Performance limit of image restoration must be studied. Given the convolution matrix and the noise statistics, there must be a lower bound on the least-squares residue that one can achieve.

- One important finding of this dissertation is that for many spatially variant problems it is possible to find equivalent invariant problems. For example, the motion blur can be transformed to an invariant problem by embedding the image into a space-time volume using motion estimation and compensation. The out-of-focus blur problem can be transformed to an invariant problem by generating the artificial backgrounds for the foreground and background objects. We believe there are more similar situations.

Appendix A

Proofs of Chapter 3

Proposition 1. *The dual of the problem*

$$\underset{\mathbf{f}}{\textit{minimize}} \quad \frac{1}{2}\left\|\mathbf{Hf}-\mathbf{g}\right\|_2^2+\lambda\left\|\mathbf{Df}\right\|_1 \tag{A.1}$$

is given by

$$\begin{aligned} \underset{\mu,\nu}{\textit{maximize}} \quad & -\tfrac{1}{2}\|\mu\|_2^2-\mu^T\mathbf{g} \\ \textit{subject to} \quad & |\nu_i|\leq\lambda, \quad \forall i, \\ & \mathbf{H}^T\mu+\mathbf{D}^T\nu=0. \end{aligned} \tag{A.2}$$

Proof. Define $G:\mathbb{R}^{MN\times 1}\to\mathbb{R}$ and $F:\mathbb{R}^{MN\times 1}\to\mathbb{R}$ by $G(\mathbf{u}):=\frac{1}{2}\|\mathbf{u}-\mathbf{v}\|_2^2$ and $F(\mathbf{u}):=\lambda\|\mathbf{u}\|_1$. Then we have $G(\mathbf{Hf}):=\frac{1}{2}\|\mathbf{Hf}-\mathbf{g}\|_2^2$ and $F(\mathbf{Df}):=\lambda\|\mathbf{Df}\|_1$. Thus problem (A.1) can be rephrased as $\underset{\mathbf{f}}{\text{minimize}} \quad G(\mathbf{Hf})+F(\mathbf{Df})$, which is equivalent to

$$\begin{aligned} \underset{\mathbf{f}}{\text{minimize}} \quad & G(\mathbf{y})+F(\mathbf{z}) \\ \text{subject to} \quad & \mathbf{y}=\mathbf{Hf} \\ & \mathbf{z}=\mathbf{Df} \end{aligned}$$

The Lagrange function of this constrained problem is

$$L(\mathbf{f},\mathbf{y},\mathbf{z},\mu,\nu)=G(\mathbf{y})+F(\mathbf{z})+(\mathbf{y}-\mathbf{Hf})^T\mu+(\mathbf{z}-\mathbf{Df})^T\nu.$$

Therefore, the Lagrange dual function can be found as

$$
\begin{aligned}
g(\mu,\nu) &:= \inf_{\mathbf{f},\mathbf{y},\mathbf{z}} L(\mathbf{f},\mathbf{y},\mathbf{z},\mu,\nu) \\
&= \inf_{\mathbf{f},\mathbf{y},\mathbf{z}} \left\{ G(\mathbf{y}) + F(\mathbf{z}) + (\mathbf{y}-\mathbf{H}\mathbf{f})^T\mu + (\mathbf{z}-\mathbf{D}\mathbf{f})^T\nu \right\} \\
&= \inf_{\mathbf{f},\mathbf{y},\mathbf{z}} \left\{ G(\mathbf{y}) + \mathbf{y}^T\mu + F(\mathbf{z}) + \mathbf{z}^T\nu - \mathbf{f}^T(\mathbf{H}^T\mu + \mathbf{D}^T\nu) \right\} \\
&= \inf_{\mathbf{y}} \left\{ G(\mathbf{y}) + \mathbf{y}^T\mu \right\} + \inf_{\mathbf{z}} \left\{ F(\mathbf{z}) + \mathbf{z}^T\nu \right\} - \inf_{\mathbf{f}} \left\{ \mathbf{f}^T(\mathbf{H}^T\mu + \mathbf{D}^T\nu) \right\} \\
&= -\sup_{\mathbf{y}} \left\{ -G(\mathbf{y}) - \mathbf{y}^T\mu \right\} - \sup_{\mathbf{z}} \left\{ -F(\mathbf{z}) - \mathbf{z}^T\nu \right\} - \inf_{\mathbf{f}} \left\{ \mathbf{f}^T(\mathbf{H}^T\mu + \mathbf{D}^T\nu) \right\} \\
&= -G^*(-\mu) - F^*(-\nu) - \inf_{\mathbf{f}} \left\{ \mathbf{f}^T(\mathbf{H}^T\mu + \mathbf{D}^T\nu) \right\} \\
&= \begin{cases} -G^*(-\mu) - F^*(-\nu) & \text{if} \quad \mathbf{H}^T\mu + \mathbf{D}^T\nu = 0, \\ \infty & \text{otherwise.} \end{cases}
\end{aligned}
$$

Therefore, the dual is given by

$$
\begin{aligned}
\underset{\mu,\nu}{\text{maximize}} \quad & -G^*(-\mu) - F^*(-\nu) \\
\text{subject to} \quad & \mathbf{H}^T\mu + \mathbf{D}^T\nu = 0.
\end{aligned}
$$

Finally, using the following observations from [52],

$$
\begin{aligned}
F^*(\nu) &= \begin{cases} 0 & \text{if} \quad |\nu_i| \le \lambda, \quad \forall i, \\ \infty & \text{otherwise,} \end{cases} \\
G^*(\mu) &= \frac{1}{2}\|\mu\|^2 + \mu^T\mathbf{g},
\end{aligned}
$$

the dual of (A.1) is

$$
\begin{aligned}
\underset{\mu,\nu}{\text{maximize}} \quad & -\tfrac{1}{2}\|\mu\|_2^2 - \mu^T\mathbf{g} \\
\text{subject to} \quad & |\nu_i| \le \lambda, \quad \forall i, \\
& \mathbf{H}^T\mu + \mathbf{D}^T\nu = 0,
\end{aligned}
$$

which completes the proof.

□

Appendix B

Proofs of Chapter 4

B.1 Case 1: $\rho_k \to \infty$

Exact Method (Algorithm 1)

Lemma 1. *The subdifferential of the function $f(\mathbf{x}) = \|\mathbf{x}\|_1$ is a bounded set.*

Proof. A vector $\mathbf{d} \in \mathbb{R}^n$ is a subgradient of f at a point $\mathbf{x} \in \mathbb{R}^n$ if

$$f(\mathbf{z}) \geq f(\mathbf{x}) + (\mathbf{z} - \mathbf{x})^T \mathbf{d}.$$

Let x be a component of $\mathbf{x}$. Since $|z| \geq z$ for any $z \in \mathbb{R}$, so if $x > 0$, then we have $|z| \geq x + (z - x) \cdot 1$ and if $x < 0$, then $|z| \geq |x| + (z - x) \cdot (-1)$. If $x = 0$, then for any $d \in [-1,\ 1]$, we have $|z| \geq |x| + (z - x)d$. Therefore, if $\mathbf{d}$ is a subgradient of $f(\mathbf{x}) = \|\mathbf{x}\|_1$, then any component of $\mathbf{d}$ is given by $d = +1$ if $x > 0$, $d = -1$ if $x < 0$, and $d \in [-1, +1]$ if $x = 0$. So, any subgradient $\mathbf{d}$ of $\partial \|\mathbf{x}\|_1$ is bounded by $\|\mathbf{d}\|_\infty \leq 1$ and so the subdifferential $\partial \|\mathbf{x}\|_1$ is a bounded set. □

Lemma 2. *If $\mathbf{y}_k$ is generated by Algorithm 1, then the sequence $\{\mathbf{y}_k\}$ is bounded, and $\|\mathbf{y}_{k+1}\|_\infty \leq 1$.*

Proof. By optimality of $\mathbf{u}_{k+1}$, we have that $0 \in \partial_{\mathbf{u}} L_{\rho_k}(\mathbf{f}_{k+1}, \mathbf{u}_{k+1}, \mathbf{y}_k)$. So $0 \in \partial \|\mathbf{u}_{k+1}\|_1 - \mathbf{y}_k + \rho_k(\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1})$. Since $\mathbf{y}_{k+1} = \mathbf{y}_k - \rho_k(\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1})$, we have $\mathbf{y}_{k+1} \in \partial \|\mathbf{u}_{k+1}\|_1$. So $\|\mathbf{y}_{k+1}\|_\infty \leq 1$, by Lemma 1. □

Lemma 3. *If* $\mathbf{y}_{k+1} = \mathbf{y}_k - \rho_k(\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1})$, *then*

$$-\mathbf{y}_k^T(\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1}) + \frac{\rho_k}{2}\|\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1}\|^2 = \frac{1}{2\rho_k}\left(\|\mathbf{y}_{k+1}\|^2 - \|\mathbf{y}_k\|^2\right).$$

Proof. By substitution. □

Theorem 4. *Suppose that* $\rho_k \to \infty$ *as* $k \to \infty$. *The sequence* $(\mathbf{f}_k, \mathbf{u}_k)$ *generated by Algorithms 1 converges to* $(\mathbf{f}^*, \mathbf{u}^*)$, *where* $(\mathbf{f}^*, \mathbf{u}^*)$ *is the optimal solution to the problem*

$$\begin{array}{ll} \underset{\mathbf{f},\mathbf{u}}{\text{minimize}} & \frac{\mu}{2}\|\mathbf{H}\mathbf{f} - \mathbf{g}\|^2 + \|\mathbf{u}\|_1 \\ \text{subject to} & \mathbf{D}\mathbf{f} = \mathbf{u}, \end{array}$$

Proof. First of all, note that

$$\begin{aligned} L(\mathbf{f}_{k+1}, \mathbf{u}_{k+1}, \mathbf{y}_k, \rho_k) &= \min_{\mathbf{f},\mathbf{u}} L(\mathbf{f}, \mathbf{u}, \mathbf{y}_k, \rho_k) \le \min_{\mathbf{D}\mathbf{f}=\mathbf{u}} L(\mathbf{f}, \mathbf{u}, \mathbf{y}_k, \rho_k) \\ &= \min_{\mathbf{D}\mathbf{f}=\mathbf{u}} \left(\frac{\mu}{2}\|\mathbf{H}\mathbf{f} - \mathbf{g}\|^2 + \|\mathbf{u}\|_1\right) \triangleq \phi^*. \end{aligned}$$

Therefore, using Lemma 3 we have

$$\begin{aligned} \frac{\mu}{2}\|\mathbf{H}\mathbf{f}_{k+1} - \mathbf{g}\|^2 + \|\mathbf{u}_{k+1}\|_1 &= L(\mathbf{f}_{k+1}, \mathbf{u}_{k+1}, \mathbf{y}_k, \rho_k) - \frac{1}{2\rho_k}\left(\|\mathbf{y}_{k+1}\|^2 - \|\mathbf{y}_k\|^2\right) \\ &\le \phi^* - \frac{1}{2\rho_k}\left(\|\mathbf{y}_{k+1}\|^2 - \|\mathbf{y}_k\|^2\right) \\ &\le \phi^* + o(\rho_k^{-1}), \end{aligned} \tag{B.1}$$

as $\{\mathbf{y}_k\}$ is bounded.

To prove the other direction, using triangle inequality we have

$$\begin{aligned} \frac{\mu}{2}\|\mathbf{H}\mathbf{f}_{k+1} - \mathbf{g}\|^2 + \|\mathbf{u}_{k+1}\|_1 &= \frac{\mu}{2}\|\mathbf{H}\mathbf{f}_{k+1} - \mathbf{g}\|^2 + \|\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1} + \mathbf{D}\mathbf{f}_{k+1}\|_1 \\ &\ge \frac{\mu}{2}\|\mathbf{H}\mathbf{f}_{k+1} - \mathbf{g}\|^2 + \left\|\mathbf{D}\mathbf{f}_{k+1}^*\right\|_1 - \|\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1}\|_1 \\ &\ge \phi^* - \|\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1}\|_1 \\ &= \phi^* - \rho_k^{-1}\|\mathbf{y}_{k+1} - \mathbf{y}_k\|_1 \\ &= \phi^* - o(\rho_k^{-1}). \end{aligned} \tag{B.2}$$

Therefore, as $k \to \infty$, $\rho_k \to \infty$ and hence $\frac{\mu}{2}\|\mathbf{H}\mathbf{f}_{k+1} - \mathbf{g}\|^2 + \|\mathbf{u}_{k+1}\|_1 \to \phi^*$. This

shows that the limit $\mathbf{f}^*$ of $\{\mathbf{f}_k\}$ and the limit $\mathbf{u}^*$ of $\{\mathbf{u}_k\}$ attains the minimum value for (4.12).

It remains to check the feasibility of $(\mathbf{f}^*, \mathbf{u}^*)$. As $\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1} = \frac{1}{\rho_k}(\mathbf{y}_{k+1} - \mathbf{y}_k)$ and $\{\mathbf{y}_k\}$ is bounded, by letting $k \to \infty$ we have $\mathbf{D}\mathbf{f}^* = \mathbf{u}^*$. So, $(\mathbf{f}^*, \mathbf{u}^*)$ is feasible. □

Inexact Method (Algorithm 2)

Lemma 4. *If $\mathbf{y}_k$ is generated by Algorithm 2, then $\{\mathbf{y}_k\}$ is bounded.*

Proof. The proof is similar to that of Lemma 2 □

Lemma 5. *Suppose that $\sum_{k=1}^{\infty} \frac{\rho_{k+1}}{\rho_k^2} < \infty$. Let $(\mathbf{f}_k^*, \mathbf{u}_k^*, \mathbf{y}_k^*)$ be the iterates generated by Algorithm 1, and let $(\mathbf{f}_k, \mathbf{u}_k, \mathbf{y}_k)$ be the iterates generated by Algorithm 2. The sequences $\{\mathbf{f}_k^*\}$, $\{\mathbf{u}_k^*\}$, $\{\mathbf{f}_k\}$ and $\{\mathbf{u}_k\}$ are bounded.*

Proof. First, since $(\mathbf{f}_{k+1}^*, \mathbf{u}_{k+1}^*)$ is the minimizer of $L(\mathbf{f}_{k+1}^*, \mathbf{u}_{k+1}^*, \mathbf{y}_k^*, \rho_k)$, we have

$$\begin{aligned} L(\mathbf{f}_{k+1}^*, \mathbf{u}_{k+1}^*, \mathbf{y}_k^*, \rho_k) &\le L(\mathbf{f}_k^*, \mathbf{u}_k^*, \mathbf{y}_k^*, \rho_k) \\ &= \frac{\mu}{2}\left\|\mathbf{H}\mathbf{f}_k^* - \mathbf{g}\right\|^2 + \left\|\mathbf{u}_k^*\right\|_1 - \mathbf{y}_k^{*T}(\mathbf{u}_k^* - \mathbf{D}\mathbf{f}_k^*) + \frac{\rho_k}{2}\left\|\mathbf{u}_k^* - \mathbf{D}\mathbf{f}_k^*\right\|^2. \end{aligned}$$

Substitute $\mathbf{y}_k^* = \mathbf{y}_{k-1}^* - \rho_{k-1}(\mathbf{u}_k^* - \mathbf{D}\mathbf{f}_k^*)$ yields

$$\begin{aligned} L(\mathbf{f}_{k+1}^*, \mathbf{u}_{k+1}^*, \mathbf{y}_k^*, \rho_k) &\le \frac{\mu}{2}\left\|\mathbf{H}\mathbf{f}_k^* - \mathbf{g}\right\|^2 + \left\|\mathbf{u}_k^*\right\|_1 - \mathbf{y}_k^{*T}(\mathbf{u}_k^* - \mathbf{D}\mathbf{f}_k^*) + \frac{\rho_k}{2}\left\|\mathbf{u}_k^* - \mathbf{D}\mathbf{f}_k^*\right\|^2 \\ &= \frac{\mu}{2}\left\|\mathbf{H}\mathbf{f}_k^* - \mathbf{g}\right\|^2 + \left\|\mathbf{u}_k^*\right\|_1 - \mathbf{y}_{k-1}^{*T}(\mathbf{u}_k^* - \mathbf{D}\mathbf{f}_k^*) \\ &\quad + \frac{\rho_{k-1}}{2}\left\|\mathbf{u}_k^* - \mathbf{D}\mathbf{f}_k^*\right\|^2 + \frac{\rho_k + \rho_{k-1}}{2\rho_{k-1}^2}\left\|\mathbf{y}_k^* - \mathbf{y}_{k-1}^*\right\|^2 \\ &= L(\mathbf{f}_k^*, \mathbf{u}_k^*, \mathbf{y}_{k-1}^*, \rho_{k-1}) + \frac{\rho_k + \rho_{k-1}}{2\rho_{k-1}^2}\left\|\mathbf{y}_k^* - \mathbf{y}_{k-1}^*\right\|^2. \end{aligned}$$

Since $\{\mathbf{y}_k^*\}$ is bounded and $\sum_{k=1}^{\infty} \frac{\rho_k + \rho_{k-1}}{\rho_{k-1}^2} \le 2\sum_{k=1}^{\infty} \frac{\rho_k^2}{\rho_{k-1}}$ as ρ_k is increasing, so the hypothesis implies that the sequence $L(\mathbf{f}_{k+1}^*, \mathbf{u}_{k+1}^*, \mathbf{y}_k^*, \rho_k)$ is bounded.

Now, note also that

$$\frac{\mu}{2}\left\|\mathbf{H}\mathbf{f}_k^* - \mathbf{g}\right\|^2 + \left\|\mathbf{u}_k^*\right\|_1 = L(\mathbf{f}_k^*, \mathbf{u}_k^*, \mathbf{y}_{k-1}^*, \rho_{k-1}) - \frac{\rho_k + \rho_{k-1}}{2\rho_{k-1}^2}\left\|\mathbf{y}_k^* - \mathbf{y}_{k-1}^*\right\|^2.$$

Since the terms on the right hand side are bounded, $\frac{\mu}{2}\|\mathbf{H}\mathbf{f}_k^* - \mathbf{g}\|^2 + \|\mathbf{u}_k^*\|_1$ is also bounded. Thus, $\{\mathbf{u}_k^*\}$ and $\{\mathbf{f}_k^*\}$ must be bounded.

The proof of $\{\mathbf{f}_k\}$ and $\{\mathbf{u}_k\}$ being bounded is similar: replace $\mathbf{f}_k^*$ by $\mathbf{f}_k$ and $\mathbf{u}_k^*$ by $\mathbf{u}_k$, we arrive the same conclusion. □

Lemma 6. *Let* $\hat{\mathbf{y}}_k = \mathbf{y}_{k-1} - \rho_{k-1}(\mathbf{u}_{k-1} - \mathbf{D}\mathbf{f}_k)$*, then the sequence* $\{\hat{\mathbf{y}}_k\}$ *is bounded.*

Proof. Let $\hat{\mathbf{y}}_{k+1} = \mathbf{y}_k - \rho_k(\mathbf{u}_k - \mathbf{D}\mathbf{f}_{k+1})$. Consider the optimality of $\mathbf{f}_{k+1}$, we have that $0 \in \partial_{\mathbf{f}} L(\mathbf{f}_{k+1}, \mathbf{u}_k, \mathbf{y}_k, \rho_k)$. So $0 \in \mu\mathbf{H}^T(\mathbf{H}\mathbf{f}_{k+1} - \mathbf{g}) + \mathbf{D}^T\mathbf{y}_k - \rho_k\mathbf{D}^T(\mathbf{u}_k - \mathbf{D}\mathbf{f}_{k+1})$, and hence $\mathbf{D}^T\hat{\mathbf{y}}_k = -\mu\mathbf{H}^T(\mathbf{H}\mathbf{f}_{k+1} - \mathbf{g})$. Since $\{\mathbf{f}_k\}$ is bounded, the right hand side is also bounded. Therefore, $\mathbf{D}^T\hat{\mathbf{y}}_k$ has to be bounded. As $\mathbf{D}$ is a finite difference operator, $\hat{\mathbf{y}}_k$ is bounded too. □

Theorem 5. *If* $\sum_{k=1}^{\infty}\frac{1}{\rho_k} < \infty$*,* $\sum_{k=1}^{\infty}\frac{\rho_{k+1}}{\rho_k^2} < \infty$ *and* $\lim_{k\to\infty}\rho_k(\mathbf{u}_{k+1} - \mathbf{u}_k) = 0$*, then* $(\mathbf{f}_k, \mathbf{u}_k)$ *converges to* $(\mathbf{f}^*, \mathbf{u}^*)$.

Proof. Since $\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1} = \rho_k^{-1}(\mathbf{y}_{k+1} - \mathbf{y}_k)$, and $\{\mathbf{y}_k\}$ is bounded, we have

$$\lim_{k\to\infty} \mathbf{u}_k - \mathbf{D}\mathbf{f}_k = 0,$$

and so $(\mathbf{f}_k, \mathbf{u}_k)$ converges to a feasible solution.

Next, observe that $\hat{\mathbf{y}}_{k+1} = \mathbf{y}_k - \rho_k(\mathbf{u}_k - \mathbf{D}\mathbf{f}_{k+1})$ and $\mathbf{y}_{k+1} = \mathbf{y}_k - \rho_k(\mathbf{u}_{k+1} - \mathbf{D}\mathbf{f}_{k+1})$. So $\hat{\mathbf{y}}_{k+1} - \mathbf{y}_{k+1} = \rho_k(\mathbf{u}_{k+1} - \mathbf{u}_k)$, and hence $\|\mathbf{u}_{k+1} - \mathbf{u}_k\| = \rho_k^{-1}\|\hat{\mathbf{y}}_{k+1} - \mathbf{y}_{k+1}\|$. By boundedness of $\hat{\mathbf{y}}_{k+1}$ and $\mathbf{y}_{k+1}$, we have

$$\sum_{k=1}^{\infty}\|\mathbf{u}_{k+1} - \mathbf{u}_k\| \le M\left(\sum_{k=1}^{\infty}\rho_k^{-1}\right) < \infty,$$

for some constant M. By the hypothesis that $\sum_{k=1}^{\infty}\frac{1}{\rho_k} < \infty$, so $\|\mathbf{u}_{k+1} - \mathbf{u}_k\| \to 0$ as $k \to \infty$. So $\{\mathbf{u}_k\}$ is Cauchy, and has a limit $\mathbf{u}^*$. Then by $\lim_{k\to\infty}\mathbf{u}_k - \mathbf{D}\mathbf{f}_k = 0$, we have $\{\mathbf{f}_k\}$ is also Cauchy and has a limit $\mathbf{f}^*$. In addition, since $\mathbf{u}^* = \mathbf{D}\mathbf{f}^*$, $(\mathbf{f}^*, \mathbf{u}^*)$ is feasible.

Last, we need to check if $(\mathbf{f}^*, \mathbf{u}^*)$ attains a minimum for (4.12). As the function $f(\mathbf{x}) = \|\mathbf{x}\|$ is convex, and for any convex function we have $f(\mathbf{z}) \ge f(\mathbf{x}) + (\mathbf{z} - \mathbf{x})^T\mathbf{d}$ for any $\mathbf{d} \in \partial f(x)$, so

$$\left\|\mathbf{u}_{k+1}^*\right\|_1 \ge \|\mathbf{u}_{k+1}\|_1 + (\mathbf{u}_{k+1}^* - \mathbf{u}_{k+1})^T\mathbf{y}_{k+1},$$

because $\mathbf{y}_{k+1} \in \partial \|\mathbf{u}_{k+1}\|_1$. Also,

$$\begin{aligned}\frac{\mu}{2}\left\|\mathbf{H}\mathbf{f}^*_{k+1}-\mathbf{g}\right\|^2 &\geq \frac{\mu}{2}\|\mathbf{H}\mathbf{f}_{k+1}-\mathbf{g}\|^2+(\mathbf{f}^*_{k+1}-\mathbf{f}_{k+1})^T\left(\mu\mathbf{H}^T(\mathbf{H}\mathbf{f}_{k+1}-\mathbf{g})\right)\\ &= \frac{\mu}{2}\|\mathbf{H}\mathbf{f}_{k+1}-\mathbf{g}\|^2+(\mathbf{f}^*_{k+1}-\mathbf{f}_{k+1})^T(-\mathbf{D}^T\hat{\mathbf{y}}_{k+1})\\ &= \frac{\mu}{2}\|\mathbf{H}\mathbf{f}_{k+1}-\mathbf{g}\|^2-[\mathbf{D}(\mathbf{f}^*_{k+1}-\mathbf{f}_{k+1})]^T\hat{\mathbf{y}}_{k+1}.\end{aligned}$$

Therefore, by substituting $\hat{\mathbf{y}}^*_{k+1}=\mathbf{y}_k-\rho_k(\mathbf{u}_k-\mathbf{D}\mathbf{f}_{k+1})$ and $\mathbf{y}^*_{k+1}=\mathbf{y}^*_k-\rho_k(\mathbf{u}^*_{k+1}-\mathbf{D}\mathbf{f}^*_{k+1})$, we have

$$\begin{aligned}&\frac{\mu}{2}\|\mathbf{H}\mathbf{f}_{k+1}-\mathbf{g}\|^2+\|\mathbf{u}_{k+1}\|_1\\ &\leq \frac{\mu}{2}\left\|\mathbf{H}\mathbf{f}^*_{k+1}-\mathbf{g}\right\|^2+\left\|\mathbf{u}^*_{k+1}\right\|_1+[\mathbf{D}(\mathbf{f}^*_{k+1}-\mathbf{f}_{k+1})]^T\hat{\mathbf{y}}_{k+1}-(\mathbf{u}^*_{k+1}-\mathbf{u}_{k+1})^T\mathbf{y}_{k+1}\\ &= \frac{\mu}{2}\left\|\mathbf{H}\mathbf{f}^*_{k+1}-\mathbf{g}\right\|^2+\left\|\mathbf{u}^*_{k+1}\right\|_1+[\mathbf{D}(\mathbf{f}^*_{k+1}-\mathbf{f}_{k+1})]^T[\mathbf{y}_k-\rho_k(\mathbf{u}_k-\mathbf{D}\mathbf{f}_{k+1})]\\ &\quad-\left(\frac{\mathbf{y}^*_k-\mathbf{y}^*_{k+1}}{\rho_k}+\mathbf{D}\mathbf{f}^*_{k+1}-\mathbf{u}_{k+1}\right)^T\mathbf{y}_{k+1}\\ &= \frac{\mu}{2}\left\|\mathbf{H}\mathbf{f}^*_{k+1}-\mathbf{g}\right\|^2+\left\|\mathbf{u}^*_{k+1}\right\|_1+\rho_k[\mathbf{D}(\mathbf{f}^*_{k+1}-\mathbf{f}_{k+1})]^T(\mathbf{u}_{k+1}-\mathbf{u}_k)\\ &\quad+\rho_k^{-1}(\mathbf{y}^*_{k+1}-\mathbf{y}^*_k)^T\mathbf{y}_{k+1}-\rho_k^{-1}(\mathbf{y}_{k+1}-\mathbf{y}_k)^T\mathbf{y}_{k+1}.\end{aligned}$$

By Theorem 4, $\frac{\mu}{2}\left\|\mathbf{H}\mathbf{f}^*_{k+1}-\mathbf{g}\right\|^2+\left\|\mathbf{u}^*_{k+1}\right\|_1\to\phi^*$. The next term approaches zero because $\{\mathbf{f}^*_{k+1}\}$ and $\{\mathbf{f}_{k+1}\}$ are bounded, and by assumption that $\rho_k(\mathbf{u}_{k+1}-\mathbf{u}_k)\to 0$. The next two terms also vanish because of the boundedness of $\{\mathbf{y}_k\}$ and $\{\mathbf{y}^*_k\}$. Therefore, we have

$$\frac{\mu}{2}\|\mathbf{H}\mathbf{f}^*-\mathbf{g}\|^2+\|\mathbf{u}^*\|_1\leq\phi^*,$$

and so the result is proven. □

B.2 Case 2: $\rho_k \to \rho^*$, $\rho^* < \infty$

Theorem 6. *Suppose that $\rho_k \to \rho^*$ as $k \to \infty$, where $\rho^* < \infty$. The sequence $(\mathbf{f}_k, \mathbf{u}_k)$ generated by Algorithms 1 and Algorithm 2 converges to $(\mathbf{f}^*, \mathbf{u}^*)$, where $(\mathbf{f}^*, \mathbf{u}^*)$ is the*

optimal solution to the problem

$$\begin{aligned} \underset{\mathbf{f},\mathbf{u}}{\text{minimize}} \quad & \tfrac{\mu}{2}\left\|\mathbf{H}\mathbf{f}-\mathbf{g}\right\|^2+\left\|\mathbf{u}\right\|_1 \\ \textit{subject to} \quad & \mathbf{D}\mathbf{f}=\mathbf{u}, \end{aligned}$$

Proof. The update scheme states that: if $\|\mathbf{u}_{k+1}-\mathbf{D}\mathbf{f}_{k+1}\| \geq \alpha\|\mathbf{u}_k-\mathbf{D}\mathbf{f}_k\|$, where $0<\alpha<1$, then $\rho_{k+1}=\rho_k\gamma$ for some $\gamma>1$. So $\{\rho_k\}$ is an increasing sequence. By hypothesis, $\rho_k\to\rho^*<\infty$, so $\{\rho_k\}$ must be bounded. Therefore, there exists K such that $\rho_k=\rho^*$ whenever $k>K$.

Now, for any $k>K$, it remains to show that $(\mathbf{f}_k,\mathbf{u}_k)$ generated by Algorithm 1 and Algorithm 2 converges to $(\mathbf{f}^*,\mathbf{u}^*)$, for fixed ρ^*. In this case, since ρ^* is fixed, the proof follows from [33]. □

Appendix C

Proofs of Chapter 7

C.1 Proof of Theorem 2

Proof. Let $\mathbf{E}_k = \text{diag}\{0, \ldots, 1, \ldots, 0\}$ be a diagonal matrix with the (k,k)-th entry being 1. Using $\mathbf{E}_k$, we can express $\mathbf{H}$ as a sum of its circulant components as $\mathbf{H} = \mathbf{H}_1\mathbf{E}_1 + \ldots + \mathbf{H}_n\mathbf{E}_n$. Taking the conjugate transpose and multiplying with $\mathbf{H}$ yields

$$\mathbf{H}^H\mathbf{H} = \sum_{i,j} \mathbf{E}_i^H\mathbf{H}_i^H\mathbf{H}_j\mathbf{E}_j. \tag{C.1}$$

Let $\mathbf{u}$ be the eigenvector associated with $\lambda_{\min}(\mathbf{H}^H\mathbf{H})$. By multiplying $\mathbf{u}^H$ and $\mathbf{u}$ on both sides of (C.1), the (i,j)-th term in the sum is

$$\begin{aligned}
&\mathbf{u}^H(\mathbf{E}_i^H\mathbf{H}_i^H\mathbf{H}_j\mathbf{E}_j)\mathbf{u} = \mathbf{u}^H\mathbf{E}_i^H\mathbf{F}^H\mathbf{\Lambda}_i^*\mathbf{F}\mathbf{F}^H\mathbf{\Lambda}_j\mathbf{F}\mathbf{E}_j\mathbf{u} \\
&\geq \mathbf{u}^H\mathbf{E}_i^H\mathbf{F}^H\left(|\lambda_{\min}(\mathbf{\Lambda}_i^*)||\lambda_{\min}(\mathbf{\Lambda}_j)|\mathbf{I}\right)\mathbf{F}\mathbf{E}_j\mathbf{u} \\
&= \mathbf{u}^H\mathbf{E}_i^H\mathbf{F}^H\left(|\lambda_{\min}(\mathbf{H}_i)||\lambda_{\min}(\mathbf{H}_j)|\mathbf{I}\right)\mathbf{F}\mathbf{E}_j\mathbf{u} \\
&= |\lambda_{\min}(\mathbf{H}_i)||\lambda_{\min}(\mathbf{H}_j)|\mathbf{u}^H\mathbf{E}_i^H\mathbf{F}^H\mathbf{F}\mathbf{E}_j\mathbf{u} \\
&= \begin{cases} |\lambda_{\min}(\mathbf{H}_i)|^2|u_i|^2, & \text{if } i = j, \\ 0, & \text{if } i \neq j, \end{cases}
\end{aligned}$$

because $\mathbf{F}^H\mathbf{F} = \mathbf{F}\mathbf{F}^H = \mathbf{I}$, and $\mathbf{E}_i^H\mathbf{E}_j = \mathbf{0}$ if $i \neq j$. Here, u_i is the i-th element of $\mathbf{u}$. Note that the first inequality holds because the eigenvalues of a Gaussian point spread function are real and nonnegative.

Therefore, the smallest eigenvalue of $\mathbf{H}^H\mathbf{H}$ is

$$\begin{aligned}|\lambda_{\min}(\mathbf{H}^H\mathbf{H})| &\geq \sum_{i=1}^{n} |\lambda_{\min}(\mathbf{H}_i)|^2 |u_i|^2 \\ &\geq \left(\min_i \{|\lambda_{\min}(\mathbf{H}_i)|^2\}\right) \sum_{i=1}^{n} |u_i|^2 \\ &= \min_i \{|\lambda_{\min}(\mathbf{H}_i)|^2\},\end{aligned}$$

because the eigenvector $\mathbf{u}$ has unit norm so that $\sum_{i=1}^{n} |u_i|^2 = 1$. Thus, we have $|\lambda_{\min}(\mathbf{H}^H\mathbf{H})| \geq \min_i \left\{|\lambda_{\min}(\mathbf{H}_i)|^2\right\}$. □

C.2 Proof of Corollary 2

Proof. Let $\mathbf{u}$ be the eigenvector associated with the minimum eigenvalue of $\mathbf{H}^H\mathbf{H} + \alpha\mathbf{D}^H\mathbf{D}$. It can be shown that

$$\begin{aligned}\left|\lambda_{\min}\left(\mathbf{H}^H\mathbf{H} + \alpha\mathbf{D}^H\mathbf{D}\right)\right| &= \left|\mathbf{u}^H \left(\sum_{i,j=1}^{n} \mathbf{E}_i^H \mathbf{H}_i^H \mathbf{H}_j \mathbf{E}_j + \alpha\mathbf{D}^H\mathbf{D}\right) \mathbf{u}\right| \\ &\geq \sum_{k=1}^{n} \lambda_{\min}\left(|\mathbf{\Lambda}^{\mathbf{H}_k}|^2 + \alpha|\mathbf{\Lambda}^{\mathbf{D}}|^2\right) \mathbf{u}^H \mathbf{E}_i^H \mathbf{F}^H \mathbf{F} \mathbf{E}_j \mathbf{u} \\ &\geq \min_k \left\{\min_j \left\{|\lambda_j^{\mathbf{H}_k}|^2 + \alpha|\lambda_j^{\mathbf{D}}|^2\right\}\right\}.\end{aligned}$$

□

Appendix D

Proofs of Chapter 8

Proposition 2. *Suppose that $g[-1]$ and $g[-2]$ are bounded, and hence $g[0] = 2g[-1] - g[-2]$ is also bounded. $g[n]$ satisfying the condition $g'[n] = \frac{1}{n}g'[n-1]$ has the recursion*

$$g[n] = \left(1 + \frac{1}{n}\right) g[n-1] - \frac{1}{n}g[n-2], \quad \textit{for } n > 0, \tag{D.1}$$

and $g[n]$ is bounded for all n.

Proof. Since $g'[n] = g[n] - g[n-1]$, $g'[n] = \frac{1}{n}g'[n-1]$ implies $g[n] - g[n-1] = \frac{1}{n}(g[n-1] - g[n-2])$. By rearranging the terms we have (D.1). The boundedness can be proved by induction: $g[1]$ and $g[2]$ are bounded, because $g[0]$, $g[-1]$ are bounded. Assume that $g[k]$ and $g[k+1]$ are bounded, then by triangle inequality $|g[k+2]| \leq \left(1 + \frac{1}{k+2}\right)|g[k+1]| + \frac{1}{k+2}|g[k]|$ is also bounded. □

Proposition 3.

$$\|(\alpha * \mathbf{h}_F) \cdot (1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B\| \leq \|(1 - \alpha * \mathbf{h}_F) \cdot \Delta\widehat{\mathbf{f}}_B\|,$$

where the norm $\|\cdot\|$ is Frobenius-norm.

Proof. Note that for each pixel $\Delta\widehat{\mathbf{f}}_B(i,j)$, $|(\alpha * \mathbf{h}_F)(i,j) \cdot (1 - \alpha * \mathbf{h}_F)(i,j) \cdot \Delta\widehat{\mathbf{f}}_B(i,j)| \leq (1 - \alpha * \mathbf{h}_F)(i,j) \cdot \Delta\widehat{\mathbf{f}}_B(i,j)|$ because $0 \leq (\alpha * \mathbf{h}_F)(i,j) \leq 1$. Summing the squares of individual elements completes the proof. □

Bibliography

[1] Middleburry stereo dataset. Available at http://bj.middlebury.edu/~schar/stereo/web/results.php.

[2] M. Afonso, J. Bioucas-Dias, and M. Figueiredo. Fast image recovery using variable splitting and constrained optimization. *IEEE Trans. Image Process.*, 19(9):2345–2356, 2010.

[3] M. Allain, J. Idier, and Y. Goussard. On global and local convergence of half-quadratic algorithm. *IEEE Trans. Image Process.*, 15:1130–1142, 2006.

[4] N. Asada, H. Fujiwara, and T. Matsuyama. Seeing behind the scene: Analysis of photometric properties of occluding edges by the reversed projection blurring model. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(2):155–167, 1998.

[5] M. Aubailly, M. A. Vorontsov, G. W. Carhat, and M. T. Valley. Automated video enhancement from a stream of atmospherically-distorted images: the lucky-region fusion approach. In *Proceedings of SPIE*, volume 7463, 2009.

[6] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal of Imaging Science*, 2:183–202, 2009.

[7] J. Bect, L. Blang Feraud, G. Aubert, and A. Chambolle. A l_1-unified variational framework for image restoration. In *European Conf. Computer Vision*, pages 1–13, 2004.

[8] S. Belekos, N. Galatsanos, and A. Katsaggelos. Maximum a posteriori video super-resolution using a new multichannel image prior. *IEEE Trans. Image Process.*, 19:1451–1464, 2010.

[9] E. V. D. Berg and M. P. Friedlander. Probing the pareto frontier for basis pursuit solutions. *SIAM Journal of Scientific Computing*, 31(2):890–912, 2008.

[10] D. Bertsekas. Multiplier methods: A survey. *Automatica*, 12:133–145, 1976.

[11] D. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Academic Press, 1982.

[12] J. Bioucas-Dias and M. Figueiredo. A new TwIST: Two-step iterative shrinkage/ thresholding algorithm for image restoration. *IEEE Trans. Image Process.*, 16:2980–2991, 2007.

[13] J. Bioucas-Dias, M. Figueiredo, and J. Oliveira. Total variation-based image deconvolution: a majorization-minimization approach. In *IEEE Int. Conf. Acoustics, Speech and Signal Process. (ICASSP '06)*, volume 2, pages 861–864, May 2006.

[14] P. Blomgren, T. F. Chan, P. Mulet, and C. K. Wong. Total variation image restoration: Numerical methods and extensions. In *IEEE Int. Conf. Image Process. (ICIP '97)*, pages 384–387, 1997.

[15] S. Boyd, L. Xiao, and A. Mutapcic. Subgradient methods. Classnote of EE 392O, Stanford University, Oct 2003. Avaiable at http://www.stanford.edu/class/ee392o/subgrad_method.pdf.

[16] R. Bracewell. *The Fourier Transform and Its Applications*. McGraw-Hill, 1999.

[17] A. Chambolle. An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision*, 20(1-2):89–97, 2004.

[18] S. Chan. Constructing a sparse convolution matrix for shift varying image restoration problems. In *IEEE Int. Conf. Image Process. (ICIP '10)*, pages 3601–3604, 2010.

[19] S. Chan, R. Khoshabeh, K. Gibson, P. Gill, and T. Nguyen. An augmented Lagrangian method for total variation video restoration. *IEEE Trans. Image Process.*, 2011. To appear. Preprint available at http://videoprocessing.ucsd.edu/~stanleychan/deconvtv.

[20] S. Chan and T. Nguyen. LCD motion blur: modeling, analysis and algorithm. *IEEE Trans. Image Process.*, 20:2352–2365, Aug 2011.

[21] S. Chan, D. Vo, and T. Nguyen. Subpixel motion estimation without interpolation. In *IEEE Int. Conf. Acoustics, Speech and Signal Process. (ICASSP '10)*, pages 722–725, 2010.

[22] T. Chan, G. Golub, and P. Mulet. A nonlinear primal-dual method for total variation-based image restoration. *SIAM Journal Scientific Computing*, 20:1964–1977, 1999.

[23] T. Chan and C. Wong. total variation blind deconvolution. *IEEE Trans. Image Process.*, 7:370–375, 1998.

[24] Z. Chen, B. Abidi, D. Page, and M. Abidi. Gray-level grouping (GLG): an automatic method for optimized image contrast enhancement-part I. *IEEE Trans. Image Process.*, 15:2290 – 2302, 2006.

[25] Z. Chen, B. Abidi, D. Page, and M. Abidi. Gray-level grouping (GLG): an automatic method for optimized image contrast enhancement-part II. *IEEE Trans. Image Process.*, 15:2303 – 2314, 2006.

[26] S. Cho and S. Lee. Fast motion deblurring. *ACM Transactions on Graphics*, 28(5):1–8, 2009.

[27] S. Cho, Y. Matsushita, and S. Lee. Removing non-uniform motion blur from images. In *IEEE Int. Conf. Computer Vision (ICCV '07)*, pages 1–8, 2007.

[28] M. Choi, N. Galatsanos, and A. Katsaggelos. Multichannel regularized iterative restoration of motion compensated image sequences. *J. of Visual Communication and Image Representation*, 7(3):244–258, 1996.

[29] A. Criminisi, P. Perez, and K. Toyama. Object removal by exemplar-based inpainting. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR '03)*, 2003.

[30] S. Dai and Y. Wu. Motion from blur. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR '08)*, 2008.

[31] S. Dai and Y. Wu. Removing partial blur in a single image. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR '03)*, 2009.

[32] P. Davis. *Circulant Matrices*. Chelsea Pub Co, 1994.

[33] J. Eckstein and D. Bertsekas. On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55:293–318, 1992.

[34] E. Esser. Applications of Lagrangian-based alternating direction methods and connections to split Bregman. Technical Report 09-31, ULCA, 2009. Available at ftp://ftp.math.ucla.edu/pub/camreport/cam09-31.pdf.

[35] J. Fan, F. Liu, W. Bao, and H. Xia. Disparity estimation algorithm for stereo video coding based on edge detecetion. In *IEEE Int. Conf. Wireless Comm. and Signal Process. (WCSP '09)*, pages 1–5, 2009.

[36] S. Farsiu, M. Elad, and P. Milanfar. Multi-frame demosaicing and super-resolution of color images. *IEEE Trans. Image Process.*, 15:141–159, 2006.

[37] S. Farsiu, M. Elad, and P. Milanfar. Video-to-video dynamic super-resolution for grayscale and color sequences. *EURASIP J. Appl. Signal Process.*, pages 232–232, 2006.

[38] S. Farsiu, D. Robinson, M. Elad, and P. Milanfar. Advances and challenges in super-resolution. *Int. J. Imaging Systems and Tech., Special Issue on High Resolution Image Reconstruction*, 14(2):47–57, 2004.

[39] S. Farsiu, D. Robinson, M. Elad, and P. Milanfar. Fast and robust multi-frame super-resolution. *IEEE Trans. Image Process.*, 13:1327–1344, 2004.

[40] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W.T. Freeman. Removing camera shake from a single photograph. *ACM Transactions on Graphics, SIGGRAPH 2006 Conference Proceedings, Boston, MA*, 25:787–794, 2006.

[41] M. Figueiredo and R. Nowak. An EM algorithm for wavelet-basd image restoration. *IEEE Trans. Image Process.*, 12:906–916, 2003.

[42] D. L. Fried. Probability of getting a lucky short-exposure image through turbulence. *Journal of Optical Society of America*, 68:1651–1658, 1978.

[43] D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite-element approximations. *Computers and Mathematics with Applications*, 2:17–40, 1976.

[44] E. Gastal and M. Oliveira. Shared sampling for real-time alpha matting. *Computer Graphics Forum*, 29(2):575–584, May 2010. Proceedings of Eurographics.

[45] D. Geman and G. Reynolds. Constrained restoration and the recovery of discontinuities. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14:367–383, 1992.

[46] D. Geman and C. Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE Trans. Image Process.*, 4:932–946, 1995.

[47] R. Glowinski and P. Le Tallec. *Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics*. SIAM, Philadelphia, 1989.

[48] T. Goldstein and S. Osher. The split bregman algorithm for L1 regularized problems. Technical Report 08-29, UCLA, 2008.

[49] R. Gonzalez and R. Woods. *Digital Image Processing*. Prentice Hall, 2007.

[50] J. Goodman. *Statistical Optics*. Wiley-Interscience, 2000.

[51] J. Goodman. *Introduction to Fourier Optics*. Roberts & Company Publishers, 4 edition, 2004.

[52] R. Griesse and D. Lorenz. A semismooth Newton method for Tikhonov functionals with sparsity constraints. *Inverse Problems*, 24(3):035007, Jun 2008.

[53] E. Hale, W. Yin, and Y. Zhang. A fixed-point continuation method for L_1-regularized minimization with applications to compressed sensing. Technical report, Rice University, 2007.

[54] P. Hansen. *Rank-deficient and discrete ill-posed problems*. SIAM, 1998.

[55] P. Hansen, J. Nagy, and D. O' Leary. *Deblurring Images: Matrices, Spectra, and Filtering*. Fundamentals of Algorithms 3. SIAM, 2006. MATLAB code available at http://www2.imm.dtu.dk/~pch/HNO.

[56] S. Har-Noy and T. Nguyen. LCD motion blur reduction: A signal processing approach. *IEEE Trans. Image Process.*, 17:117–125, Feb 2008.

[57] Y. Huang, M. Ng, and Y. Wen. A fast total variation minimization method for image restoration. *SIAM Multiscale model and simulation*, 7:774–795, 2008.

[58] J. Idier. Convex half quadratic criteria and interacting auxiliary variables for image restoration. *IEEE Trans. Image Process.*, 10:1001–1009, 2001.

[59] B. Jähne. *Spatio-temporal image processing: theory and scientific applications*. Springer, 1993.

[60] J. Jia. Single image motion deblurring using transparency. In *IEEE Int. Conf. Computer Vision (ICCV '07)*, 2007.

[61] B. Kim. *Numerical Optimization Methods for Image Restoration*. PhD thesis, Stanford University, Dec 2002. Available at http://www.stanford.edu/group/SOL/dissertations/kimthesis.pdf.

[62] M. Klompenhouwer and L. Velthoven. Motion blur reduction for liquid crystal displays: Motion compensated inverse filtering. In *Proceedings of SPIE-IS&T Electronic Imaging*. SPIE, 2004.

[63] D. Krishnan, P. Lin, and A. Yip. A primal-dual and active-set method for non-negativity constrained total variation deblurring problems. Technical report, University of California, Los Angeles, 2007. Available at ftp://ftp.math.ucla.edu/pub/camreport/cam07-03.pdf.

[64] D. Kundur and D. Hatzinakos. A novel blind deconvolution scheme for image restoration using recursive filtering. *IEEE Trans. Signal Process.*, 46:375–390, 1998.

[65] K. Lee, J. Nagy, and L. Perrone. Iterative methods for image restoration: A matlab object oriented approach. Technical report, Emory University, 2002. Available at http://www.mathcs.emory.edu/~nagy/RestoreTools/index.html.

[66] Y. Lee and T. Nguyen. Fast one-pass motion compensated frame interpolation in high-definition video processing. In *IEEE Int. Conf. Image Process. (ICIP '09)*, pages 369–372, November 2009.

[67] A. Levin. Blind motion deblurring using image statistics. In *Advances in Neural Information Processing Systems (NIPS '06)*, 2006. Available at http://www.wisdom.weizmann.ac.il/~levina/papers/levin-deblurring-nips06.pdf.

[68] A. Levin, D. Lischinski, and Y. Weiss. A closed form solution to natural image matting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(2):228–242, 2008.

[69] C. Li. *An Efficient Algorithm For Total Variation Regularization with Applications to the Single Pixel Camera and Compressive Sensing*. PhD thesis, Rice University, 2009. Available at http://www.caam.rice.edu/~optimization/L1/TVAL3/tval3_thesis.pdf.

[70] C. Liu and W. Freeman. Noise estimation from a single image. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR '06)*, pages 901–908, 2006.

[71] C. Liu, R. Szeliski, S. Kang, C. Zitnick, and W. Freeman. Automatic estimation and removal of noise from a single image. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30:299–314, 2008.

[72] B. Lucas. *Generalized image matching by the method of differences*. PhD thesis, Carnegie Mellon University, 1984.

[73] L. Lucy. An iterative technique for the rectification of observed distributions. *Astronomical Journal*, 79:745, 1974.

[74] A. Marquina and S. Osher. Explicit algorithms for a new time dependent model based on level set motion for nonlinear deblurring and noise removal. *SIAM Journal Scientific Computing*, 22:387–405, 1999.

[75] M. McGuire, W. Matusik, H. Pfister, J. Hughes, and F. Durand. Defocus video matting. In *ACM SIGGRAPH*, 2005.

[76] V. Mesarovic, N. Galatsanos, and A. Katsaggelos. Regularized constrained total least squares image restoration. *IEEE Trans. Image Process.*, 4:1096–1108, 1995.

[77] J. Nagy and D. O'Leary. Restoring images degraded by spatially variant blur. *SIAM Journal on Scientific Computing*, 19(4), 1998.

[78] R. Neelamani, H. Choi, and R. Barabiuk. ForWaRD: Fourier-Wavelet regularized deconvolution for ill-conditioned systems. *IEEE Trans. Signal Process.*, 52:418–433, 2004.

[79] Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103:127–152, 2005.

[80] M. Ng. *Iterative Methods for Toeplitz Systems*. Oxford University Press, Inc, 2004.

[81] M. Ng, H. Shen, E. Lam, and L. Zhang. A total variation regularization based super-resolution reconstruction algorithm for digital video. *EURASIP Journal on Advances in Signal Processing*, page 74585, 2007.

[82] N. Nguyen, P. Milanfar, and G. Golub. A computationally efficient image super-resolution algorithm. *IEEE Trans. Image Process.*, 10:573–583, Apr 2001.

[83] M. Nikolova and M. Ng. Analysis of half-quadratic minimization methods for signal and image recovery. *SIAM Journal of Scientific Computing*, 27:937–966, 2005.

[84] J. Oh, S. Ma, and C. Kuo. Disparity estimation and virtual view synthesis from stereo video. In *IEEE Int. Symposium on Circuits and Systems (ISCAS '07)*, pages 993–996, 2007.

[85] S. Osher and L. Rudin. Feature-oriented image enhancement usingshock filters. *SIAM J. Numerical Analysis*, 27:919–940, 1990.

[86] C. Paige and M. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematmal Software*, Vol 8(1):43–71, March 1982.

[87] M. Powell. A method for nonlinear constraints in minimization problems. In Fletcher, editor, *Optimization*, pages 283–298. Academic Press, 1969.

[88] K. Rank, M. Lendl, and R. Unbehauen. Estimation of image noise variance. *IEE Proc., Vis. Image Process.*, 146:80–84, 1999.

[89] C. Rhemann, C. Rother, J. Wang, M. Gelautz, P. Kohli, and P. Rott. A perceptually motivated online benchmark for image matting. In *IEEE CVPR*, June 2009.

[90] C. Richardt, D. Orr, I. Davies, A. Criminisi, and N. Dodgson. Real-time spatiotemporal stereo matching using the dual-cross-bilateral grid. In *Proceedings of ECCV*, 2010.

[91] R. Rockafellar. Augmented Lagrangians and applications of the proximal point algorithm in convex programming. *Math. of Operations Research*, 1:97–116, 1976.

[92] R. Rockafellar. Monotone operators and proximal point algorithm. *SIAM Journal on Control and Optimization*, 14:877–898, 1976.

[93] M. Roggemann and B. Welsh. *Imaging through turbulence*. CRC Press, 1996.

[94] L. Rudin and S. Osher. Total variation based restoration with free local constraints. In *IEEE Int. Conf. Image Process. (ICIP '94)*, pages 31–35, 1994.

[95] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[96] M. Saunders. PDCO: Primal-dual method for optimization with convex objectives. Technical report, System Optimization Lab, Stanford University, 2002.

[97] Q. Shan, J. Jia, and A. Agarwala. High-quality motion deblurring from a single image. *ACM Transactions on Graphics (SIGGRAPH '08)*, 27(3), 2008.

[98] E. Shechtman, Y. Caspi, and M. Irani. Space-time super-resolution. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27:531–545, 2005.

[99] M. Shimizu, S. Yoshimura, M. Tanaka, and M. Okutomi. Super-resolution from image sequence under influence of hot-air optical turbulence. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR '08), 2008*, 2008.

[100] N. Shor. *Minimization Methods for Non-differentiable Functions*. Springer Series in Computational Mathematics. Springer, 1985.

[101] R. Szeliski and J. Coughlan. Spline-based image registration. *International Journal of Computer Vision*, 22(93):199–218, 1997.

[102] H. Takeda, S. Farsiu, and P. Milanfar. Deblurring using regularized locally adaptive kernel regression. *IEEE Trans. Image Process.*, 17(4):550–563, April 2008.

[103] H. Takeda, P. Milanfar, M. Protter, and M.Elad. Super-resolution without explicit subpixel motion estimation. *IEEE Trans. Image Process.*, 18(9):1958–1975, Septmeber 2009.

[104] M. Tao and J. Yang. Alternating direction algorithms for total variation deconvolution in image reconstruction. Technical Report TR0918, Nanjing University, China, 2009. Available at http://www.optimization-online.org/DB_FILE/2009/11/2463.pdf.

[105] P. Tseng. Further applications of a splitting algorithm to decomposition in variational inequalities and convex programming. *Mathematical Programming*, 48:249–263, 1990.

[106] P. Tseng. Applications of a splitting algorithm to decomposition in convex programming and variational inequalities. *SIAM Journal of Control and Optimization*, 29:119–138, 1991.

[107] V. Vezhnevets and V. Konouchine. Grow-cut - interactive multi-label n-d image segmentation. In *Proceedings of Graphicon*, page 150156, 2005.

[108] C. Vogel. *Computational Methods for Inverse Problems*. SIAM, Philadelphia, PA, 2002.

[109] C. Vogel and M. Oman. Iterative methods for total variation denoising. *SIAM Journal Scientific Computing*, 17:227–238, 1996.

[110] J. Wang and M. Cohen. Image and video matting: A survey. *Foundations and Trends in Computer Graphics and Vision*, 3(2), 2007.

[111] Y. Wang, J. Ostermann, and Y. Zhang. *Video Processing and Communications*. Prentice Hall, 2002.

[112] Y. Wang, J. Yang, W. Yin, and Y. Zhang. An efficient TVL1 algorithm for deblurring multichannel images corrupted by impulsive noise. Technical report, CAAM, Rice University, Sep 2008.

[113] Y. Wexler, E. Shechtman, and M. Irani. Space-time completion of video. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29:1–14, March 2007.

[114] S. Wright, R. Nowak, and M. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Trans. Signal Process.*, 57:2479–2493, July 2009.

[115] C. Wu and X. Tai. Augmented lagrangian method, dual methods, and split bregman iteration for ROF, vectorial TV and high order methods. Technical Report CAM-09-05, UCLA, 2009.

[116] L. Xu and J. Jia. Two-phase kernel estimation for robust motion deblurring. In *European Conference on Computer Vision (ECCV' 10)*, 2010. Available at http://www.cse.cuhk.edu.hk/~leojia/projects/robust_deblur/.

[117] J. Yang and Y. Zhang. Alternating direction algorithms for l1 problems in compressive sensing,. Technical report, Rice University, 2009. Available at www.caam.rice.edu/~yzhang/reports/tr0937.

[118] K. Yoon and I. Kweon. Locally adaptive support-weight approach for visual correspondence search. In *Proceedings of IEEE CVPR*, 2005.

[119] X. Zhu and P. Milanfar. Image reconstruction from videos distorted by atmospheric turbulence. *Visual Information Processing and Communication*, 7543(1):75430S, 2010. Available at http://link.aip.org/link/?PSI/7543/75430S/1.