Motion Blur Removal from Photographs

by

Taeg Sang Cho

B.S., Electrical Engineering and Computer Science,
Korea Advanced Institute of Science and Technology, 2005

S.M., Electrical Engineering and Computer Science,
Massachusetts Institute of Technology, 2007

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in Electrical Engineering and Computer Science
at the Massachusetts Institute of Technology

September 2010

Author: ____

Department of Electrical Engineering and Computer Science
August 20, 2010

Certified by: ____

William T. Freeman
Thesis Supervisor

Accepted by: ____

Terry P. Orlando
Chair, Department Committee on Graduate Students

Motion blur removal from photographs

by Taeg Sang Cho

Submitted to the Department of Electrical Engineering and Computer Science
on August 20, 2010, in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Electrical Engineering and Computer Science

Abstract

One of the long-standing challenges in photography is motion blur. Blur artifacts are generated from relative motion between a camera and a scene during exposure. While blur can be reduced by using a shorter exposure, this comes at an unavoidable trade-off with increased noise. Therefore, it is desirable to remove blur computationally.

To remove blur, we need to (i) estimate how the image is blurred (i.e. the blur kernel or the point-spread function) and (ii) restore a natural looking image through deconvolution. Blur kernel estimation is challenging because the algorithm needs to distinguish the correct image–blur pair from incorrect ones that can also adequately explain the blurred image. Deconvolution is also difficult because the algorithm needs to restore high frequency image contents attenuated by blur. In this dissertation, we address a few aspects of these challenges.

We introduce an insight that a blur kernel can be estimated by analyzing edges in a blurred photograph. Edge profiles in a blurred image encode projections of the blur kernel, from which we can recover the blur using the inverse Radon transform. This method is computationally attractive and is well suited to images with many edges. Blurred edge profiles can also serve as additional cues for existing kernel estimation algorithms. We introduce a method to integrate this information into a maximum-a-posteriori kernel estimation framework, and show its benefits.

Deconvolution algorithms restore information attenuated by blur using an image prior that exploits a heavy-tailed gradient profile of natural images. We show, however, that such a sparse prior does not accurately model textures, thereby degrading texture renditions in restored images. To address this issue, we introduce a content-aware image prior that adapts its characteristics to local textures. The adapted image prior improves the quality of textures in restored images. Sometimes even the content-aware image prior may be insufficient for restoring rich textures. This issue can be addressed by matching the restored image's gradient distribution to its original image's gradient distribution, which is estimated directly from the blurred image. This new image deconvolution technique called iterative distribution reweighting (IDR) improves the visual realism of reconstructed images.

Subject motion can also cause blur. Removing subject motion blur is especially challenging because the blur is often spatially variant. In this dissertation, we address a restricted class of subject motion blur: the subject moves at a constant velocity locally. We design a new computational camera that improves the local motion estimation and, at the same time, reduces the image information loss due to blur.

Thesis Supervisor: William T. Freeman

Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

Contents

List of Figures

Chapter 1

Introduction

MOTION blur is one of the salient sources of degradation in photographs. Although motion blur can sometimes be desirable for artistic purposes, it often severely limits the image quality. Blur artifacts result from relative motion between a camera and a scene during exposure. While blur can be reduced using a faster shutter speed, this comes with an unavoidable trade-off with increased noise.

One source of a motion blur is camera shake. When a camera moves during exposure, it blurs the captured image according to its trajectory. We can mitigate the camera shake blur by using a mechanical image stabilization hardware [40]. However, a camera shake can be too large for assistive devices to accommodate when a camera takes a long exposure shot of a dark scene and/or when a camera uses a telephoto lens. A second source of blur is a movement of objects in the scene, and this type of blur is harder to avoid. Therefore it is often desirable to remove blur computationally.

Motion blur removal, often called motion deblurring or blind deconvolution, is challenging in two aspects. The first challenge is estimating blur kernels, or point-spread functions (PSF), from blurred images. Because many blur–image pairs can explain the observed blurry image, kernel estimation is a difficult problem. Blur estimation can be especially difficult if the blur is spatially variant, for instance due to a dynamic scene or a camera rotation. The second challenge is removing the blur to recover a blur-free image. Motion blur averages neighboring pixels and attenuates high frequency information of the scene. Consequently, recovering a blur-free image is an ill-posed problem which needs to be addressed by deblurring systems or algorithms.

This thesis explores both hardware and software solutions to address a few of these challenges. We introduce (i) a spatially-invariant blur kernel estimation method using blurred edge profiles, (ii) an adaptive image prior for improving the rendition of textures in restored images,

(iii) an image deconvolution technique that matches gradient distributions to improve visual realism of textures, and (iv) a new computational camera that near-optimally captures the image information of moving objects.

1.1 Overview of techniques and contributions

This section provides a preview of techniques and contributions.

1.1.1 Chapter 2: Blur kernel estimation from blurred edge profiles

We present a method to recover a spatially invariant blur kernel from a single blurry photograph. The key idea is that blur kernel projections can be estimated by analyzing blurred edge profiles. These projections are also known as the *Radon Transform*. Blur kernel projections are useful because we can apply the inverse Radon transform to restore the blur kernel. This method is computationally attractive because we do not need to estimate the latent image in order to iteratively refine the kernel and because most of the computation is performed at the scale of the blur kernel. Although this technique applies only to images with a sufficient number of straight edges in many orientations, these encompass a large set of images including many man-made scenes. Kernel projections can also provide additional cues to improve the kernel estimation performance of existing algorithms. We propose a method to integrate kernel projection information in a MAP based kernel estimation framework.

1.1.2 Chapter 3: A content-aware image prior for image restoration

Even if we could accurately estimate the blur kernel, restoring a blur-free image from a blurry photograph is still challenging because we lose high frequency information during a blurry observation process. To "fill-in" the missing information, we often exploit prior knowledge about natural images. One of the most popular image priors is a heavy-tailed gradient distribution of natural images. A MAP estimator, when used with a heavy-tailed, or sparse, gradient prior, reconstructs images with piecewise smooth characteristics. While a sparse gradient prior removes ringing and noise artifacts, it also removes mid-frequency textures, degrading visual quality. We can attribute such degradations to imposing an incorrect image prior. As is seen in Fig. 1.1, the gradient profile in fractal-like textures, such as trees, is close to a Gaussian distribution, therefore a sparse gradient prior would penalize small gradients from such regions, over-smoothing textures.

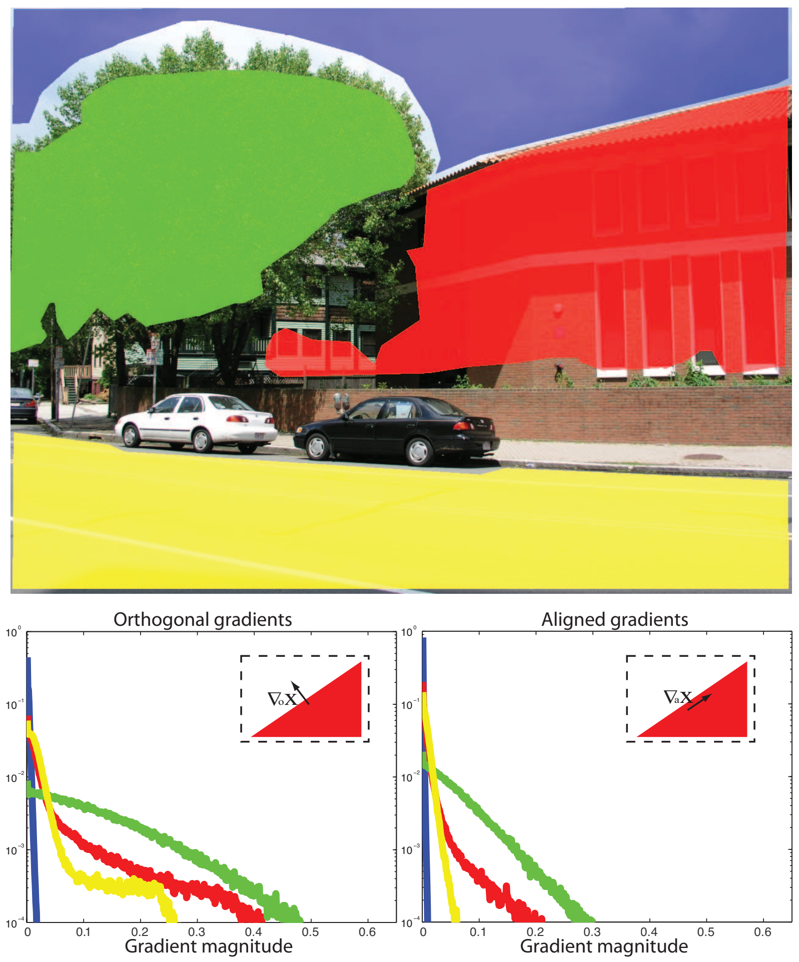


Figure 1.1: *The gradient profile of natural images is often used as an image prior in image restoration tasks. Oftentimes, we use a single heavy-tailed gradient profile as an image prior for the entire image. However, this figure shows that gradient profiles differ significantly in response to underlying textures. This observation suggests that we should adapt the image prior to the image content. This so called a content-aware image prior improves the visual realism of restored images.*

To address this issue, we introduce an image prior that *adapts* to local texture. We adapt the prior to both low-level local structures as well as mid-level textural characteristics. We demonstrate improvements on deblurring and denoising tasks.

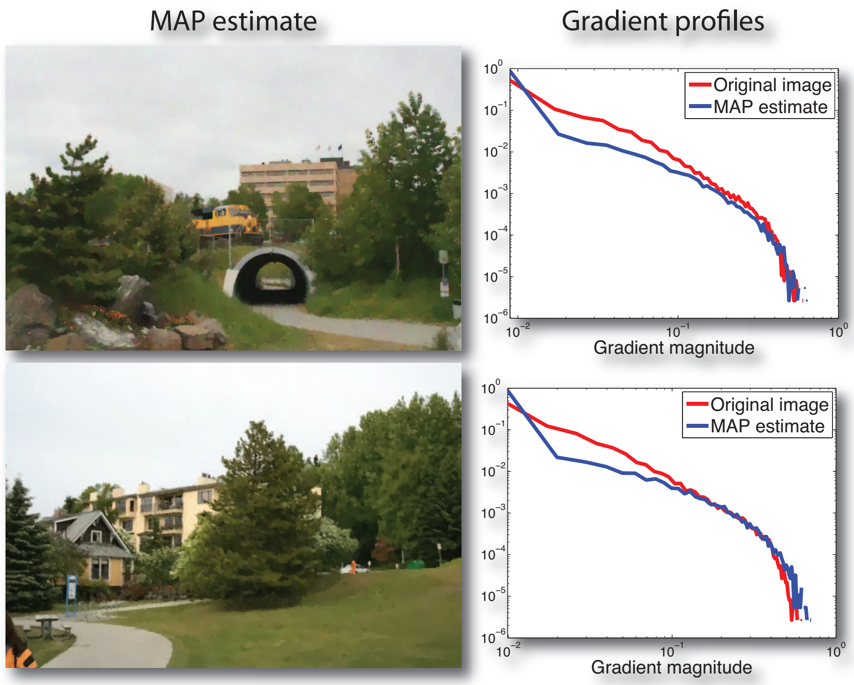


Figure 1.2: *The gradient distribution of images restored using a MAP estimator can differ from that of the original images, and this manifests itself as smoothed textures. In Chapter 4, we present an alternative deconvolution method that matches the reconstructed image's gradient distribution to its reference distribution (i.e. the gradient distribution of the original image) to restore visually pleasing textures.*

■ 1.1.3 Chapter 4: Image restoration by matching gradient distributions

Even with a content-aware image prior, a MAP estimator does not always reliably reconstruct rich textures. We present an alternative image restoration method called *iterative distribution reweighting (IDR)* to improve the rendition of rich textures. IDR imposes a global constraint on image gradients: the restored image should have a gradient distribution close to a reference distribution. As is explored in Chapter 3, a reference distribution not only varies from one image to another, but also within an image depending on texture. Therefore, we estimate a reference distribution directly from an input image for each texture segment. We show through experiments that IDR is able to restore rich mid-frequency textures that are visually more appealing than MAP estimates. User studies support our claim that IDR improves the visual realism of reconstructed images compared to MAP estimates.

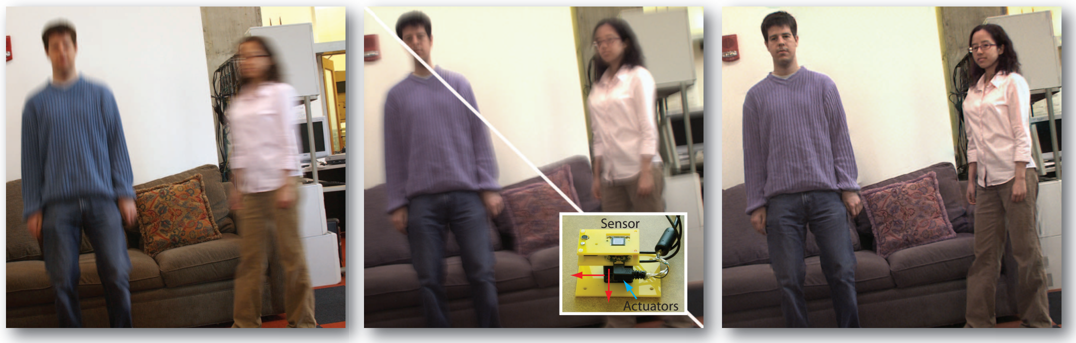

Sensor
Actuators
Static camera image
Orthogonal parabolic camera: input
Orthogonal parabolic camera: deblurred output

Figure 1.3: *In Chapter 5, we address spatially variant motion blurs induced by subject motions. We address two challenges associated with motion blur removal, namely the blur kernel estimation and the reduction of information loss, by developing a new computational camera that takes two consecutive images of a moving scene.* **Left:** *An image captured by a static camera.* **Middle:** *Our solution takes two consecutive images of a scene using a parabolic camera moving in two orthogonal directions.* **Right:** *The restored image.*

■ 1.1.4 Chapter 5: Orthogonal parabolic exposures for motion deblurring

In this chapter, we address spatially variant blur induced by moving objects. Removing subject motion blur is challenging because one has to *locally* estimate the motion. Even if the motion is successfully identified, blur inversion can still be unstable because the blur kernel attenuates high frequency image content.

We present a computational camera to address above challenges. We assume that the object is moving at constant velocities in arbitrary 2D directions parallel to the imaging plane. This assumption is often satisfied when the exposure time is relatively short. Our solution captures two images of a scene with a parabolic camera motion in two orthogonal directions. We show that this strategy near-optimally preserves the image content of moving objects, which allows for a stable blur inversion. Taking two images of a scene also helps us estimate spatially varying object motions. We present a prototype camera and demonstrate successful motion deblurring on real world motions.

■ 1.2 Other work not included in the thesis

During the PhD studies, I also had a chance to contribute to disciplines other than deblurring. In collaboration with Dr. Hensin Tsao and Prof. William Freeman, I developed an automatic skin mole localization method [16] that can be used as a front-end system for automatic melanoma detection. With Prof. Shai Avidan and Prof. William Freeman, I developed a new image editing framework called "The Patch Transform" [17, 19]. We extended this framework to solving image jigsaw puzzles consisting of square pieces [18].

1.3 Notes

Parts of the work presented in this thesis appeared previously at 2010 IEEE International Conference on Computational Photography (ICCP) [21] and 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) [20].

This work was supported in part by Samsung Scholarship Foundation, by NGA NEGI-1582-04-0004, by ONR-MURI Grant N00014-06-1-0734, by gift from Microsoft, Google, Adobe and Quanta.

Chapter 2

Blur kernel estimation from blurred edge profiles

■ 2.1 Introduction

MANY challenges in deblurring stem from the severely under-constrained nature of the problem: many image–blur pairs can explain the blurred image. Fortunately, most image–blur pairs are implausible because the corresponding images contain ringing and noise; kernels are not continuous. Therefore, existing deblurring techniques exploit prior knowledge about natural images and blur kernels to distinguish the correct solution pair from incorrect ones. Although this prior knowledge is effective, it is often not strong enough to reliably distinguish the correct solution from others. In this chapter, we present additional cues to exploit in blur kernel estimation.

Our algorithm estimates a blur kernel by analyzing blurred edges. Intuitively, edges along different orientations are affected differently by blur, therefore we can consider different edge profiles as "signatures" of a blur. We formalize this intuition and show how to use blurred edges to recover the *Radon transform* of the blur kernel, that is, a set of projections of the blur kernel in different orientations. We can restore the blur kernel by inverting the estimated Radon transform. Advantages of our method are that (i) we do not deconvolve the blurred image to refine the estimated kernel and that (ii) we perform a bulk of the computation at the size of the kernel, which is often considerably smaller than the image. We demonstrate that our approach is well-suited for scenes with numerous edges such as man-made environments.

Even if a blurred image does not contain many edges in different orientations, we can still exploit kernel projections. We introduce a method to integrate Radon transform constraints in a maximum-a-posteriori kernel estimation framework to improve the kernel estimation performance. This alternative method is computationally more expensive, but it is more stable than simply inverting the Radon transform.

Contributions We can summarize the contributions of this chapter as follows:

- We demonstrate that the blur kernel can be estimated from blurred edge profiles using the inverse Radon transform.
- We describe a method to detect stable edges for use in kernel estimation.
- We introduce a method to integrate blur kernel projection constraints in a maximum-a-posteriori estimation framework to jointly estimate the blur kernel and the sharp image.

2.1.1 Related work

In this work, we consider spatially invariant blur. Spatially invariant blur arises when the scene is static and the camera undergoes a small out-of-plane rotation or a translation (for a constant-depth scene.) A spatially invariant blur model is popular because one can exploit a simple global convolution model to describe an image formation process. Even with the spatially invariant blur assumption, however, estimating the correct blur from a single image is a challenging task due to inherent ambiguities: the observed blurry input image can be interpreted as a blurry image of a sharp scene or a sharp image of a blurry scene. This ambiguity can be address by taking multiple photos, each of which contains different blur [9, 13, 15, 52, 64, 90]. Taking images with modified cameras [6, 79] can also improve the kernel estimation performance.

Oftentimes, however, we are provided only with a single blurry image from a conventional camera, therefore a single-image blur kernel estimation problem received a lot of attention. To resolve the inherent ambiguity, different assumptions on blur kernels and natural images have been incorporated. Fergus *et al.* [31] exploit the knowledge that a histogram of gradients from natural images exhibits a heavy-tailed profile and that a histogram of intensities in blur kernels is sparse. They use a variational inference technique to first estimate a blur kernel, which is then used to restore a blur-free image using the Richardson-Lucy deconvolution algorithm [54, 65]. Shan *et al.* [74] introduce a local prior, in addition to a sparse gradient prior of natural images, to detect and smooth surfaces. Cai *et al.* [10] assume that a blur kernel should be sparse in the Curvelet domain and an image should be sparse in the Framelet domain. These techniques solve a large system of equations to find the sharp image and/or the blur kernel that satisfy the observation model while conforming to prior knowledge about blur and natural images.

Several prior work explicitly leverage blurred edges to estimate blur, as in our method. Jia [41] estimates an alpha matte from user-selected edges, and subsequently estimates the blur kernel from the matte by minimizing a non-linear cost function consisting of an image observation term as well as an image prior. Joshi *et al.* [42] predict sharp edges directly from a blurry photo and estimate the blur kernel *given* the location of predicted sharp edges. Their edge prediction scheme assumes that the blur kernel is uni-modal; Cho *et al.* [14] extend Joshi *et al.* [42] in a multi-scale manner to estimate more general blur kernels with multiple modes. Cho *et al.* [14] reduce the computation by deblurring only edges in the gradient domain: their GPU implementation runs in near real-time. Levin *et al.* [50] compare the performance of several single-image blind deconvolution algorithms, and empirically show that the algorithm introduced by Fergus *et al.* [31] is the state-of-the-art in single-image blur kernel estimation [1].

2.2 Kernel estimation from edges

We model the image formation as a convolution of a blur kernel k and a sharp latent image I:

$$B = k \otimes I + n \tag{2.1}$$

where B is an observed, blurry image and n is input noise. Our goal is to reconstruct a sharp, natural-looking latent image I from the observed image B.

2.2.1 The Radon transform and blurred line profiles

We briefly review the Radon transform for two-dimensional signals and illustrate how it is related to blur. For an in-depth review of the Radon transform, we refer the readers to [25, 81]. The Radon transform of a signal $f(x,y)$ is an integral of the signal along a straight line:

$$\phi_\theta^f(\rho) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta\left(\rho - x\cos(\theta) - y\sin(\theta)\right)dxdy \tag{2.2}$$

where θ is the orientation of the straight line that we integrate over and ρ is the offset of that line from the origin of the $x-y$ coordinate (See Fig. 2.1). ϕ_θ^f can be viewed as a projection of the signal f along the direction orthogonal to orientation θ. If we take enough projections of the signal f in all possible orientations, asymptotically we can recover the original signal f using the inverse Radon transform [81].

[1]Levin *et al.* [50] do not consider Cho *et al.* [14].

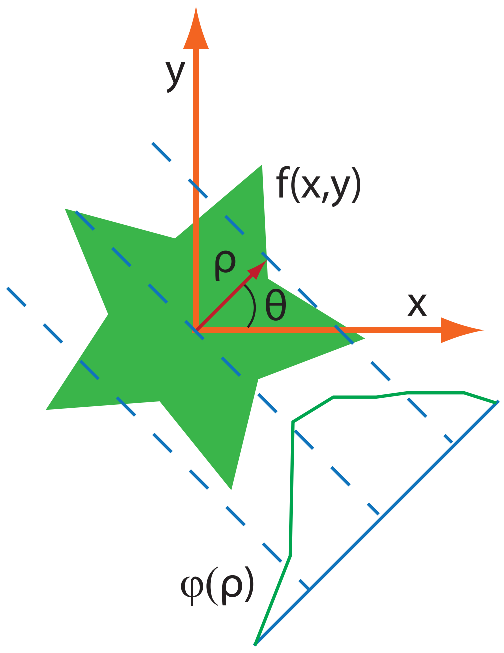


Figure 2.1: *The Radon transform $\phi_\theta^f(\rho)$ of a signal f (i.e. the star) is an integral of the signal along the line $\rho = x\cos(\theta) + y\sin(\theta)$ (i.e the dotted line).*

Interestingly, we can relate the Radon transform to our imaging model in Eq. 2.1. The imaging model in Eq. 2.1 can be expressed in the continuous domain:

$$B(\rho_x, \rho_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} k(x,y)I(\rho_x - x, \rho_y - y)dxdy \tag{2.3}$$

If the latent image is an ideal straight line along orientation θ, we can parameterize the latent image I as $\delta(\rho - x\cos(\theta) - y\sin(\theta))$, where $\rho = \sqrt{\rho_x^2 + \rho_y^2}$. Therefore,

$$\begin{aligned} B_L(\rho_x, \rho_y) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} k(x,y)\delta(\rho - x\cos(\theta) - y\sin(\theta))\,dxdy \\ &= \phi_\theta^k(\rho) \end{aligned} \tag{2.4}$$

In other words, every orthogonal slice of a blurred line, taken along the orientation θ, is a projection of the blur kernel $\phi_\theta^k(\rho)$. Fig. 2.2 shows graphically that $B_L(\rho_x, \rho_y)$, evaluated at a fixed point $[\rho_x, \rho_y]$, is a sum of intersections between the blur kernel and the line: $B_L(\rho_x, \rho_y)$ is a projection of the blur kernel.

To illustrate this concept numerically, we blur lines in different orientations and compare

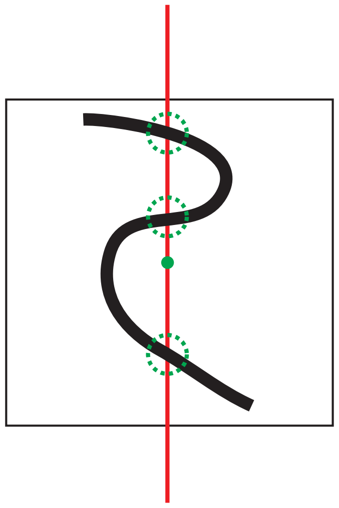


Figure 2.2: *The value of the convolved image at the green dot is a sum of intersections of the blur kernel (the black line) and the line (the red line). The dotted green circles indicate the intersections.*

orthogonal slices of blurred lines to explicit projections of the blur kernel in those orientations. Fig. 2.3 shows the results. As expected, the orthogonal line profiles are very close to explicit kernel projections.

This relationship between the Radon transform and blurred line profiles implies that we can estimate blur kernel projections and use them for kernel estimation if we can detect blurred lines. However, detecting lines reliably from a blurred image is a challenging problem, especially when the blur is multi-modal. Furthermore, many lines in images are not ideal: each line has a finite width, therefore blurred line profiles are no longer perfect projections of the blur kernel [2].

Fortunately, a blurred edge can provide information similar to a blurred line. An ideal binary step edge with orientation θ can be modeled as an integral of a line along θ:

$$e(\rho) = \int_{-\infty}^{\rho} \delta\left(\tau - x\cos(\theta) - y\sin(\theta)\right) d\tau \tag{2.5}$$

Therefore, a blurred edge profile can be modeled as follows:

[2]We can show that a slice of a blurred line of a finite width is a projection of the kernel convolved with a box filter of that width.

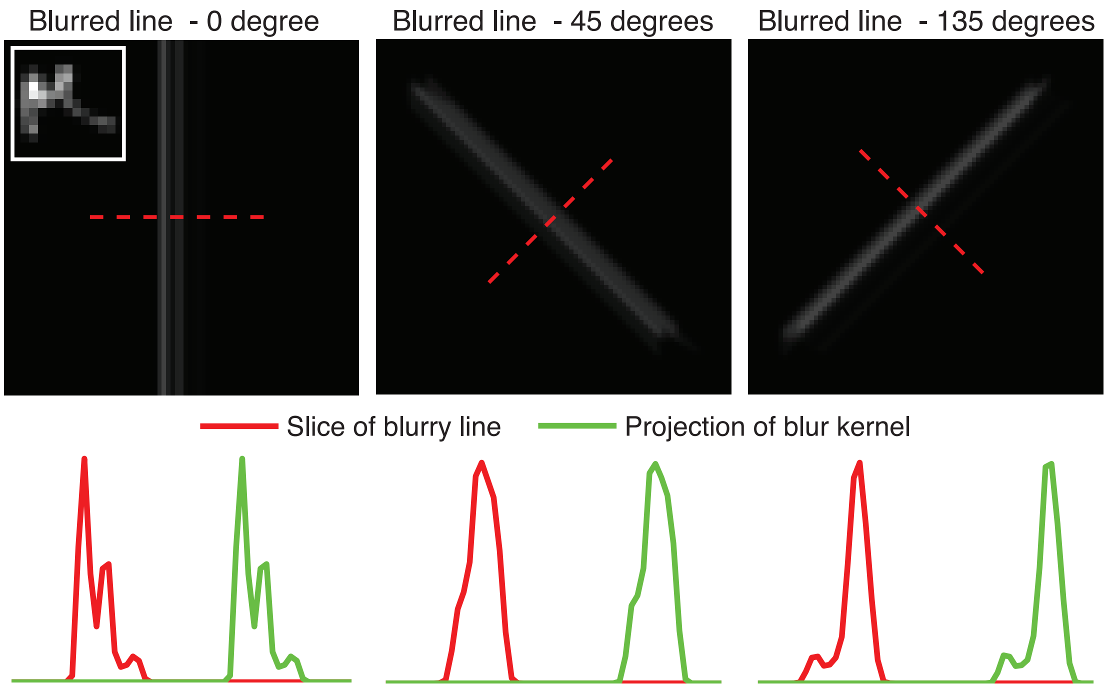


Figure 2.3: *We show experimentally that a slice (i.e. the dotted red line) orthogonal to a blurred line is the same as an explicit projection of the blur kernel along the line.*

$$
\begin{aligned}
& B_E(\rho_x, \rho_y) \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} k(x,y) \int_{-\infty}^{\rho} \delta\left(\tau - x\cos(\theta) - y\sin(\theta)\right) d\tau dx dy \\
&= \int_{-\infty}^{\rho} \left\{ \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} k(x,y)\delta\left(\tau - x\cos(\theta) - y\sin(\theta)\right) dx dy \right\} d\tau \\
&= \int_{-\infty}^{\rho} \phi_{\theta}^{k}(\tau) d\tau
\end{aligned}
\tag{2.6}
$$

In other words, an orthogonal slice of a blurred edge is an integral of a blurred line profile. Therefore, blurred line profiles can be recovered by differentiating blurred edge profiles.

Extracting edge profiles from color images To extract blur kernel projections from a color image, we assume a color-line image model [59] within a local neighborhood of an edge: a local region in a natural image has two dominant colors. Two dominant colors for a given pixel are estimated by averaging pixels at two ends of the slice (Fig. 2.4). Given the two dominant colors W, Z, we can represent each pixel on the orthogonal slice c_i as a linear combination of W, Z:

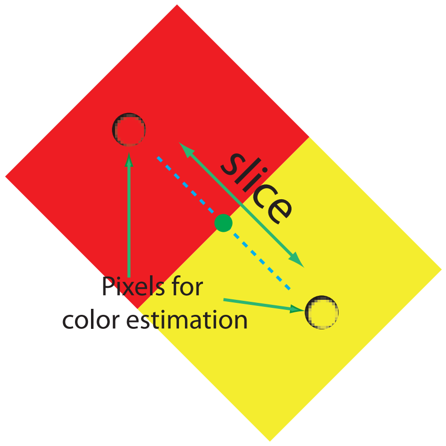


Figure 2.4: *To find two dominant colors on either side of an edge, we average the pixels separated from the edge by* 3/4 *the size of the blur kernel.*

$$c_i = \alpha_i W + (1-\alpha_i)Z \tag{2.7}$$

We use α's as the blurred binary edge slice.

■ 2.2.2 Recovering the blur kernel from its projections

Recovering a two-dimensional signal from its one-dimensional projections, also known as the inverse Radon transform, has been studied extensively in literature [25, 81]. In this work, we view the inverse Radon transform as maximizing the posterior probability of the blur kernel k given the observed image B. This framework allows us to incorporate prior knowledge on k.

From the Bayes' rule,

$$p(k|B) \propto p(B|k)p(k) \tag{2.8}$$

We directly model each term as follows. We model the likelihood term $p(B|k)$ from the constraint that explicit projections of the blur kernel k should match its projections ϕ_{θ_i} *estimated* from blurred edge slices:

$$\begin{aligned} p(B|k) &= \prod_{i=1}^{N} p(\phi_{\theta_i}|k) \\ &\propto \exp\left(-\frac{\sum_{i=1}^{N} \|\phi_{\theta_i} - R_{\theta_i}k\|^2}{2\eta_p^2}\right) \end{aligned} \tag{2.9}$$

where i indexes edge samples, N is the total number of edge samples, R_{θ_i} is a projection operator along i^{th} sample's dominant orientation θ_i, and η_p^2 is the variance of observation noise. We set the noise variance η_p^2 as $(2+\alpha)\eta_n^2$ where η_n^2 is the variance of the imaging noise. The factor of 2 results from differentiating edge slices (see Eq. 2.6) and $\alpha = 1$ models orientation estimation error, which increases with the image noise level.

The number of edge samples (i.e. N in Eq. 2.9) affects the speed of our algorithm: N depends on the size of the image and/or the image content. We observe, however, that having many edge samples in similar orientations is beneficial mostly in terms of reducing noise of the projection along that orientation. In light of this observation, we average out the noise "off-line" in order to accelerate the kernel reconstruction. In particular, we approximate $\sum_i^N \|\phi_{\theta_i} - R_{\theta_i}k\|^2$ as a sum over binned angles:

$$\sum_{i=1}^{N} \|\phi_{\theta_i} - R_{\theta_i}k\|^2 \approx \sum_{j=1}^{360} w_j \|\tilde{\phi}_{\theta_j} - R_{\theta_j}k\|^2 \tag{2.10}$$

where j indexes angles in steps of 1^o, $\tilde{\phi}_{\theta_j}$ is the average of kernel projections that have the same binned orientation θ_j, and w_j is the number of samples that have the same binned orientation θ_j. This approximation allows us to efficiently recover the kernel even for images with many edge samples.

In addition to kernel projection constraints, we incorporate the knowledge that intensity profiles of blur kernels, as well as gradient profiles of blur kernels, are sparse:

$$p(k) \propto \exp\{-(\lambda_1\|k\|^{\gamma_1} + \lambda_2\|\nabla k\|^{\gamma_2})\} \tag{2.11}$$

We use the same parameters for all experiments, determined through cross-validation: $\lambda_1 = 1.5, \gamma_1 = 0.9, \lambda_2 = 0.1, \gamma_2 = 0.5$.

Given this model, we can recover the blur kernel by minimizing the negative log-posterior:

$$\hat{k} = \underset{k}{\operatorname{argmin}} \left\{ \frac{\sum_{j=1}^{360} w_j \|\tilde{\phi}_{\theta_j} - R_{\theta_j}k\|^2}{2\eta_p^2} + \lambda_1\|k\|^{\gamma_1} + \lambda_2\|\nabla k\|^{\gamma_2} \right\} \tag{2.12}$$

We use an iterative reweighted least squares method [48, 78] to minimize the energy in Eq. 2.12.

Aligning blur kernel projections In order to reconstruct an accurate blur kernel, it is important that blur kernel projections are aligned: the center of projection among all kernel projections should be the same. If the center of projection are not aligned, details of the blur kernel could be smeared out.

To align blur kernel projections, we exploit the fact that the center of gravity in an object's

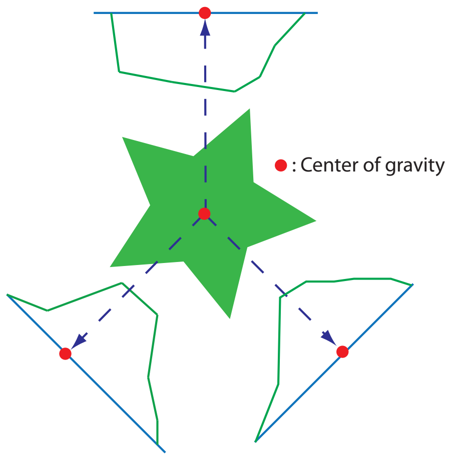


Figure 2.5: *The center of gravity within an object's projection is equivalent to the projection of the object's center of gravity.*

projection is equivalent to the projection of the object's center of gravity, as shown in Fig. 2.5. We shift the edge slices such that the center of gravity in each projection is at the center of the projection. This ensures that the center of projection among all kernel projections is aligned.

Synthetic experiments We analyze the performance of our kernel estimation algorithm using a synthetically blurred test pattern. We generated a test pattern with ideal lines and ideal step edges in 12 orientations, shown in Fig. 2.6. We blur this test pattern using a blur kernel shown at the top of Fig. 2.6, and add 0.5% Gaussian noise to the blurred pattern.

As a first experiment, shown in Fig. 2.6(a-c), we take 120 slices of blurred lines (at edge samples indicated with green dots) and recover a blur kernel from those slices using three different inverse Radon transform algorithms. We consider a back projection algorithm in Fig. 2.6(a), a filtered back projection algorithm in Fig. 2.6(b), and our algorithm in Fig. 2.6(c). We add different amount of Gaussian noise to the ground-truth orientation of each slice to stress-test algorithms to orientation estimation error. Recovered kernels under different orientation noise levels are shown in colored boxes. We observe that our algorithm faithfully reconstructs the blur kernel at all orientation noise level, whereas other algorithms reconstruct kernels that are too "blurred" or that have streaks even at a low orientation noise level. This shows that a sparse prior on blur kernels improves the kernel reconstruction performance.

As a second experiment, shown in Fig. 2.6(d), we take 120 slices of blurred *edges* and recover the blur kernel from the derivatives of blurred edge profiles. Again, we add different

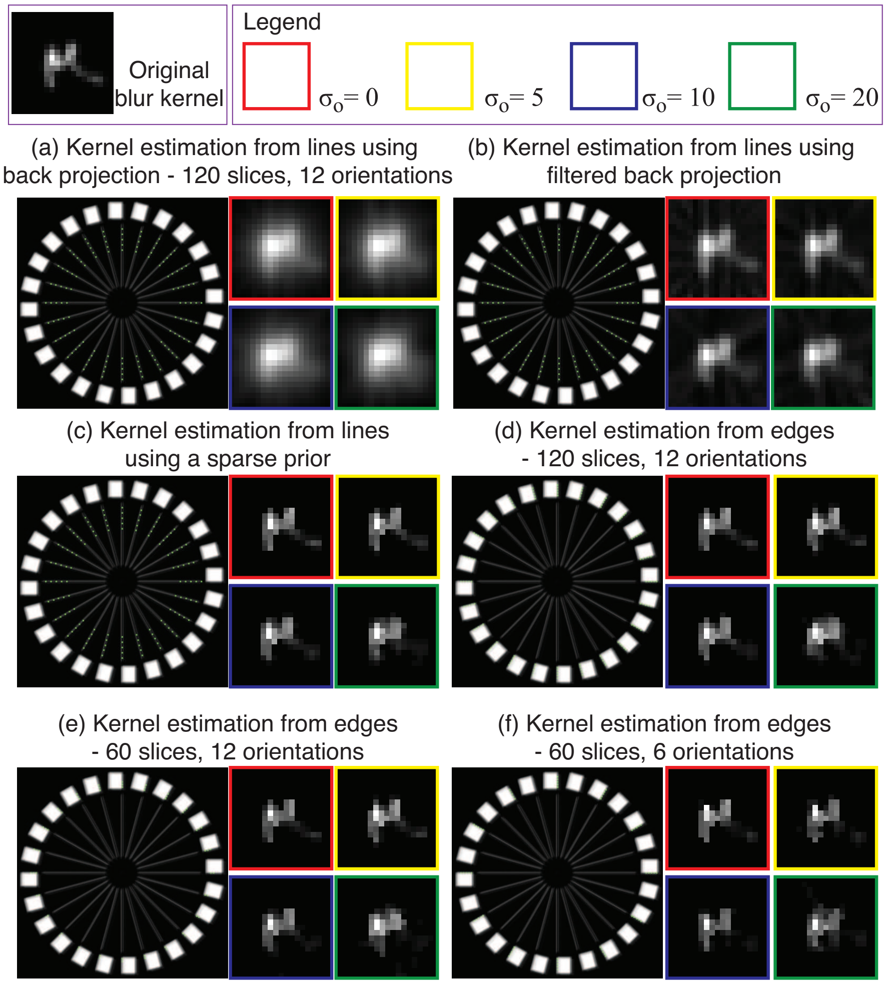

Original blur kernel
Legend
σ_o= 0
σ_o= 5
σ_o= 10
σ_o= 20
(a) Kernel estimation from lines using back projection - 120 slices, 12 orientations
(b) Kernel estimation from lines using filtered back projection
(c) Kernel estimation from lines using a sparse prior
(d) Kernel estimation from edges - 120 slices, 12 orientations
(e) Kernel estimation from edges - 60 slices, 12 orientations
(f) Kernel estimation from edges - 60 slices, 6 orientations

Figure 2.6: *We estimate a blur kernel from a synthetically blurred test pattern. We blur the test pattern using the blur kernel shown on the top left. As a first experiment, we compare three different inverse Radon transform algorithms: (a) the back projection algorithm (b) the filtered back projection algorithm (c) our algorithm in Eq. 2.12. We estimate the blur kernel from 120 slices of lines in 12 orientations. Green dots correspond to pixels at which we take the slices. We add different amount of orientation noise, of standard deviation σ_o (in terms of degrees), to the ground-truth orientation, and show reconstructed blur kernels in each case. We observe that our algorithm faithfully reconstructs the kernel across all orientation noise levels, whereas other algorithms reconstruct kernels that are too "blurred" or that have streaks. We test the stability of our kernel reconstruction algorithm by varying the number of edge slices and the number of orientations. (d) 120 slices of edges in 12 orientations (e) 60 slices of edges in 12 orientations (f) 60 slices of edges in 6 orientations. We observe that it is important to sample enough edges in many orientations.*

amount of Gaussian noise to the ground-truth orientation. The kernel estimation performance deteriorates slightly since differentiation of edge profiles doubles the observation noise variance. However, recovered kernels are close to the ground-truth kernel across all orientation noise levels. In Fig. 2.6(e), we reduce the *number* of edge slices for kernel estimation while sampling edges in all 12 orientations. Reducing the number of slices by a factor of 2 increases the noise variance by a factor of $\sqrt{2}$, but even in this case the estimated kernels are still quite accurate. When we reduce the number of orientations by a factor of two Fig. 2.6(f) while using 60 slices as in Fig. 2.6(e), however, our algorithm is less stable. This experiment shows that, if possible, we should take many edge samples in many orientations.

■ 2.2.3 Detecting reliable edges from a blurry image

For an accurate kernel reconstruction, we need to find stable, isolated step edges. We introduce an image analysis technique that selects stable edges from a blurry image. As a first step, we run an edge detector to find an edge map E of candidate edge samples.

Our goal is to sieve isolated step edges that satisfy four desired characteristics. First, selected pixels should correspond to a step edge with enough contrast on either side, which ensures that the signal to noise ratio of the blurred profile is high. We enforce this constraint by discarding edge samples with a small color difference between two locally dominant colors (Sec. 2.2.1). In RGB space, if $\|W - Z\| < 0.03$, we discard that edge sample. Second, the blurred edge profile should not be contaminated by adjacent edges. To ensure that two adjacent step edges are sufficiently separated, we take an orthogonal slice s_E of the edge map E at each edge candidate, and we discard edge samples with $\sum s_E > 1$. Third, a local neighborhood of an edge candidate should conform to a color-line image model. In other words, blurred edge profiles (i.e. α's from Eq. 2.7) should lie between 0 and 1. An edge sample with a slice that lies outside of $0 - \varepsilon$ and $1 + \varepsilon$, where $\varepsilon = 0.03$, is discarded. Lastly, the edge should be locally straight. The "straightness" is measured as the norm of the average orientation phasor in the complex domain. At each edge candidate l, we compute the following measure:

$$\frac{\| \sum_{j \in N(l)} \exp(-i2\theta_j) \|}{\sum_{j \in N(l)} 1} \tag{2.13}$$

where $i = \sqrt{-1}$, and $N(l)$ indicates edge candidates in the neighborhood of pixel l. If this norm is close to 1, then the edge is locally straight in the neighborhood of pixel l. We discard edge samples with the norm less than 0.97.

Our edge selection algorithm depends on the blur kernel size, which is estimated by users. If the estimated blur kernel size is too large, the second and third step of our edge selection algorithm would reject many edges since (i) more slices of the edge map E would contain more than one edge (ii) the size of the neighborhood in which the color-line model should hold increases. Therefore, users should ensure that the estimated blur size is just enough to contain the blur.

■ 2.3 Experimental results

This section provides experimental results that illustrate the performance of our deblurring algorithm. We compare our algorithm's performance to three competing methods: Fergus *et al.* [31], Shan *et al.* [74], and Cho *et al.* [14]. In order to compare just the kernel estimation performance, we used the same deconvolution algorithm [48] to restore images.

Fig. 2.7 shows deblurred images. In most test images, our algorithm performs favorably compared to prior art. As long as we can find enough stable edges in many orientations, our algorithm can reliably estimate the kernel. Fig. 2.8 shows more comparisons.

Our algorithm sometimes recover blur kernels with spurious "islands", as in Fig. 2.8(a), when the edge selection algorithm erroneously includes unstable edges at which edge slices intersect other neighboring edges. A better edge selection algorithm should reduce such error.

Another limitation of this algorithm is that it can be unstable when there are not enough edges, as shown in Fig. 2.9(a), and/or when there are not enough edges in different orientations, as shown in Fig. 2.9(b). When there are not enough edges, there simply isn't much information to estimate the kernel with; when there are only few dominant orientations in selected edges, we can only constrain the blur in those orientations and cannot recover meaningful blur kernel in other orientations. In some cases, this is less problematic. An interesting aspect of estimating the blur kernel explicitly from blurred edge profiles is that the estimated blur kernel contains enough information to properly deblur edges in those orientations, even if the blur kernel is not entirely correct. For instance, if an image is a single step edge, as in Fig. 2.10, we do not need to recover the original blur kernel to adequately remove the blur. We can remove the blur from the stripes as long as we recover the horizontal component of the blur kernel, and this is what our algorithm does.

Kernel projection constraints in Eq. 2.9 assume that the image B is a "linear" image. In other words, the blurred image B is not processed by any non-linear operators such as non-

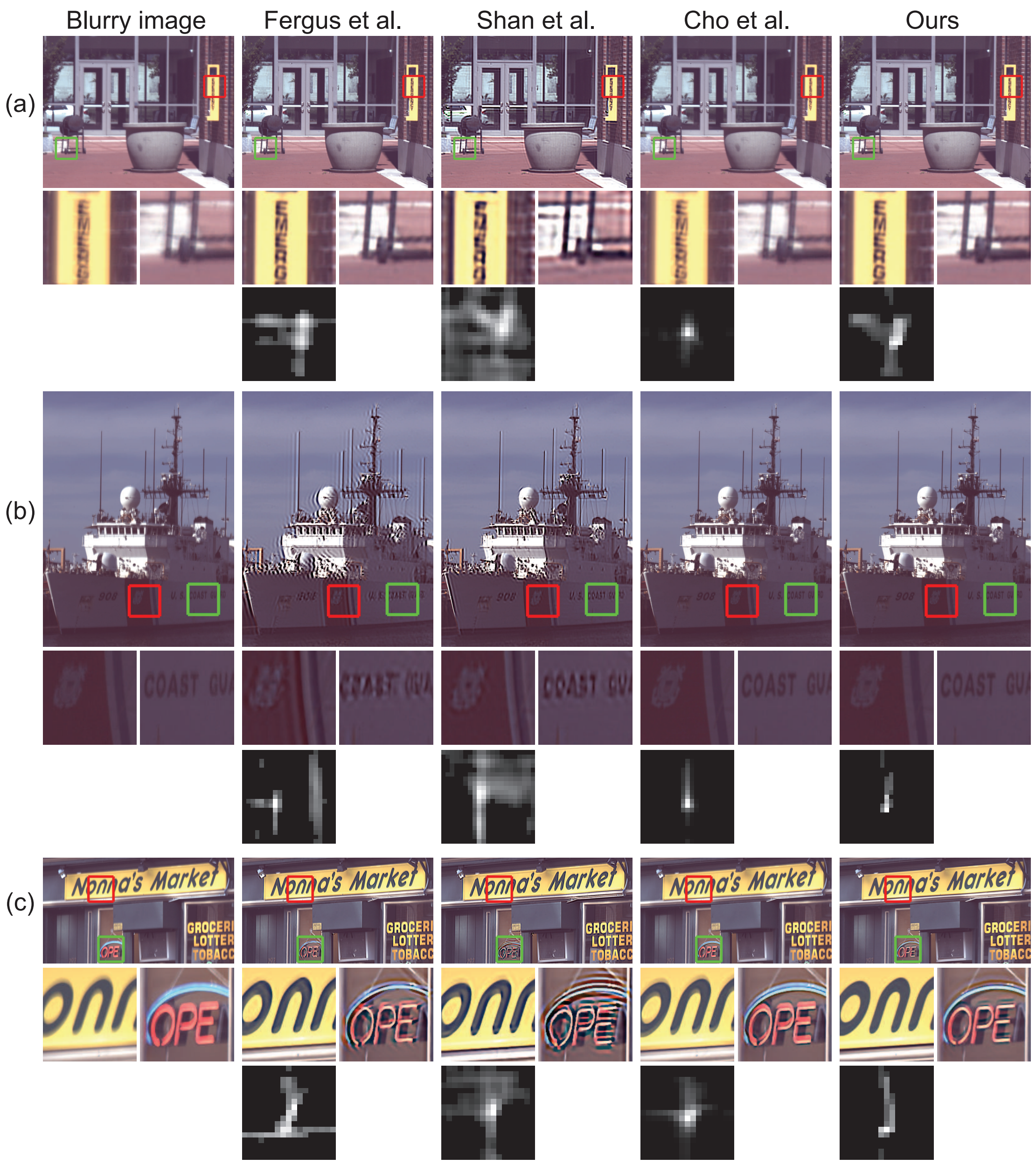


Figure 2.7: *This figure compares our algorithm's kernel estimation performance to three previous work: Fergus et al. [31], Shan et al. [74], and Cho et al. [14]. In most examples, our algorithm compares favorably to prior art.*

linear tone maps. We observe experimentally that our algorithm is vulnerable to non-linearities in B, therefore it is important to properly linearize the input image B. in this work, we used

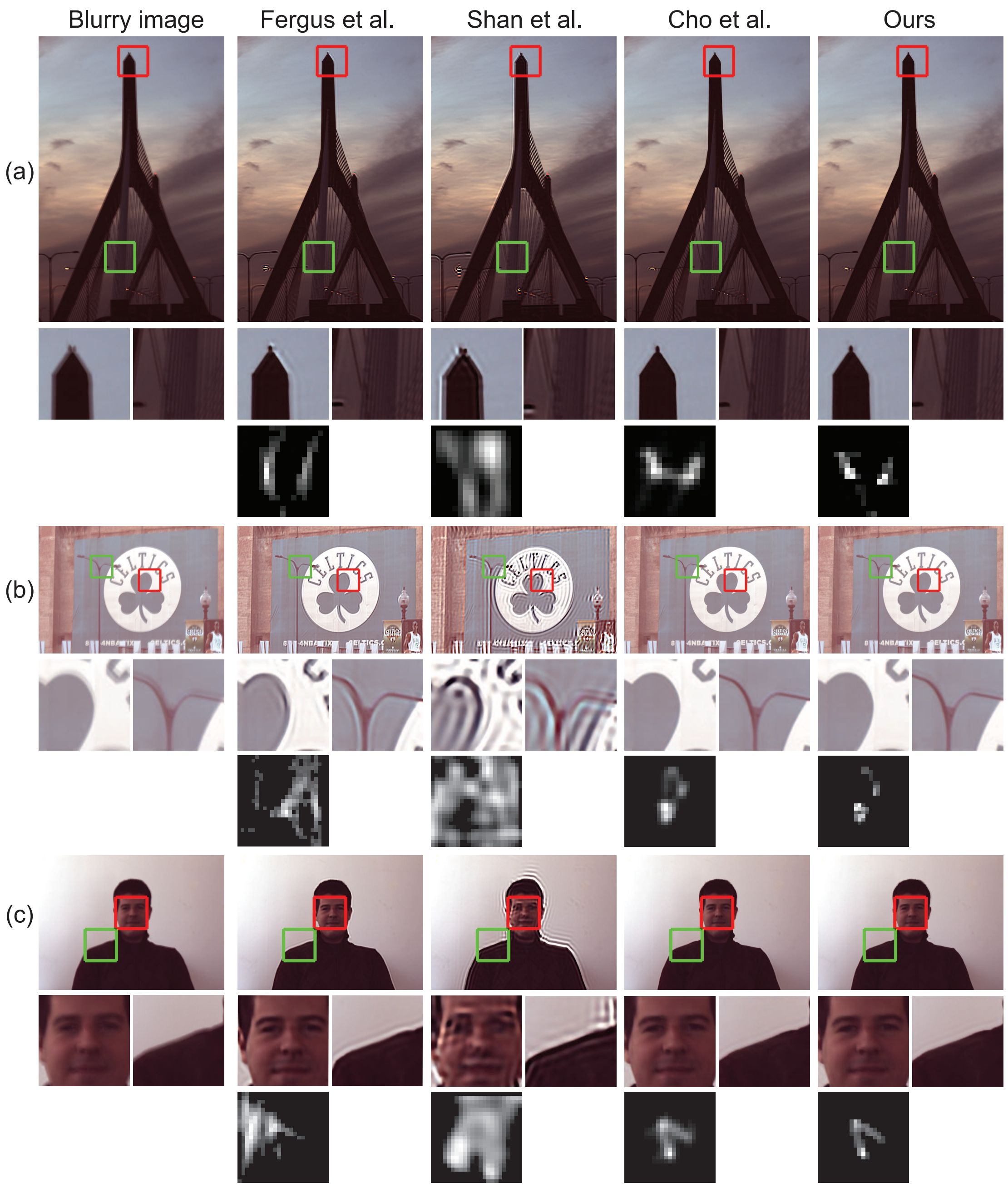


Figure 2.8: *This figure shows more comparisons of our kernel estimation algorithm and prior art.*

only raw images as our test set in order to factor out artifacts from non-linearities. We observe that while competing algorithms are less susceptible to non-linearities, using raw images also

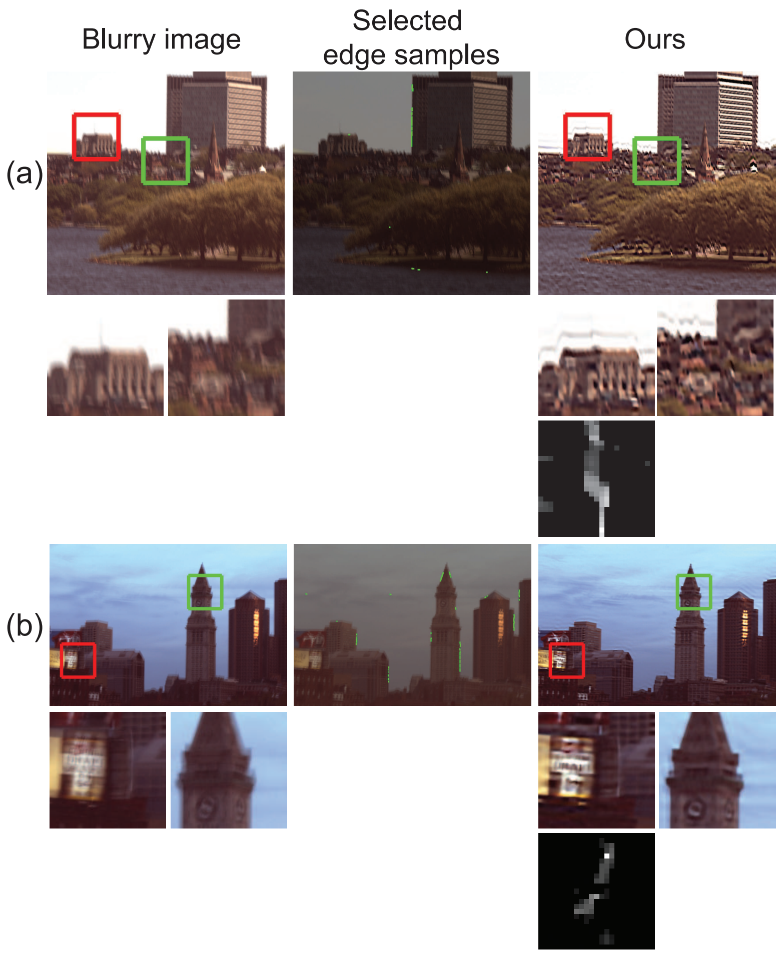


Figure 2.9: *Our kernel estimation algorithm is sensitive (a) when there are not enough edges and/or (b) when there are not enough edges in different orientations. This figure illustrates these failure modes.*

improves their performance.

Chromatic aberration from a camera lens may also affect kernel estimation performance. When chromatic aberration is significant, our edge selection algorithm will discard most edge samples because an edge slice would not be explain by two dominant colors (Sec. 2.2.1).

■ 2.4 The joint estimation of the blur kernel and the sharp image

As discussed in the previous section, our kernel estimation algorithm is less stable when there are not enough edges in many orientations. To handle images that do not have enough isolated

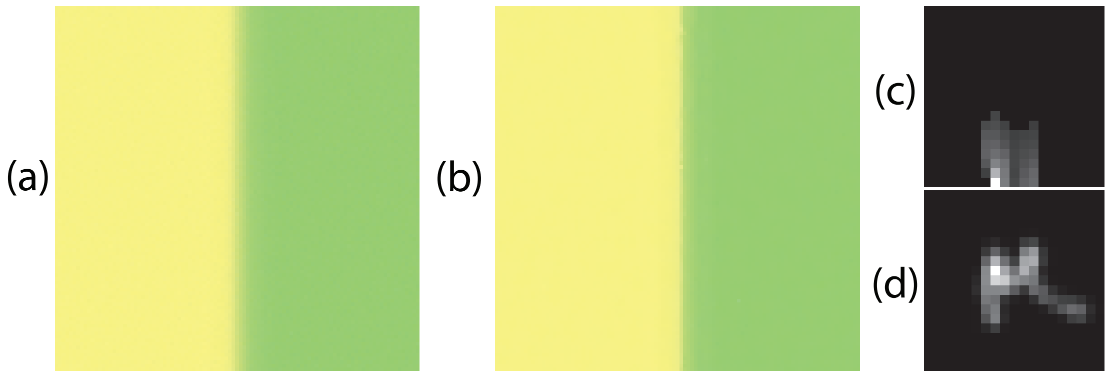


Figure 2.10: *(a) A blurred stripe (b) The deblurred stripe using the kernel estimated from our algorithm (c) Estimated blur kernel (d) The ground-truth blur kernel. Our kernel estimation algorithm only recovers the "horizontal" component of the ground-truth blur kernel, but the deblurred image is still crisp and is free of ringing.*

edges, we develop a method to incorporate kernel projection constraints in a more general deblurring framework.

One method to estimate a blur kernel k and a sharp image I is by maximizing the joint distribution of k and I [14, 74]:

$$\begin{aligned}[\hat{k},\hat{I}] &= \underset{k,I}{\operatorname{argmax}}\, p(k,I|B) \\ &= \underset{k,I}{\operatorname{argmax}}\, p(B|k,I)p(k)p(I) \end{aligned} \tag{2.14}$$

$[\hat{k},\hat{I}]$ is called a maximum-a-posteriori (MAP) of the joint distribution $p(k,I|B)$.

One often models the likelihood term $p(B|k,I)$ using the image observation model (Eq. 2.1):

$$p(B|k,I) \propto \exp\left(-\frac{\|B-k\otimes I\|^2}{2\eta_n^2}\right) \tag{2.15}$$

The image prior $p(I)$ favors a piecewise-smooth latent image:

$$p(I) \propto \exp\left(-\lambda\|\nabla I\|^{\gamma}\right) \tag{2.16}$$

The blur kernel prior $p(k)$ favors blur kernels with sparse intensity profiles as well as sparse gradient profiles (Eq. 2.11). Because maximizing $p(k,I|B)$ with respect to k,I jointly is challenging, we can resort to an alternating maximization algorithm to solve Eq. 2.14: we first

```
% Initial kernel estimation
$\hat{k} \Leftarrow \text{argmin}_k$ Eq. 2.12
for l = 1 to 5 do
    $\hat{I} \Leftarrow \text{argmax}_I\, p(\hat{k}, I|B)$ % Latent image estimation
    $\hat{\mathbf{I}} \Leftarrow \text{bilateralFiltering}(\hat{I})$
    $\hat{k} \Leftarrow \text{argmax}_k\, p(k, \hat{\mathbf{I}}|B)$ % Kernel estimation
end for
$\hat{I} \Leftarrow \text{argmax}_I\, p(\hat{k}, I|B)$
```

Algorithm 1: `MAP kernel estimation`

maximize the joint distribution $p(k,I|B)$ with respect to the blur kernel k while keeping the image I fixed, and then we maximize $p(k,I|B)$ with respect to I while holding k fixed. We iterate these two steps until convergence.

Despite the simplicity, Levin *et al.* [50] argue that the joint estimation of the kernel and the sharp image is not a good idea because the joint probability Eq. 2.14 is often maximized when k is an impulse function (i.e. no-blur) and I is the input blurry image B. For instance, the no-blur solution pair maximizes the likelihood (Eq. 2.15), an impulse function is not penalized by the blur kernel prior, and a blurry version of an image is sometimes favored by the image prior over its original version [50].

To resolve this issue, we augment the likelihood term in Eq. 2.15 using the Radon transform constraint in Eq. 2.10:

$$p(B|k,I) \propto \exp\left(-\left\{\frac{\|B - k \otimes I\|^2}{2\eta_n^2} + \frac{\sum_{j=1}^{360} w_j \|\tilde{\phi}_{\theta_j} - R_{\theta_j} k\|^2}{2\eta_p^2}\right\}\right) \tag{2.17}$$

The Radon transform term penalizes the no-blur solution, and this steers the joint distribution $p(k,I|B)$ to favor the correct solution. Algorithm 1 shows the pseudocode for the joint estimation algorithm. Notice that we filter the latent image estimate $\hat{I}$ using a bilateral filter before re-estimating the kernel, as in [14]. $\hat{I}$ usually contains ringing and noise that violate the sparse gradient profile model, therefore $\hat{I}$ is not a good estimate to refine the blur kernel with. The bilateral filter removes such ringing and noise such that the filtered image $\hat{\mathbf{I}}$ is closer to the image we want to recover. The bilateral filter step is important for improving the kernel estimation performance.

Experimental results Fig. 2.11(a-b) show how the MAP kernel estimation algorithm improves the failure cases shown in Fig. 2.9. The images deblurred using the MAP kernel estimation al-

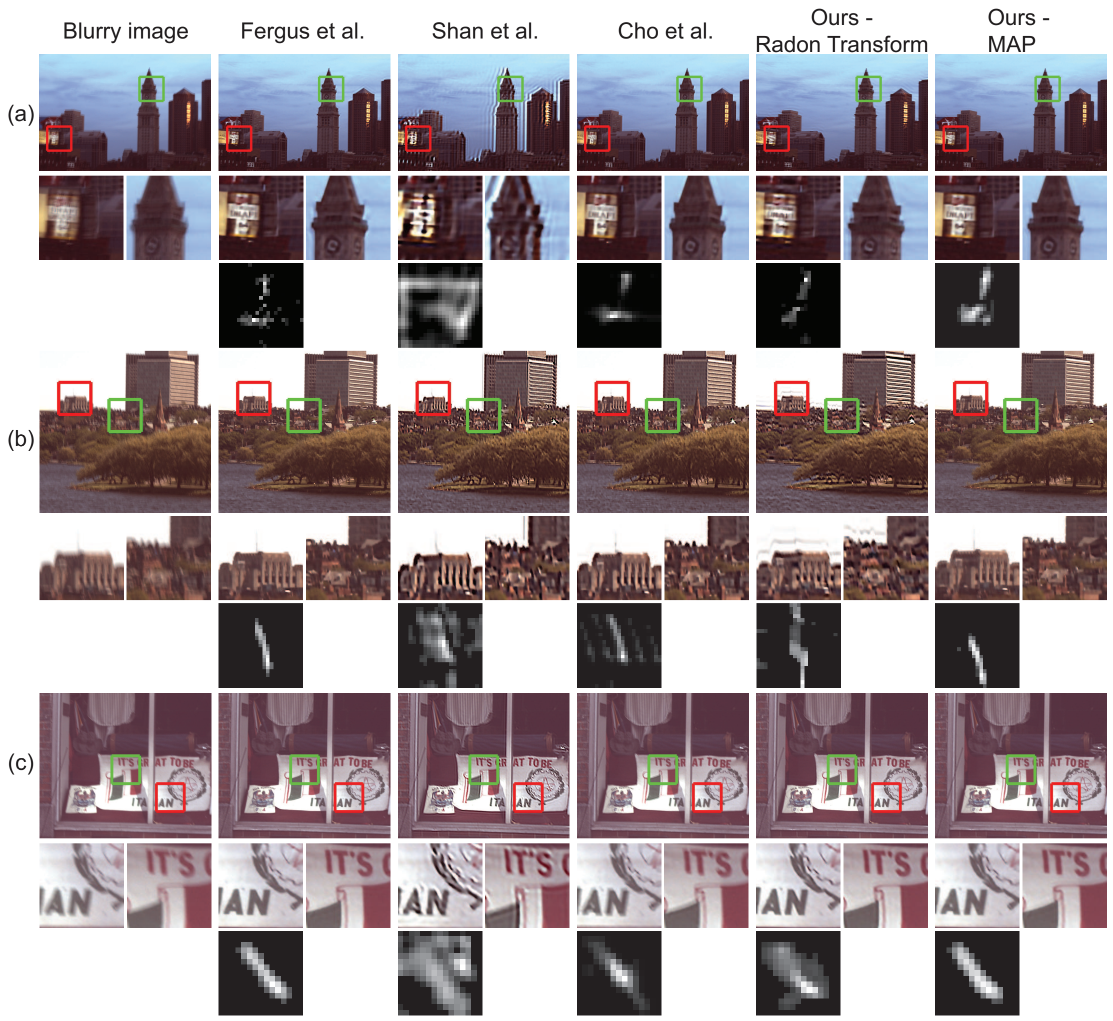


Figure 2.11: *By integrating kernel projection constraints to a MAP kernel estimation method, we can improve the kernel estimation performance. (a-b) show that even when there are not enough edges in different orientations (as shown in Fig. 2.9), the MAP kernel estimation algorithm can reliably reconstruct the kernel.*

gorithm are more crisp and have less ringing compared to those of our original kernel estimation algorithm. In general, the MAP kernel estimation algorithm cleans up spurious “islands” in estimated kernels and improves the quality of deblurred images. Fig. 2.12 shows more examples comparing the performance of competing deblurring algorithms.

To double-check that the new posterior probability models the problem better than the conventional posterior probability, we compare the negative log-posterior of our MAP solution and

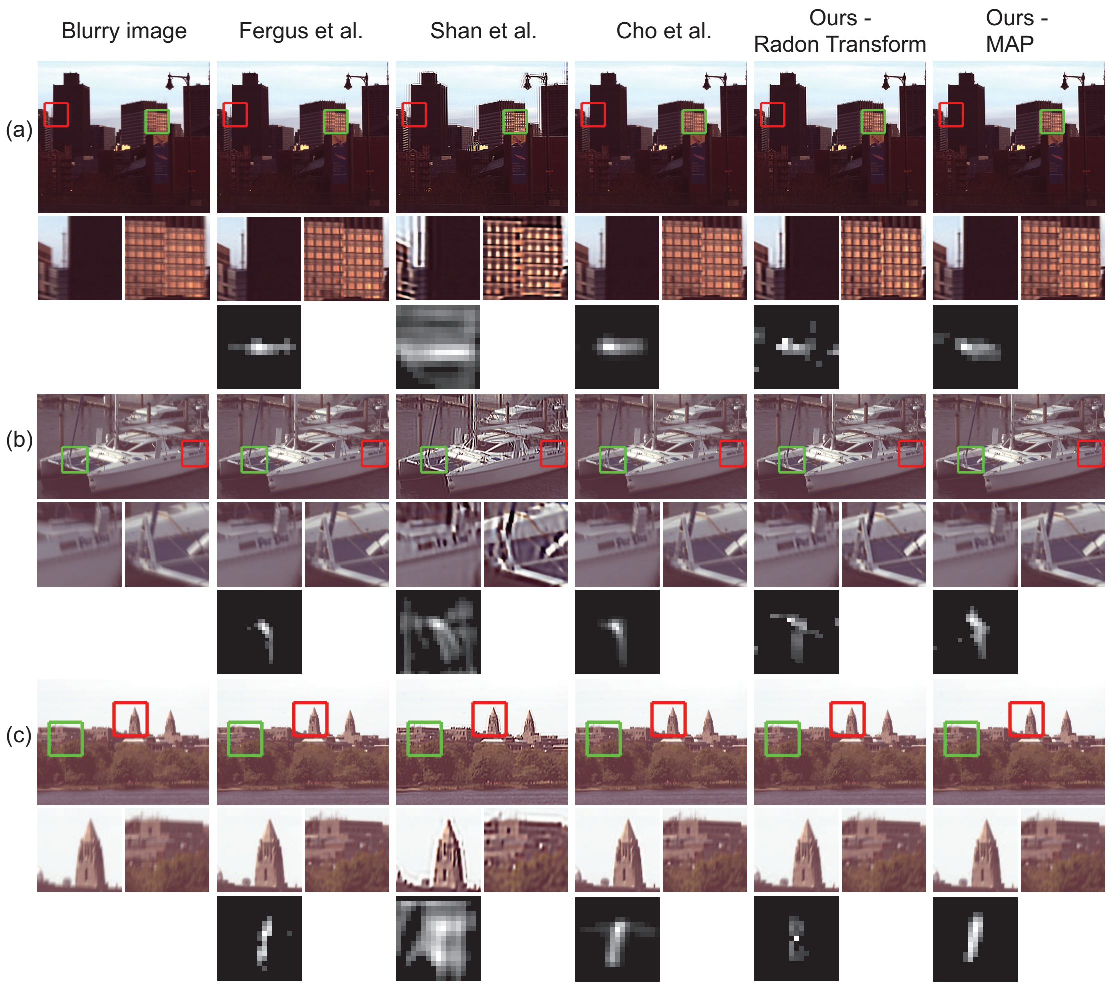


Figure 2.12: *We show more examples in which the MAP kernel estimation algorithm improves the estimated blur kernels.*

the no-blur solution. The negative log-posterior of our solution in Fig. 2.11(a) is 2.29×10^4, whereas that of the no-blur solution is 7.95×10^5: the Radon transform constraint effectively penalizes the no-blur solution.

■ 2.5 Conclusion

In this work, we introduce a new insight to the blur kernel estimation problem: blur kernels can be estimated by analyzing blurred edge profiles. Our technique is especially well suited to images that have many step edges in different orientations, such as man-made scenes. Our insight can also be useful for existing blur estimation methods. We presented a method to integrate kernel projection constraints in a MAP based kernel estimation framework. Experimental results show that our kernel estimation algorithm compares favorably to prior art.

Chapter 3

A content-aware image prior for image restoration

3.1 Introduction

EVEN if we could perfectly estimate the blur kernel from a blurry photograph, restoring a clean, sharp image is still a challenging problem because blur attenuates information about the original image. Image enhancement algorithms often resort to image priors to hallucinate the lost information.

Natural images often consist of smooth regions with abrupt edges, and this characteristic leads to a heavy-tailed gradient profile. In recent years, a heavy-tailed gradient profile has been extensively exploited as an image prior: the gradient statistics are represented by fitting a flexible parametric distribution to gradient samples. Fitted parameters are often kept uniform for the entire image, effectively imposing the same image prior everywhere [31, 48, 71]. Unfortunately, different textures have different gradient statistics even within an image, therefore imposing a single image prior for the entire image is inappropriate (Fig. 3.1).

We introduce an algorithm that adapts the image prior to both low-level local structures as well as mid-level texture cues, thereby imposing an adapted prior for each texture. [1] Adapting the image prior to the image content improves the image restoration performance. In Sec. 3.3, we analyze a large database of natural images and present an empirical result that gradient statistics in certain textures are not sparse. This observation justifies our argument that we should adapt the image prior to underlying textures. We provide algorithmic details of the content-aware image prior in Sec. 3.4, and show image restoration results in Sec. 3.5.

[1]Strictly speaking, an estimate of image statistics made after examining the image is no longer a “prior” probability. But the fitted gradient distributions play the same role as an image prior in image reconstruction equations, and we keep that terminology.

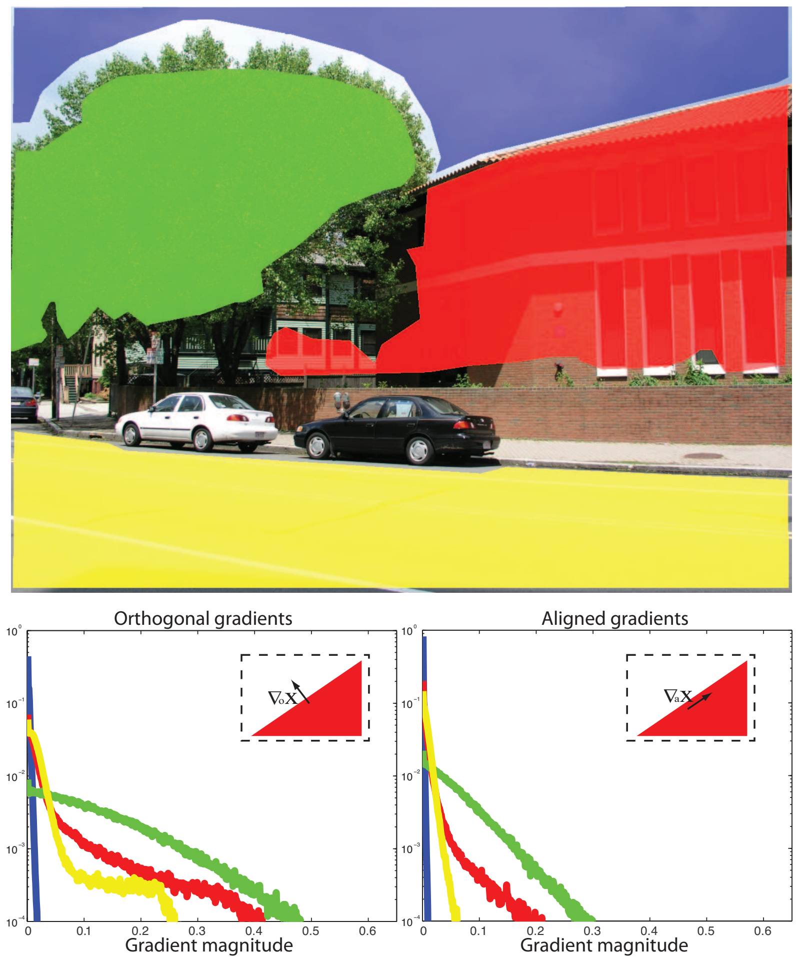


Figure 3.1: *Colored histograms represent gradient statistics of regions with the same color mask. In many images, the steered gradient profile is spatially variant. Therefore, an image prior should adapt to the image content. Insets illustrate how steered gradients adapt to local structures.*

■ 3.2 Related work

Image prior research revolves around finding a good image transform or basis functions under which a transformed natural image exhibits unique characteristics. Transforms derived from signal processing have been exploited in the past, including the Fourier transform [34], the Wavelet transform [83], the Curvelet transform [11], and the Contourlet transform [26].

Basis functions learned from natural images have also been introduced. Most techniques learn filters that lie in the null-space of the natural image manifold [66, 86, 87, 91]. Aharon *et al.* [4] learn a vocabulary that a natural image is composed of. However, none of these techniques adapt the basis functions to the image.

Edge-preserving smoothing operators do adapt to local structures. Anisotropic diffusion operators [8] detect edges, and smooth along edges but not across them. A similar idea appeared in a probabilistic framework called a Gaussian conditional random field [80]. A bilateral filter [82] is also closely related to anisotropic operators. Elad [27] and Barash [5] discuss relationships between edge-preserving operators.

Some image models adapt to edge orientation as well as magnitude. Hammond *et al.* [38] present a Gaussian scale mixture model that captures statistics of gradients that are adaptively steered in the dominant orientation within an image patch. Roth *et al.* [67] extends this idea to a random field to model oriented structures in images.

Adapting the image prior to textural characteristics was investigated for gray-scale images consisting of a single texture [71]. Bishop *et al.* [7] present a variational image restoration framework that breaks an image into square blocks and adapts the image prior to each block independently (i.e. the image prior is fixed within the block). However, Bishop *et al.* [7] do not address the stability issues at texture boundaries.

3.3 Image characteristics

We analyze statistics of gradients adaptively steered in the dominant orientation of local structures. Roth *et al.* [67] observe that the gradient profile of orthogonal gradients $\nabla_o I$ show a higher variance compared to that of aligned gradients $\nabla_a I$, thereby they propose imposing different priors on $\nabla_o I$ and $\nabla_a I$. We show that different textures within the same image also have distinct gradient profiles, therefore we propose adapting the prior to local textures.

We parameterize gradient profiles using a generalized Gaussian distribution:

$$p(\nabla I; \gamma, \lambda) = \frac{\gamma \lambda^{(\frac{1}{\gamma})}}{2\Gamma(\frac{1}{\gamma})} \exp(-\lambda \|\nabla I\|^{\gamma}) \tag{3.1}$$

where Γ is a Gamma function, and γ, λ are shape parameters. Qualitatively, γ determines the peakiness and λ determines the width of a distribution. We assume that $\nabla_o I$ and $\nabla_a I$ are independent: $p(\nabla_o I, \nabla_a I) = p(\nabla_o I) p(\nabla_a I)$.

■ 3.3.1 The distribution of γ, λ in natural images

Different textures give rise to different gradient profiles and thus different γ, λ. This section investigates the *distribution* of the shape parameters γ, λ in image patches. About $110,000$ image patches of 41×41 pixels are sampled from 500 high quality natural images, and their gradient profiles are fitted to a generalized Gaussian distribution to associate each patch with γ, λ. We fit the distribution by minimizing the Kullback-Leibler (KL) divergence between the empirical gradient distribution and the model distribution p. We can show that this is equivalent to minimizing the negative log-likelihood of the model distribution evaluated over gradient samples:

$$[\tilde{\gamma}, \tilde{\lambda}] = \underset{\gamma,\lambda}{\operatorname{argmin}} \left\{ -\frac{1}{N} \sum_{i=1}^{N} \ln \left(p(\nabla I_i; \gamma, \lambda) \right) \right\} \tag{3.2}$$

Claim 1. *Suppose $x_i, i = 1...N$ are samples from an unknown distribution, and we would like to fit a parametric distribution q to the samples x_i. Let $q_E(x) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$ be an empirical distribution of the samples x_i, and let p be a generalized Gaussian distribution parameterized by shape parameters λ, γ. We show that a distribution p that best parameterizes the empirical distribution q_E (in the KL divergence sense) minimizes the sum of negative log-likelihood over samples x_i:*

$$\min_{\lambda,\gamma} KL(q_E || p) = \min_{\lambda,\gamma} \left\{ -\frac{1}{N} \sum_{i=1}^{N} \ln(p(x_i; \gamma, \lambda)) \right\} \tag{3.3}$$

Proof. We can show that the KL divergence between p_E and q takes the following form:

$$\begin{aligned} KL(q_E || p) &= \int_x q_E(x) \ln \left(\frac{q_E(x)}{p(x)} \right) dx \\ &= \int_x \frac{1}{N} \{ \sum_{i=1}^{N} \delta(x - x_i) \} \ln \left(\frac{\frac{1}{N}\{\sum_{i=1}^{N} \delta(x - x_i)\}}{p(x)} \right) dx \\ &= \frac{1}{N} \sum_{i=1}^{N} \ln \left(\frac{\frac{1}{N}}{p(x_i)} \right) \end{aligned} \tag{3.4}$$

Therefore,

$$\min_{\lambda,\gamma} KL(q_E || p) = \min_{\lambda,\gamma} \left\{ -\frac{1}{N} \sum_{i=1}^{N} \ln(p(x_i; \gamma, \lambda)) \right\} \tag{3.5}$$

$\square$

We use a Nelder-Mead optimization method [45] to solve Eq. 3.2.

Figure 3.2 shows the Parzen-window fit to sampled $\tilde{\gamma}, \ln(\tilde{\lambda})$ for $\nabla_o I, \nabla_a I$. For orthogonal gradients $\nabla_o I$, there exists a large cluster near $\gamma = 0.5, \ln(\lambda) = 2$. This cluster corresponds

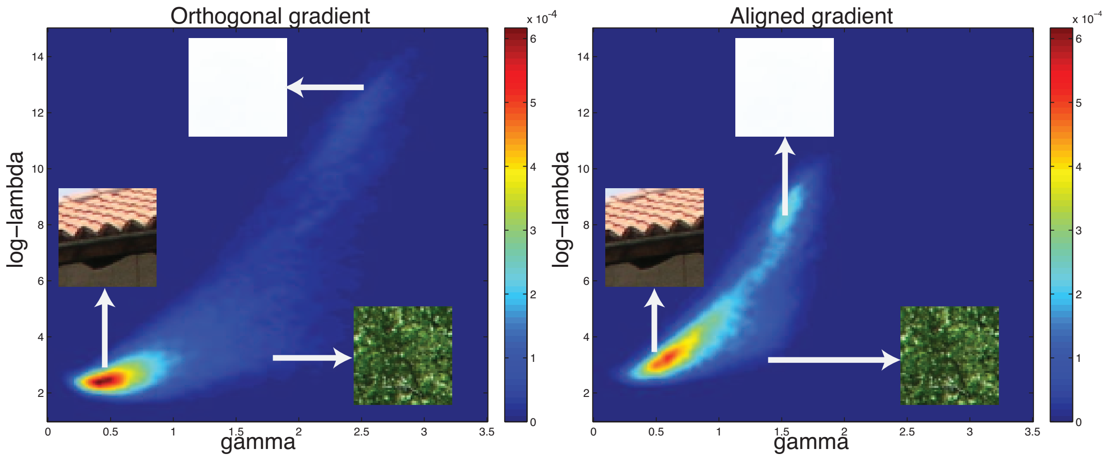


Figure 3.2: *The distribution of $\gamma, \ln(\lambda)$ for $\nabla_o I, \nabla_a I$ in natural images. While the density is the highest around $\gamma = 0.5$, the density tails off slowly with significant density around $\gamma = 2$. We show, as insets, some patches from Fig. 3.1 that are representative of different $\gamma, \ln(\lambda)$.*

to smooth patches with abrupt edges or patches from texture boundaries. This observation supports the fallen - leaves image model – an image is a collage of overlapping instances [46, 55]. However, we also observe a significant density even when γ is greater than 1. Samples near $\gamma = 2$ with large λ correspond to flat regions such as sky, and samples near $\gamma = 2$ with small λ correspond to fractal-like textures such as tree leaves or grass. We observe similar characteristics for aligned gradients $\nabla_a I$ as well. The distribution of shape parameters suggests that a significant portion of natural images is not piecewise smooth, which justifies adapting the image prior to the image content.

3.3.2 Spatially variant gradient statistics

Local gradient statistics can be different from global gradient statistics. Fig. 3.1 shows the gradient statistics of colored regions. Two phenomena are responsible for spatially variant gradient statistics: the material and the viewing distance. For example, a building is noticeably more piecewise smooth than a gravel path due to material properties, whereas the same gravel path can exhibit different gradient statistics depending on the viewing distance.

To understand the implication of spatially variant gradient statistics, we consider using the gradient profile of a specific region as an image prior for the entire image in restoring a blurry photo of Fig. 3.1. Fig. 3.3 illustrates the experiment. When we use gradient statistics from the sky to deblur the entire image, we recover the smooth sky, but over-smooth other regions.

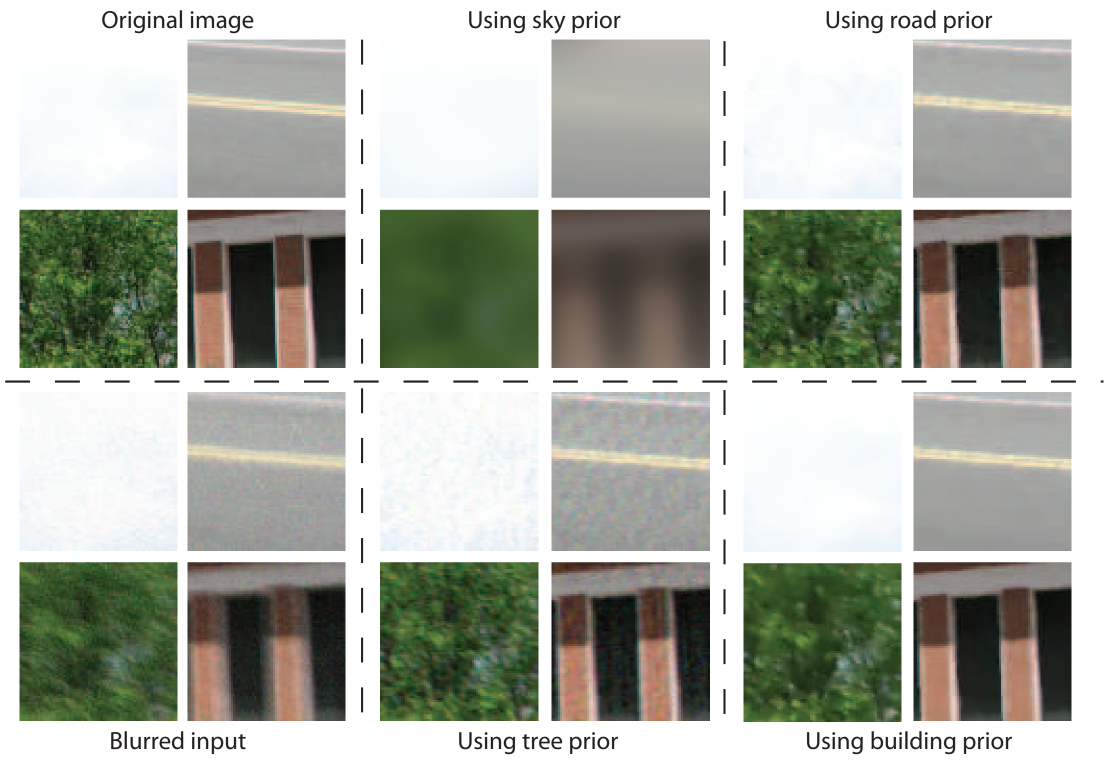


Figure 3.3: *We render a blurry photo of the sharp image in Fig. 3.1, and deblur it using priors fitted to gradient statistics of different regions. We show cropped patches of deblurred images for comparison. The prior from the sky over-smoothes all other regions to generate smooth sky; the prior from grass hallucinates high frequencies in grass at the expense of noise in other regions. While the prior from buildings generates reasonable renditions in all regions, the reconstructed image is generally piecewise smooth, which is disturbing in textured area such as trees.*

When we use the statistics from trees, we hallucinate high frequency texture in trees, but we fail remove noise from other regions. While the prior from the building does recover reasonable renditions in all regions, the reconstructed image is piecewise smooth and has visual artifacts in textured area such as trees. This experiment implies that by locally adapting the image prior to underlying textures, we can restore an image that is visually pleasing everywhere.

■ 3.3.3 Scale dependence of gradient histograms

The scale invariance of natural images has been studied extensively [68]. Two models embrace the scale invariance: an occlusion model (a.k.a. a fallen-leaves model [46, 55]) and a pink-noise

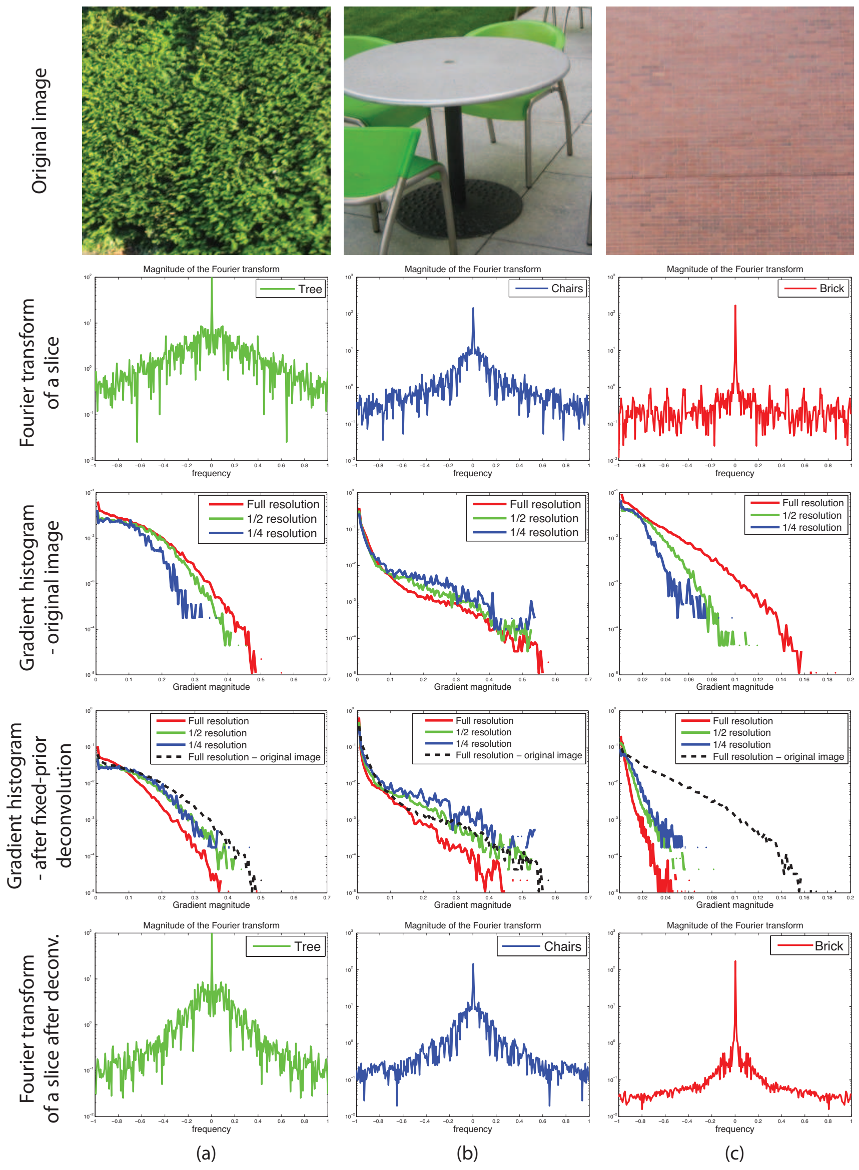


Figure 3.4: *Different textures exhibit different gradient statistics. For some textures the shape of the gradient histogram is roughly scale invariant. The gradient statistics after deconvolution also remain quite similar after down-sampling. First row: texture images. Second row: the Fourier spectrum of a 1D slice through the center of the corresponding sharp image. Third row: the gradient histogram of the corresponding sharp image. Fourth row: the gradient histogram of the deconvolved image. The dotted black line corresponds to the gradient distribution of a sharp image at the full resolution. Fifth row: the Fourier spectrum of a 1D slice through the center of the corresponding deconvolved image.*

model, which stipulates that a Fourier spectrum of a natural image falls off as a function of $1/f$. Srivastava *et al.* [77] experimentally show that the scale invariance holds only for an ensemble of images and not necessarily within each image.

In this section, we analyze the scale invariance of gradient histograms within *each* image. We assume that there are three dominant types of textures in natural scenes: a random natural texture such as grass, a man-made scene, and a structured texture such as a brick wall (Fig. 3.4). We can view a random natural texture as an instance of a pink-noise, and we can consider an image of man-made objects as an instance of an occlusion image model. We analyze the gradient profile of these textures as we vary the scale.

We first analyze the Fourier spectrum of a central row of the image. The Fourier spectra (shown in the second row of Fig. 3.4) of the grass and the chair scene (first two columns) fall off inversely proportional to frequency f, whereas that of a brick wall image, which exhibit structured textures, show humps at high frequencies (in addition to the harmonics) that correspond to structural details. From the Fourier spectrum, we can conjecture that the gradient histogram of structured textures may not be scale invariant since such structural details in high frequencies will be lost through down-sampling.

The third row of Fig. 3.4 shows the gradient histogram of each image across scale. The red line corresponds to the histogram at full resolution; the green line at one-half the full resolution; the blue line at one-quarter the full resolution. We notice that the gradient profile is roughly scale invariant for a random texture and a man-made scene, but that of structured textures is not. As is conjectured earlier, we can attribute the scale variance of structured textures to the loss of high frequency details from down-sampling.

To observe how deconvolution modifies gradient statistics, we blur the images and deconvolve them using a fixed image prior ($\gamma_o = 0.8, \gamma_a = 0.6$). The blur is about 5 pixels in width. The fourth row of Fig. 3.4 shows the gradient histogram of images after deconvolution. The dotted black line corresponds to the gradient distribution of the sharp image at the full resolution. The sparse deconvolution tends to preserve the gradient distribution of random textures and man-made scenes, but greatly modifies the gradient profile of structured textures because blur low-pass filters high frequency details.

Another interesting observation is that the gradient of the deconvolved image at one-half the original resolution mimics that of the original image at full resolution in the case of random texture and man-made objects. We leverage this observation in estimating the gradient distribution from the blurry input image.

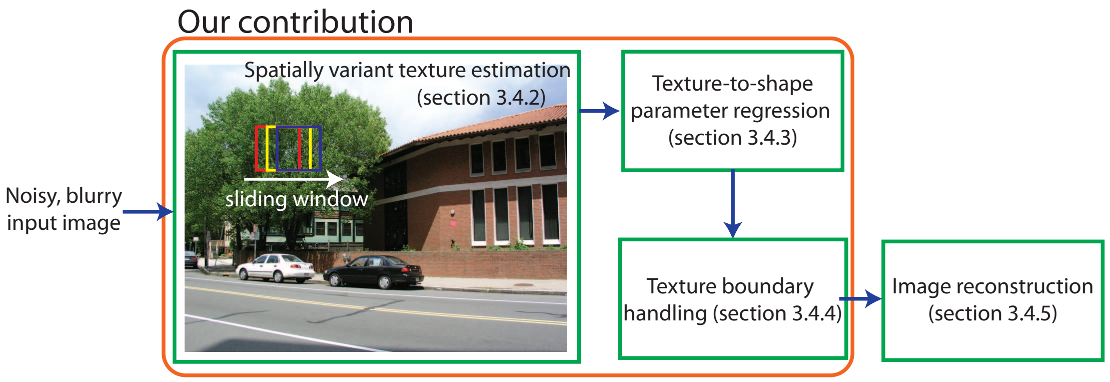

Our contribution
Spatially variant texture estimation (section 3.4.2)
sliding window
Noisy, blurry input image
Texture-to-shape parameter regression (section 3.4.3)
Texture boundary handling (section 3.4.4)
Image reconstruction (section 3.4.5)

Figure 3.5: *This figure illustrates the pipeline of our image restoration method. Detailed descriptions of each block are provided in Sec. 3.4.*

■ 3.4 The image prior estimation

The goal of this work is to identify the correct image prior for each pixel in the image. If we are given a sharp image, one way to identify the image prior at each pixel is to fit gradients in each sliding window to a generalized Gaussian distribution, and assign the fitted shape parameters to the central pixel of each window, as shown in Fig. 3.5. However, fitting gradients to a generalized Gaussian distribution requires a large amount of computation, rendering this operation for each sliding window intractable. Furthermore, we do not have a sharp image to estimate the image prior with.

This section presents our solution to this challenging problem: (i) we introduce a method to estimate the image prior directly from a blurry input image, and (ii) we present a regression-based technique to estimate the image prior. Our method is computationally more attractive compared to fitting gradients within each sliding window to a generalized Gaussian distribution.

■ 3.4.1 Image model

Let B be an observed degraded image, k be a blur kernel (a point-spread function or a PSF), and I be a latent image. Image degradation is modeled as a convolution process:

$$B = k \otimes I + n \tag{3.6}$$

where $\otimes$ is a convolution operator, and n is an observation noise. The goal of (non-blind) image restoration is to recover a clean image I from a degraded observation B given a blur kernel k and a standard deviation of noise η, both of which can be estimated through stand-alone techniques

[31, 53].

We introduce a conditional random field (CRF) model to incorporate texture variations within the image restoration framework. Typically, a CRF restoration model can be expressed as follows:

$$p(I|B,k,\eta) = \frac{1}{M}\prod_i \psi_B(I;B_i,k,\eta)\psi_I(I) \tag{3.7}$$

where M is a partition function and i is a pixel index. ψ_B is derived from the observation process; ψ_I from the assumed image prior:

$$\psi_B(I;B_i,k,\eta) \propto \exp\left(-\frac{(B_i-(k\otimes I)_i)^2}{2\eta^2}\right) \tag{3.8}$$

$$\psi_I(I) \propto \exp(-\lambda\|\nabla I\|^{\gamma}) \tag{3.9}$$

To model spatially variant gradient statistics, we introduce an additional hidden variable z, called *texture*, to the conventional CRF model. z controls the shape parameters of the image prior:

$$p(I,z|B,k,\eta) = \frac{1}{M}\prod_i \psi_B(I;B_i,k,\eta)\psi_{I,z}(I,z) \tag{3.10}$$

where $\psi_{I,z}(I,z) \propto \exp(-\lambda(z)\|\nabla I\|^{\gamma(z)})$. We model z as a continuous variable since the distribution of $[\gamma,\lambda]$ is heavy-tailed and does not form tight clusters (Fig. 3.2).

We maximize $p(I|B,k,\eta)$ to estimate a clean image $\hat{I}$. To do so, we approximate $p(I|B,k,\eta)$ by the function $p(I,z|B,k,\eta)$ at the mode $\hat{z}$:

$$p(I|B,k,\eta) = \int_z p(I,z|B,k,\eta)dz \approx p(I,\hat{z}|B,k,\eta) \tag{3.11}$$

Sec. 3.4.2 discusses how we estimate $\hat{z}$ for each pixel.

■ 3.4.2 Estimating the texture $\hat{z}$

A notable characteristic of a zero-mean generalized Gaussian distribution is that the variance v and the fourth moment f completely determine the shape parameters $[\gamma,\lambda]$ [75]:

$$v = \frac{\Gamma(3/\gamma)}{\lambda^{\frac{2}{\gamma}}\Gamma(1/\gamma)}, f = \frac{\Gamma(5/\gamma)}{\lambda^{\frac{4}{\gamma}}\Gamma(1/\gamma)} \tag{3.12}$$

To take advantage of these relationships, we define a variable called a local texture around each pixel i, $\hat{z}_i$, as a two dimensional vector. The first dimension is the variance v_i of gradients in the neighborhood of a pixel i, and the second dimension is the fourth moment f_i of gradients in the neighborhood of a pixel i:

$$\hat{z}_i = [v_i(\nabla I), f_i(\nabla I)] \tag{3.13}$$

Qualitatively, the variance of gradients $v_i(\nabla I)$ encodes the width of the distribution, and the fourth moment $f_i(\nabla I)$ encodes the peakiness of the distribution. Note that we can easily compute v_i, f_i through convolution.

Estimating the texture $\hat{z}$ from the observe image B We should ideally estimate the texture $\hat{z}$ from sharp image I, but I is not available when estimating $\hat{z}$. We address this issue by estimating the texture $\hat{z}$ from an image reconstructed using a spatially invariant image prior. We hand-select the spatially invariant prior with a weak gradient penalty so that textures are reasonably restored at the expense of slightly noisy smooth regions: $[\gamma_o = 0.8, \lambda_o = 6.5], [\gamma_a = 0.6, \lambda_a = 6.5]$. A caveat is that the fixed prior deconvolution may contaminate the gradient profile of the reconstructed image, which could induce texture estimation error. To reduce such deconvolution noise, we down-sample the deconvolved image by a factor of 2 before estimating the texture $\hat{z}$. As investigated in Sec. 3.3.3, a gradient profile of natural images is often scale invariant due to fractal properties of textures and piecewise smooth properties of surfaces [46, 55], whereas that of the deconvolution noise tends to be scale variant. Therefore, the texture $\hat{z}$ estimated from the down-sampled deconvolved image is close to the texture of the original sharp image.

■ 3.4.3 Estimating the shape parameters γ, λ from $\hat{z}$

We could numerically invert Eq. 3.12 to directly compute the shape parameters $[\gamma, \lambda]$ from the variance and fourth moment [75]. However, a numerical inversion is computationally expensive and is sensitive to noise. We instead use a kernel regression method that maps the log of the texture $\ln(\hat{z})$ to shape parameters $[\gamma, \ln(\lambda)]$.

We should train the regressor to learn how to map the texture $\hat{z}$ of the down-sampled *deconvolved* image to shape parameters to account for the effect of residual deconvolution noise in texture $\hat{z}$. Since the deconvolved image, thus $\hat{z}$, depends on the blur kernel and the noise level, we would ideally have to train regressors discriminatively for each degradation scenario, which is intractable. However, we empirically observe in Fig. 3.6 that the variance and fourth moment of the deconvolved, down-sampled image are close to those of the down-sampled original im-

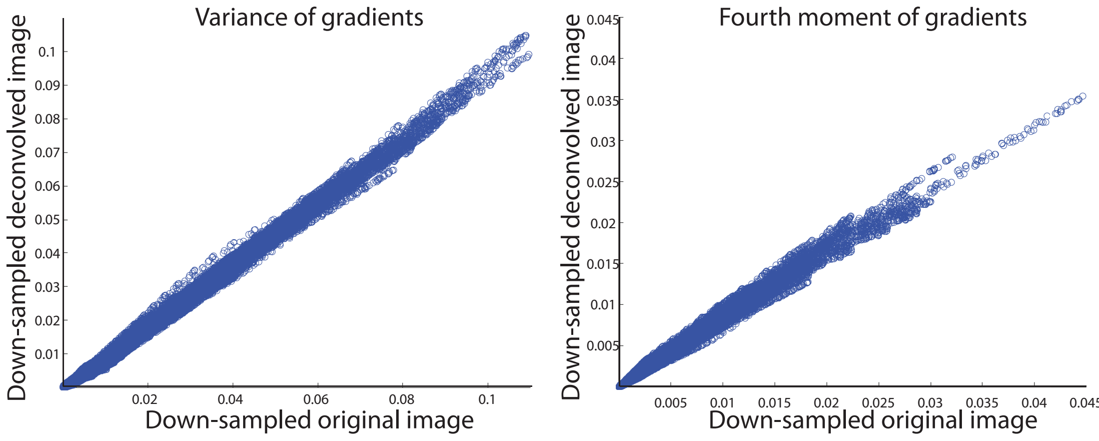


Figure 3.6: *We observe that the local variance and fourth moments of gradients computed from the deconvolved, down-sampled image of Fig. 3.1 are close to those computed from the down-sampled* original *image.*

age. Therefore we can afford to learn a *single* regressor from the variance and fourth moment of the sharp, down-sampled image to the shape parameters. The estimated shape parameters are reasonably accurate for our purpose.

To learn the regression function, we sample $\sim 125,000$ patches of size 17×17 pixels from 500 high quality natural images. We fit the gradient profile of each patch to a generalized Gaussian distribution, and associate each fit with the variance and fourth moment of gradients in the down-sampled version of each patch (9×9 pixels). We use the collected data to learn the mapping from $[\ln(v), \ln(f)]$ to $[\gamma, \ln(\lambda)]$ using LibSVM [12]. We use a 10-fold cross validation technique to avoid over-fitting.

■ 3.4.4 Handling texture boundaries

If multiple textures appear within a single window, the estimated shape prior can be inaccurate. Suppose we want to estimate the image prior for a 1-dimensional slice of an image (Fig. 3.7(a)). Ideally, we should recover two regions with distinct shape parameters that abut each other by a thin band of shape parameters corresponding to an edge. However, the estimated image prior becomes "sparse" (i.e. small γ) near the texture boundary even if pixels do not correspond to an edge (the green curve in Fig. 3.7(c)). This occurs because we use a finite-size window for computing v and f causes this issue.

To recover appropriate shape parameters near texture boundaries, we regularize the esti-

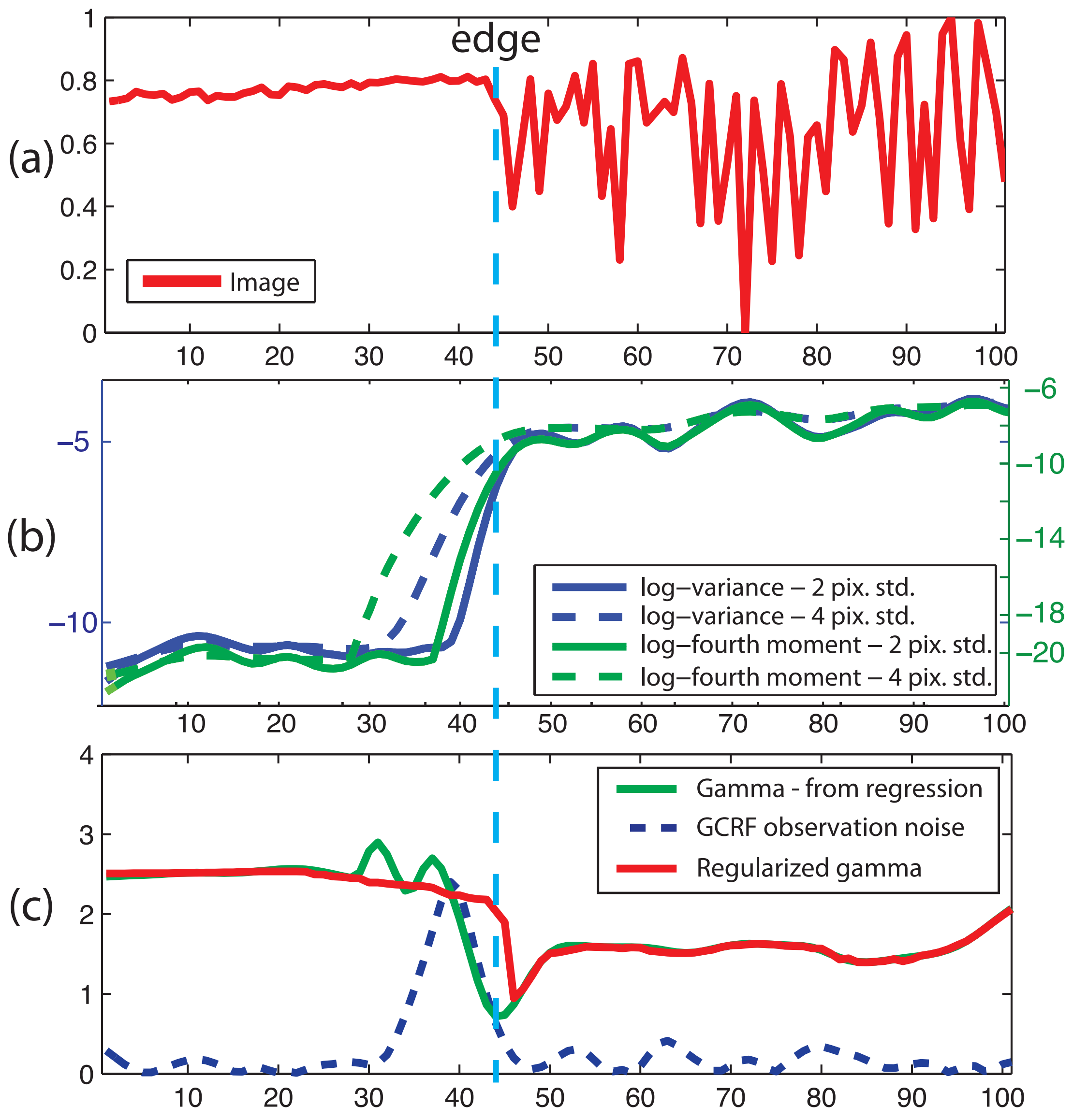


Figure 3.7: *We regularize the estimated shape parameters using a GCRF such that the texture transition mostly occurs at a texture boundary. We model the observation noise in the GCRF as the* variance *of the variance and fourth moments estimated from two Gaussian windows of different standard deviations – 2-pixel and 4-pixel, as shown in (b). This reduces the shape parameter estimation error at texture boundaries, as shown in (c) (compare green and red curves).*

mated shape parameters using a Gaussian conditional random field (GCRF) [80]. Conceptually, we want to smooth shape parameters only near texture boundaries. A notable observation at texture boundaries is that $\hat{z}$'s estimated from two different window sizes tend to be different from each other: the large window could span two different textures while the smaller window spans a homogenous texture, generating different $\hat{z}$'s. We exploit this observation to smooth only near texture boundaries.

To be more specific, we maximize the following probability to regularize $\tilde{\gamma}$ returned by the regressor:

$$p(\gamma;\tilde{\gamma}) \propto \prod_{i,j\in\mathbb{N}(i)} \psi(\tilde{\gamma}_i|\gamma_i)\Psi(\gamma_i,\gamma_j) \tag{3.14}$$

where $\mathbb{N}(i)$ denotes the neighborhood of i, ψ is the observation model and Ψ is the neighborhood potential:

$$\begin{aligned}\psi(\tilde{\gamma}_i|\gamma_i) &\propto \exp\left(-\frac{(\tilde{\gamma}_i-\gamma_i)^2}{2\sigma_l^2}\right)\\ \Psi(\gamma_i,\gamma_j) &\propto \exp\left(-\frac{(\gamma_i-\gamma_j)^2}{2\sigma_n^2(i,j)}\right)\end{aligned} \tag{3.15}$$

We set σ_l and σ_n adaptively. We set the variance $\sigma_n^2(i,j)$ of the neighboring γ as $\sigma_n^2(i,j) = \alpha(I(i)-I(j))^2$, where $\alpha = 0.01$ controls how smooth neighboring estimates should be. σ_n encourages the discontinuity at strong edges of the image I [80]. The observation noise σ_l^2 is the *mean variance* of the variance v and of the fourth moment f estimated from windows of two different sizes (Gaussian windows with 2-pixel and 4-pixel standard deviations.) If this value is large, the central pixel is likely to be near a texture boundary, thus we allow its estimated parameter to be smoothed. We use the same GCRF model to regularize $\ln(\lambda)$ with $\alpha = 0.001$.

Fig. 3.7(c) shows the estimated shape parameters before and after regularization along with the estimated GCRF observation noise. After regularization, two textures are separated by a small band of sparse image prior corresponding to an edge, as desired.

Fig. 3.8 shows the estimated shape parameters for orthogonal gradients of Fig. 3.1. In Fig. 3.8(a,b), the parameters are estimated from the image reconstructed from 5% noise and the blur in Fig. 3.11. In Fig. 3.8(a) we show the estimated shape parameters before texture boundary handling, and in Fig. 3.8(b) we show the result after texture boundary handling. Without texture boundary handling, estimated shape parameters show "ringing" at texture boundaries. After texture boundary handling, we correctly estimate the shape parameters even at texture boundaries. We observe that the estimated prior in the tree region is close to Gaussian (i.e.

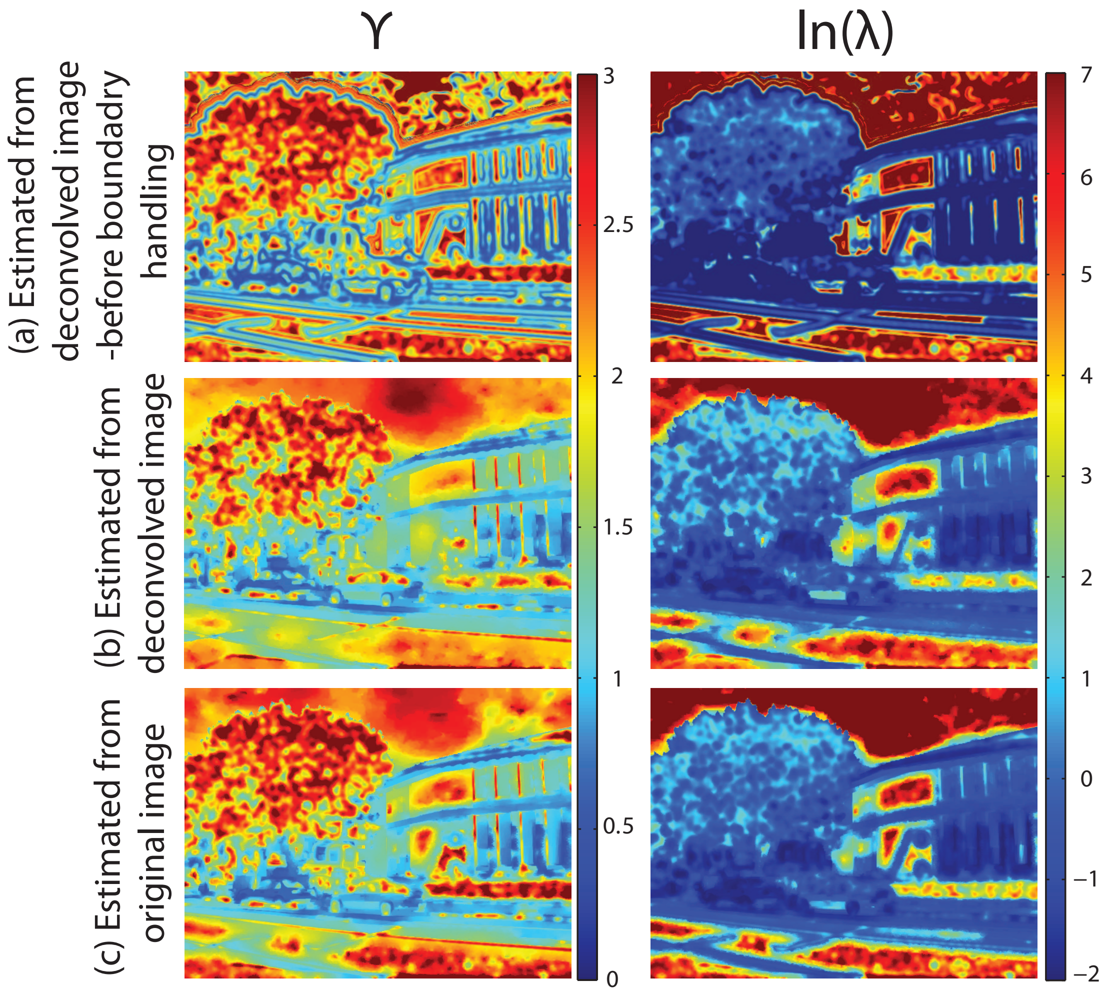


Figure 3.8: *(a) The shape parameters for orthogonal gradients, estimated from the down-sampled deconvolved image of Fig. 3.1, before texture boundary handling. We observe "ringing" at texture boundaries. (b) The estimated shape parameters after texture boundary handling. Parameters are more consistent at texture boundaries. (c) The shape parameters estimated from the down-sampled* original *image. (c) is quite close to (b), which implies that our kernel estimation method is accurate.*

$\gamma = 2 \sim 3$), whereas the estimated prior in the building region is sparse (i.e. $\gamma < 1$). The estimated shape parameters are similar to parameters estimated from the down-sampled, original image, shown in Fig. 3.8(c). This supports the claim that shape parameters estimated from a degraded input image are reasonably accurate.

3.4.5 Implementation details

We minimize the negative log-posterior to reconstruct a clean image $\hat{I}$:

$$\hat{I} = \underset{I}{\operatorname{argmin}}\{\frac{(B - k \otimes I)^2}{2\eta^2} + w \sum_{i=1}^{N} (\lambda_o(\hat{z}_i) \|\nabla_o I(i)\|^{\gamma_o(\hat{z}_i)} + \lambda_a(\hat{z}_i) \|\nabla I\|^{\gamma_a(\hat{z}_i)})\} \tag{3.16}$$

where $[\gamma_o, \lambda_o], [\gamma_a, \lambda_a]$ are estimated parameters for orthogonal and aligned gradients, respectively, and w is a weighting term that controls the gradient penalty. $w = 0.025$ in all examples. We minimize Eq. 3.16 using an iterative reweighted least squares algorithm [48, 78]. Algorithm 2 shows the pseudocode of the entire system.

```
$\gamma_o \Leftarrow 0.8, \lambda_o \Leftarrow 6.5, \gamma_a \Leftarrow 0.6, \lambda_a \Leftarrow 6.5$
$\tilde{I} \Leftarrow \arg\min_I$ Eq. 3.16
for Orthogonal and aligned gradients of $\tilde{I}$ do
    % For every pixel in the image
    for $i = 1$ to N do
        $v_i \Leftarrow \frac{1}{M} \sum_{l \in \mathbb{N}(i)}^{M} (\nabla \tilde{I})^2$
        $f_i \Leftarrow \frac{1}{M} \sum_{l \in \mathbb{N}(i)}^{M} (\nabla \tilde{I})^4$
        $z_i \Leftarrow [v_i, f_i]$
        $[\gamma_o(i), \ln(\lambda_o(i))] \Leftarrow regression_o(z_i)$
        $[\gamma_a(i), \ln(\lambda_a(i))] \Leftarrow regression_a(z_i)$
    end for
end for
$[\gamma_o, \lambda_o, \gamma_a, \lambda_a] \Leftarrow GCRFRegularize([\gamma_o, \lambda_o, \gamma_a, \lambda_a])$
$\hat{I} \Leftarrow \arg\min_I$ Eq. 3.16
```

Algorithm 2: *Image reconstruction algorithm*

3.5 Experimental results

We evaluate the performance of the content-aware image prior by applying the prior in image restoration tasks. The use of the content-aware image prior in image restoration tasks improves the rendition of textures.

3.5.1 Image restoration

We evaluate the performance of the content-aware image prior for deblurring and denoising tasks. We compare our results to those reconstructed using a sparse unsteered gradient prior

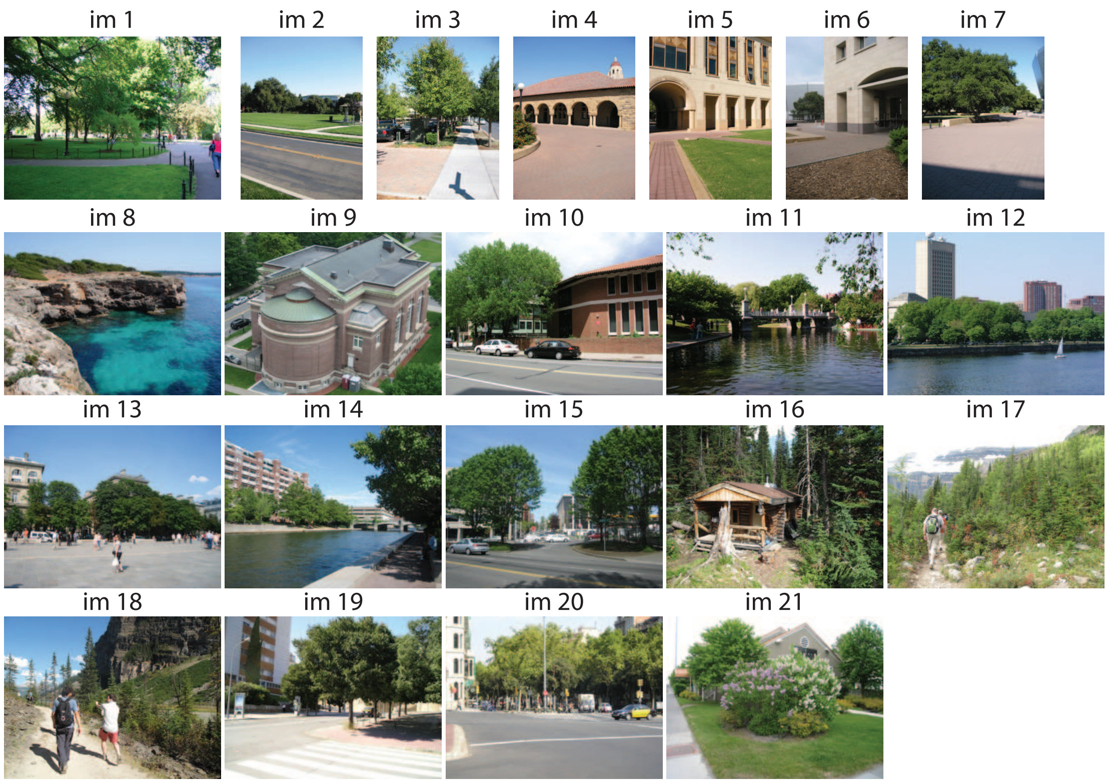


Figure 3.9: *We compare the image restoration performance on 21 natural images with spatially variant texture characteristics.*

[48] and a sparse steered gradient prior [67], using peak signal-to-noise ratio (PSNR) and gray-scale structural similarity (SSIM) [85] as quality metrics. We have augmented the steerable random fields [67], which introduced denoising and image in-painting as applications, to perform deconvolution. In all experiments, we use the first order and the second order gradient filters [32]. We can augment these algorithms with higher order gradient filters to improve reconstruction qualities, but it is not considered in this work. The test set, shown in Fig. 3.9, consists of 21 high quality images downloaded from LabelMe [69] with enough texture variations.

Non-blind deconvolution The goal of non-blind deconvolution is to reconstruct a sharp image from a blurred, noisy image given a blur kernel and a noise level. We generate our test set by blurring images with the kernel shown in Fig. 3.11, and adding 5% noise to blurred images. Fig. 3.10 shows the measured PSNR and SSIM for different deconvolution methods. The content-aware prior performs favorably compared to the competing methods, both in terms of

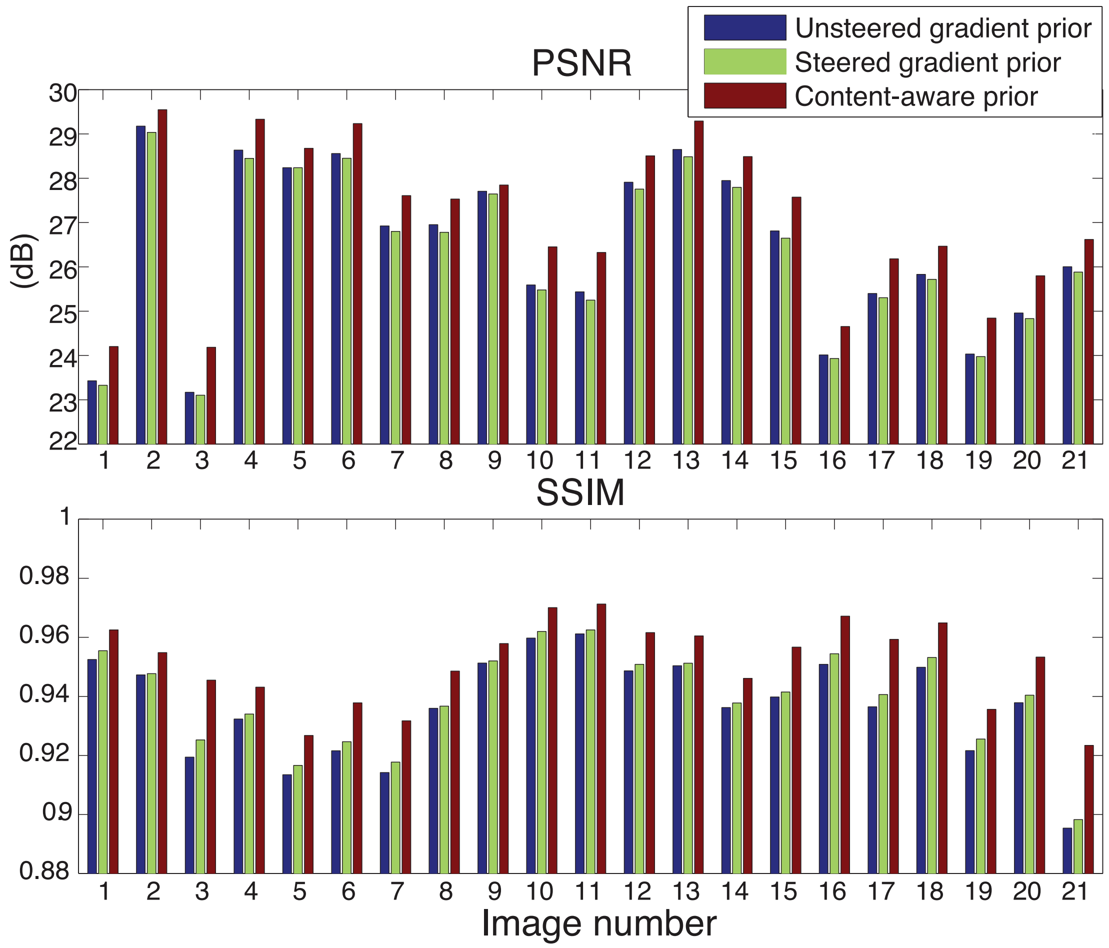


Figure 3.10: *Image deconvolution results : PSNR and SSIM. Mean PSNR: unsteered gradient prior – 26.45 dB, steered gradient prior – 26.33 dB,* **content-aware prior – 27.11 dB**. *Mean SSIM: unsteered gradient prior – 0.937, steered gradient prior – 0.940,* **content-aware prior – 0.951**.

PSNR and SSIM. The benefit of using a spatially variant prior is more pronounced for images with large textured regions. If the image consists primarily of piecewise smooth objects such as buildings, the difference between the content-aware image prior and others is minor. Fig. 3.11 compares the visual quality of restored images.

Denoising The goal of denoising is to reconstruct a sharp image from a noisy observation given a noise level. We consider reconstructing clean images from degraded images at two noise levels: 5% and 10%. Fig. 3.12 shows the measured PSNR and SSIM for the denoising task.

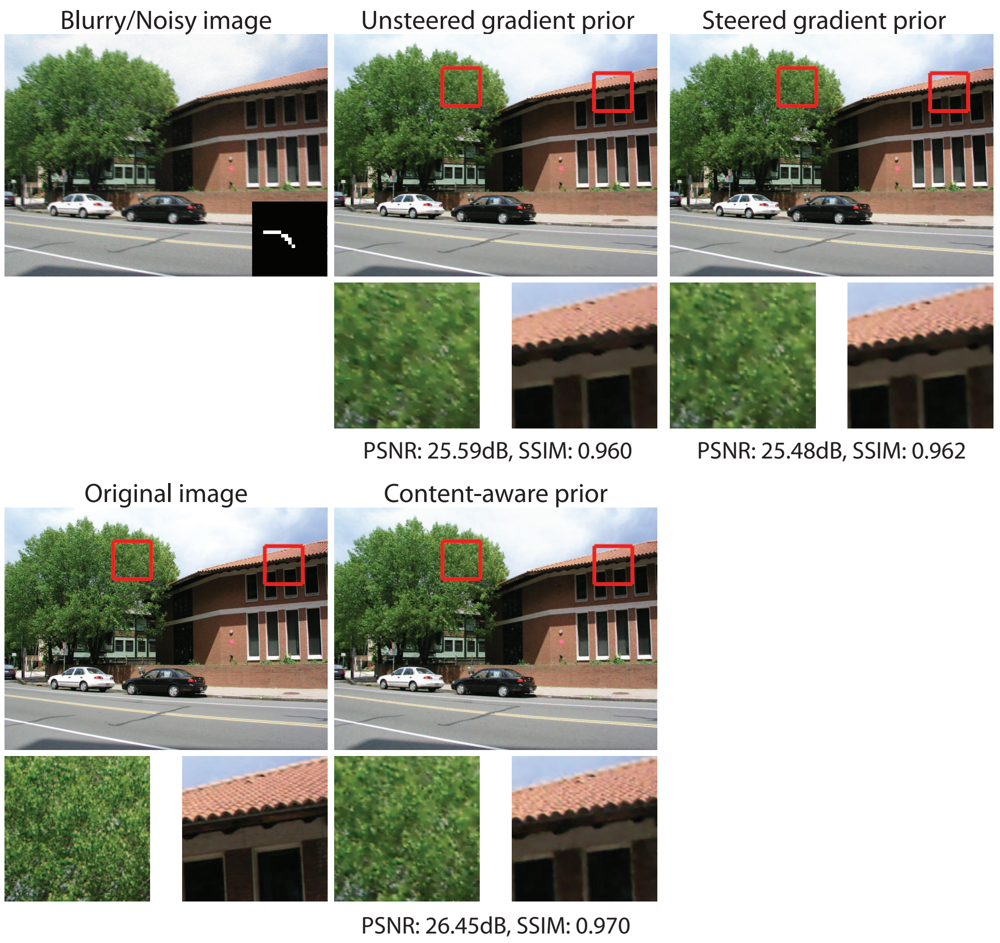


Figure 3.11: *Adapting the image prior to textures leads to better reconstructions. The red box indicate the cropped regions.*

When the noise level is low (5%), the content-aware prior reconstructs images with lower PSNR compared to competing methods. One explanation is that the content-aware prior may not remove all the noise in textured regions (such as trees) because the gradient statistics of noise is similar to that of the underlying texture. Such noise, however, does not disturb the visual quality of textures. The SSIM measure, which is better correlated with the perceptual quality [85], indicates that the content-aware image prior performs slightly worse, if not comparably, compared to other methods at a 5% noise level. The top row of Fig. 3.13 shows that at a 5% noise level, reconstructed images are visually similar. It's worth noting that when the noise level is low, image degradation is only moderate so that reconstructed images do not depend

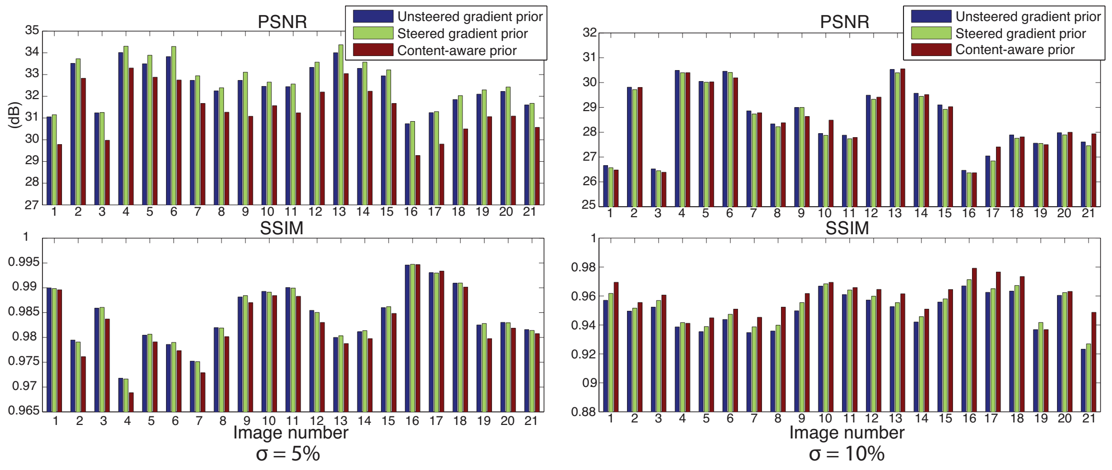


Figure 3.12: *Image denoising results : PSNR and SSIM. At* 5% *noise = Mean PSNR: unsteered gradient prior – 32.53 dB, steered gradient prior – 32.74 dB,* **content-aware prior – 31.42 dB**. *Mean SSIM: unsteered gradient prior – 0.984, steered gradient prior – 0.984,* **content-aware prior – 0.982**. *At* 10% *noise = Mean PSNR: unsteered gradient prior – 28.54 dB, steered gradient prior – 28.43 dB,* **content-aware prior – 28.52 dB**. *Mean SSIM: unsteered gradient prior – 0.950, steered gradient prior – 0.953,* **content-aware prior – 0.959**

heavily on the image prior.

When the noise level is high (10%), SSIM clearly favors images reconstructed using the content-aware prior. In this case, the observation term is weak, thus the image prior plays an important role in the quality of reconstructed images. The bottom row of Fig. 3.13 shows denoising results at a 10% noise level, supporting our claim that the content-aware image prior generates more visually pleasing textures. Fig. 3.14 shows more denoising performance comparisons.

Fig. 3.15 shows the result of deblurring a blurry image captured with a handheld camera. We estimate the blur kernel using Fergus *et al.* [31]. Again, textured regions are better reconstructed using our method.

User study We conducted a user study on Amazon Mechanical Turk to compare the visual quality of reconstructed images. Each user views two images, one reconstructed using the content-aware prior and another reconstructed using either the unsteered gradient prior or the steered gradient prior. The user has a choice of selecting the more visually pleasing image or selecting a *“There is no difference”* option.

We gathered about 20 user opinions for each comparison. In Fig. 3.16, we show the average

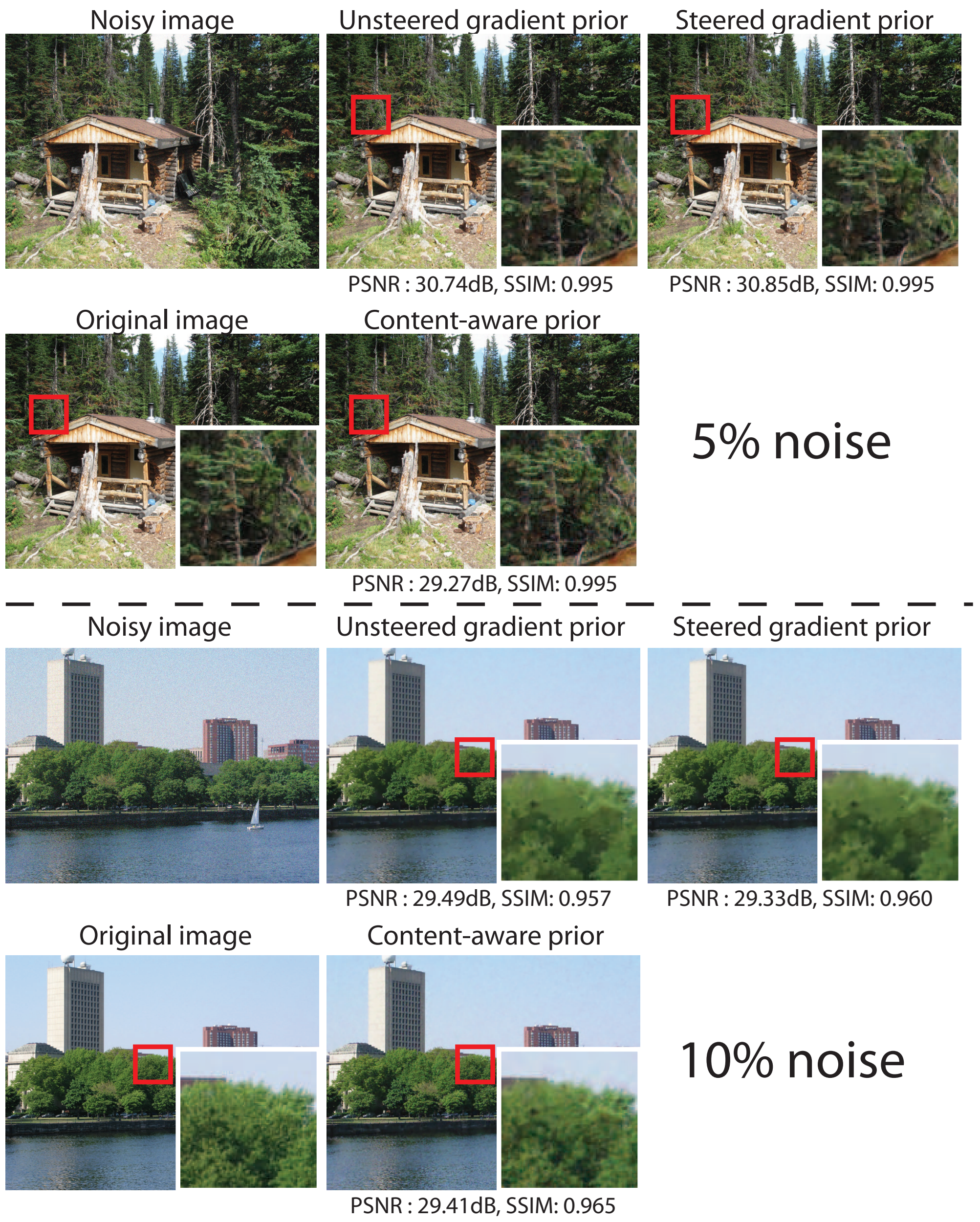


Figure 3.13: *The visual comparison of denoised images. The red box denotes the cropped region. At a* 5% *noise level, while the PSNR of our result is lower than those of competing algorithms, visually the difference is imperceptible. At a* 10% *noise level, the content-aware prior outperforms the others both in terms of the PSNR and the SSIM. Furthermore, the content-aware image prior restores visually more pleasing images.*

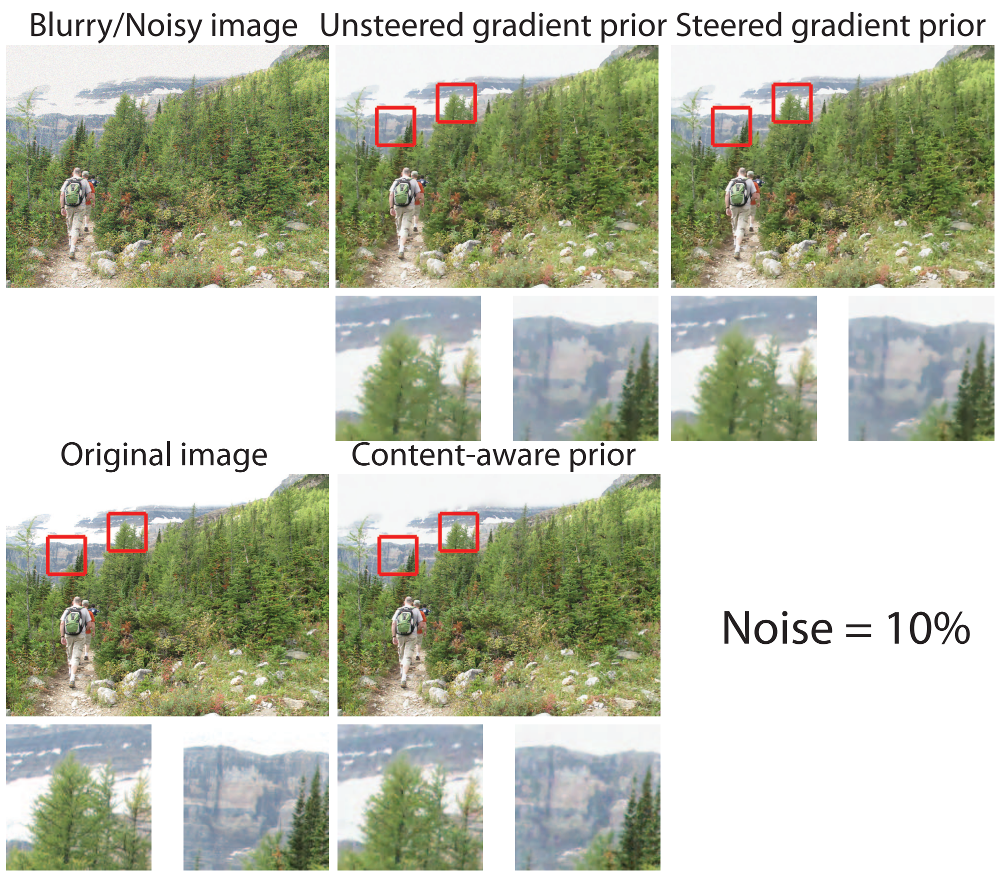


Figure 3.14: *More visual comparison of denoised images.*

user preference in each degradation scenario. Consistent with our expectations, users did not have a particular preference when the degradation was small (e.g. 5% noise), but at a high image degradation level users clearly favored the content-aware image prior over others.

3.5.2 Discussions

A limitation of our algorithm, which is shared by algorithms using a conditional random field model with hidden variables [43, 67, 80], is that hidden variables, such as the magnitude and/or orientation of an edge, or texture of a region, are estimated from the degraded input image or the image restored through other means. Any error from this preprocessing step induces error in the final result.

Although our algorithm improves the rendition of textures in restored images, the quality of the restored textures still depends on the weighting term w in Eq. 3.16. If w is too small,

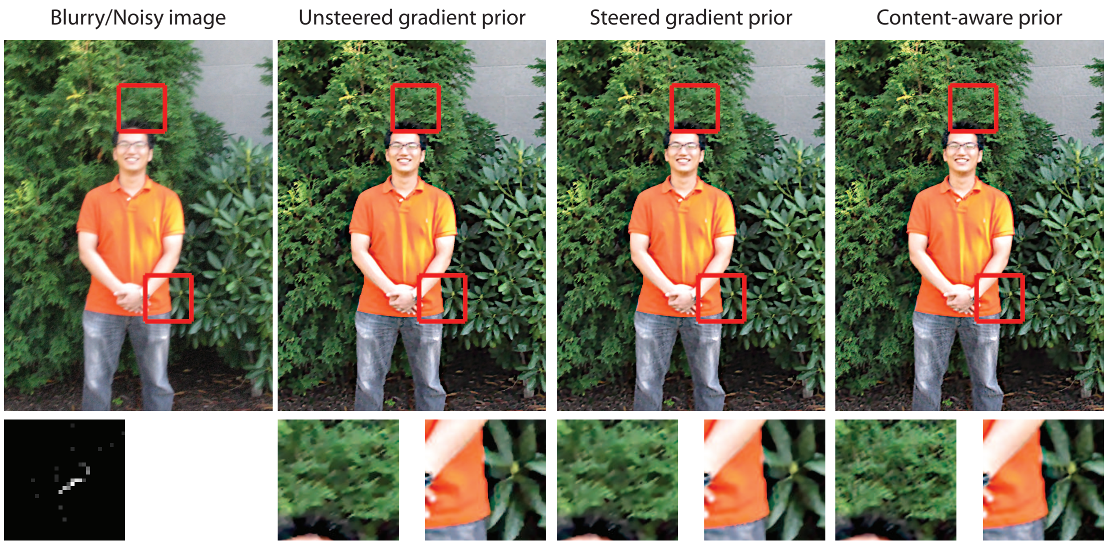


Figure 3.15: *The deconvolution of a blurred image taken with a hand-held camera. We estimate the blur kernel using Fergus et al. [31]. The red box denotes the cropped region. The textured region is better reconstructed using the content-aware image prior.*

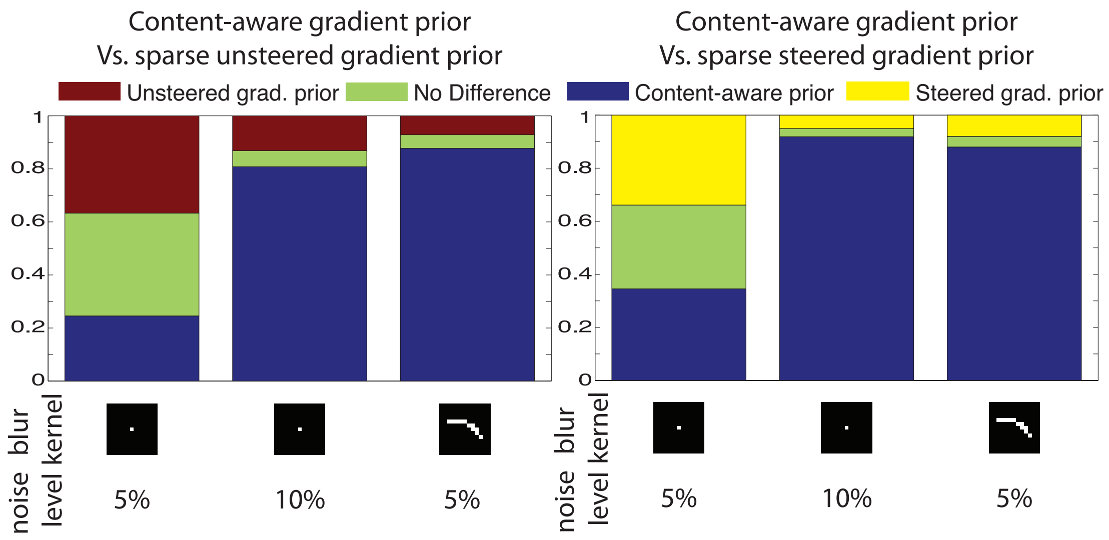


Figure 3.16: *This figure summarizes the user study results. The blue region corresponds to the fraction of users that favored our reconstructions. At a low degradation level, users do not prefer one method over another, but as the level of degradation increases, users clearly favor the content-aware image prior.*

smooth regions may become noisy, whereas w is too large, even the content-aware image prior could over-smooth textures. Also, while reconstructed images contain richer texture compared to those of previous techniques, textures are still smoother than that of the original image. In Chapter 4, we introduce a different deconvolution algorithm capable of synthesizing rich textures.

Another way to estimate a spatially variant prior is to segment the image into regions and assume a single prior within each segment. Unless we segment the image into many pieces, the estimated prior can be inaccurate. Also, the segmentation may inadvertently generate artificial boundaries in reconstructed images. Therefore, we estimate a distinct image prior for each pixel in the image.

3.6 Conclusion

We have explored the problem of estimating spatially variant gradient statistics in natural images, and exploited the estimated gradient statistics to adaptively restore different textural characteristics in image restoration tasks. We show that the content-aware image prior can restore piecewise smooth regions without over-smoothing textured regions, improving the visual quality of reconstructed images as verified through user studies. Adapting to textural characteristics is especially important when the image degradation is significant.

Chapter 4

Image restoration by matching gradient distributions

4.1 Introduction

THE content-aware image prior improves texture restoration. However, as discussed in Sec. 3.5.2, the restored texture is often not as rich as the original texture. We attribute this shortcoming to the use of a MAP estimator. When we use a MAP estimator to restore a degraded image, the gradient distribution of the reconstructed image is quite different from that of the original image. Most of the times, as shown in Fig. 4.1, the restored image has more gradients with small magnitude compared to the original image, which indicates that the restored image is smoother.

To address this issue, we introduce an alternative image restoration strategy that is capable of synthesizing rich textures. The main idea is that the difference between gradient distributions is an indication that the restored image is not as natural as the original image. Therefore, our deconvolution algorithm matches the reconstructed image's gradient distribution to its reference distribution (i.e. the gradient distribution of the original image). Essentially, our method imposes a global constraint on gradients, as opposed to imposing local constraints by simply penalizing individual gradient as in a MAP estimator. To estimate the reference distribution directly from a degraded input image, we adapt the method introduced in Chapter 3. User study substantiates the claim that images reconstructed by matching gradient distributions are visually more pleasing compared to those reconstructed using MAP estimators.

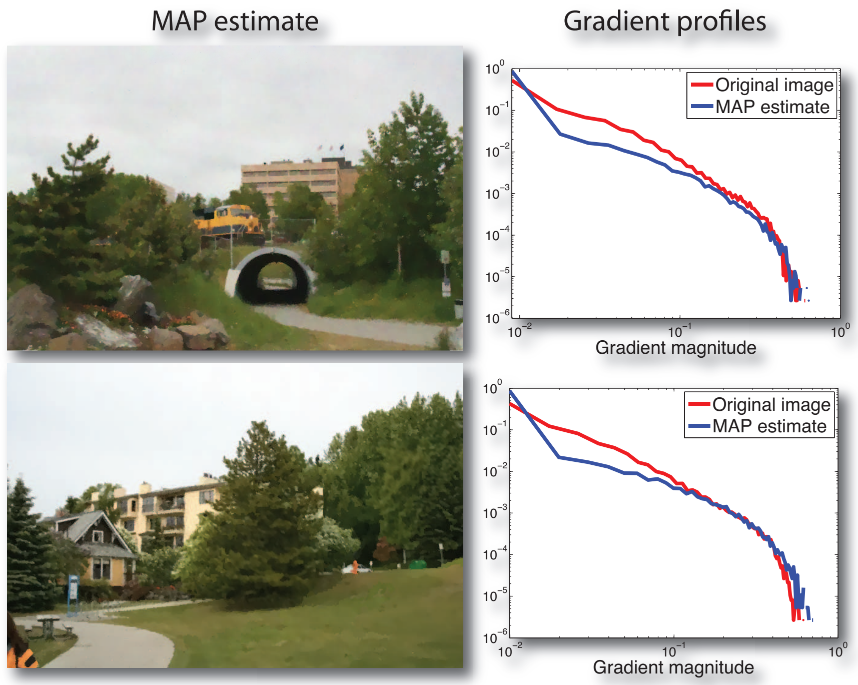


Figure 4.1: *When we use a MAP estimator restore a degraded image, the gradient distribution of the reconstructed image can be quite different from that of the original image. We consider the difference in gradient distributions as an indicator that the restored image is not as natural-looking. We present a method that matches the reconstructed image's gradient distribution to its reference distribution (i.e. the gradient distribution of the original image) to hallucinate visually pleasing textures.*

4.2 Related work

Matching gradient distributions has been well investigated in the texture synthesis literature. Heeger and Bergen [39] synthesize textures by matching wavelet sub-band histograms to those of a desired texture. Portilla and Simoncelli [62] match joint statistics of wavelet coefficients to synthesize homogeneous textures. Kopf *et al.* [44] introduce an inhomogeneous texture synthesis technique by matching histograms of texels (texture elements).

Matching gradient distributions in image restoration is not entirely new. Li and Adelson [51] introduce a two-step image restoration algorithm that first reconstructs an image using an exemplar-based technique similar to Freeman *et al.* [33], and warp the reconstructed image's gradient distribution to a reference gradient distribution using Heeger and Bergen's method [39]. Woodford *et al.* [89] propose a MAP estimation framework called a marginal probability field (MPF) that matches a histogram of low-level features, such as gradients or texels, for computer vision tasks including denoising. MPF requires that one bins features to form a discrete histogram; we propose a distribution matching method that by-passes this binning process. Also, Woodford *et al.* [89] use an image prior estimated from a database of images and use the same global prior to reconstruct images with different textures. In contrast, we estimate the image prior directly from the degraded image for each textured region. Schmidt *et al.* [72] match the gradient distribution through sampling. As with Woodford *et al.* [89], Schmidt *et al.* also use a single global prior to reconstruct images with different textures, which causes noisy renditions in smooth regions. HaCohen *et al.* [37] explicitly integrate texture synthesis to image restoration, specifically for an image up-sampling problem. To restore textures, they segment a degraded image and replace each texture segment with textures in a database of images.

■ 4.3 Characteristics of MAP estimators

In this section, we illustrate why MAP estimators with a sparse prior recover unrealistic, piecewise smooth renditions as illustrated in Fig. 4.1. Let B be a degraded image, k be a blur kernel, $\otimes$ be a convolution operator, and I be a latent image. A MAP estimator solves the following regularized problem:

$$\hat{I} = \underset{I}{\operatorname{argmin}} \left\{ \frac{\|B - k \otimes I\|^2}{2\eta^2} + w \sum_m \rho(\nabla_m I) \right\} \tag{4.1}$$

where η^2 is an observation noise variance, m indexes gradient filters, and ρ is a robust function that favors sparse gradients. We parameterize the gradient distribution using a generalized Gaussian distribution. In this case, $\rho(\nabla I) = -\ln(p(\nabla I; \gamma, \lambda))$, where the prior $p(\nabla I; \gamma, \lambda)$ is given as follows:

$$p(\nabla I; \gamma, \lambda) = \frac{\gamma \lambda^{(\frac{1}{\gamma})}}{2\Gamma(\frac{1}{\gamma})} \exp(-\lambda \|\nabla I\|^{\gamma}) \tag{4.2}$$

Γ is a Gamma function and shape parameters γ, λ determine the shape of the distribution. In most MAP-based image reconstruction algorithms, gradients are assumed to be independent for computational efficiency: $p(\nabla I; \gamma, \lambda) = \frac{1}{Z} \prod_{i=1}^{N} p(\nabla I_i; \gamma, \lambda)$, where i is a pixel index, Z is a partition function, and N is the total number of pixels in an image.

A MAP estimator balances two competing forces: the reconstructed image $\hat{I}$ should satisfy the observation model while conforming to the image prior. Counter-intuitively, the image prior term, assuming independence among gradients, *always* favors a flat image to any other images, even a natural image. Therefore, the more the MAP estimator relies on the image prior term, which is often the case when the image degradation is severe, the more the reconstructed image becomes piecewise smooth.

One way to explain this property is that the independence among local gradients fails to capture the global statistics of gradients for the whole image. The image prior tells us that gradients in a natural image *collectively* exhibit a sparse gradient profile, whereas the independence assumption of gradients forces us to minimize each gradient *independently*, always favoring a flat image. Nikolova [58] provides a theoretic treatment of MAP estimators in general to show its deficiency.

We could remove the independence assumption and impose a joint prior on all gradients, but this approach is computationally expensive. This paper introduces an alternative method to impose a global constraint on gradients – that a reconstructed image should have a gradient distribution similar to a reference distribution.

■ 4.4 Image reconstruction

In this section, we develop an image reconstruction algorithm that minimizes the KL divergence between the reconstructed image's gradient distribution and its reference distribution. This distance penalty plays the role of a global image prior that steers the solution away from piecewise smooth images.

Let $q_E(\nabla I)$ be an empirical gradient distribution of an image I, and q_R be a reference distribution. We measure the distance between distributions q_E and q_R using the Kullback-Leibler (KL) divergence:

$$KL(q_E||q_R) = \int_{\nabla I} q_E(\nabla I) \ln\left(\frac{q_E(\nabla I)}{q_R(\nabla I)}\right) d(\nabla I) \tag{4.3}$$

An empirical distribution q_E is parameterized using a generalized Gaussian distribution $p(\nabla I; \gamma, \lambda)$ (Eq. 4.2). Given gradient samples, ∇I_i, where i indexes samples, we estimate the shape parameters γ_E, λ_E of an empirical gradient distribution q_E by minimizing the log-likelihood:

$$[\gamma_E, \lambda_E] = \underset{\gamma,\lambda}{\operatorname{argmin}} \left\{ -\sum_{i=1}^{N} \frac{1}{N} \ln\left(p(\nabla I_i; \gamma, \lambda)\right) \right\} \tag{4.4}$$

This is equivalent to minimizing the KL divergence between gradient samples ∇I and a generalized Gaussian distribution (see Sec. 3.3.1). We use the Nelder-Mead optimization method

```
% Initial image estimate to start iterative minimization
$\hat{I}^0 = \text{argmin}_I \left\{ \frac{\|B-k\otimes I\|^2}{2\eta^2} + w_1\lambda_R \|\nabla I\|^{\gamma_R} \right\}$
Update ${q_E}^0$ using Eq. 4.4
% Iterative minimization
for l = 1 ... 10 do
    % KL distance penalty term update
    $\rho_G^l(\nabla I) = \frac{1}{N} \ln \left( \frac{{q_E}^{(l-1)}(\nabla I)}{q_R(\nabla I)} \right)$
    % Image reconstruction
    $\hat{I}^l = \text{argmin}_I \left\{ \frac{\|B-k\otimes I\|^2}{2\eta^2} + w_1\lambda_R \|\nabla I\|^{\gamma_R} + w_2 \rho_G^l(\nabla I) \right\}$
    Update ${q_E}^l$ using Eq. 4.4
end for
$\hat{I} = \hat{I}^{10}$
```

Algorithm 3: `MAP with KL penalty`

[45] to solve Eq. 4.4.

■ 4.4.1 Failure of penalizing KL divergence directly

To motivate our algorithm in Sec. 4.4.2, we first introduce a method that penalizes the KL divergence between an empirical gradient distribution q_E and a reference distribution q_R. We show that the performance of this algorithm is sensitive to the parameter setting and that the algorithm may not always converge. In Sec. 4.4.2, we extend this algorithm to a more stable algorithm called Iterative Distribution Reweighting (IDR).

First, we augment the MAP estimator (Eq. 4.1) with KL divergence:

$$\hat{I} = \underset{I}{\text{argmin}} \left\{ \frac{\|B - k \otimes I\|^2}{2\eta^2} + w_1\lambda_R \|\nabla I\|^{\gamma_R} + w_2 KL(q_E || q_R) \right\} \tag{4.5}$$

where w_2 determines how much to penalize the KL divergence.[1] It's hard to directly solve Eq. 4.5 because the KL divergence is a non-linear function of a latent image I. Therefore we solve Eq. 4.5 iteratively.

Algorithm 3, shown in pseudocode, solves Eq. 4.5 iteratively. We can describe Algorithm 3 qualitatively as follows: if q_E has more gradients of a certain magnitude than q_R, ρ_G penalizes those gradients *more* in the following iteration; if q_E has fewer gradients of a certain magnitude

[1] In Eq. 4.5, we have replaced the summation over multiple filters in Eq. 4.1, i.e. $\sum_m \lambda_m \|\nabla_m I\|^{\gamma_m}$, with a single derivative filter to reduce clutter, but the derivation can easily be generalized to using multiple derivative filters. We use four derivative filters in this work: x, y derivative filters and x-y, and y-x diagonal derivative filters.

than q_R, ρ_G penalizes those gradients *less* in the following iteration. Therefore, at each iteration, the solution will move in the "correct" direction. Fig. 4.2 illustrates the procedure. The full derivation of the algorithm details is available in Appendix A.

We can show that the penalty function ρ_G in Algorithm 3 is one way to evaluate the KL divergence between the empirical distribution q_E and the reference distribution q_R.

Claim 2. *Let q_E be a parametric distribution of samples $x_i, i = 1...N$ and let q_R be a fixed parametric distribution. Then we can represent the KL divergence between samples q_E and q_R as follows:*

$$KL(q_E||q_R) = \sum_i^N \rho_G(x_i) = \sum_i^N \left\{ \frac{1}{N} \ln \left(\frac{q_E(x_i)}{q_R(x_i)} \right) \right\} \tag{4.6}$$

Proof. The KL divergence between q_E and q_R is defined as follows:

$$KL(q_E||q_R) = \int_z q_E(z) \ln \left(\frac{q_E(z)}{q_R(z)} \right) dz \tag{4.7}$$

There are different ways to represent the the parametric distribution q_E. We can parameterize the distribution of samples using a generalized Gaussian distribution as follows:

$$q_E(z) = \frac{\gamma_E \lambda_E^{\left(\frac{1}{\gamma_E}\right)}}{2\Gamma(\frac{1}{\gamma_E})} \exp(-\lambda_E \|z\|^{\gamma_E}) \tag{4.8}$$

where the shape parameters γ_E, λ_E are fitted to samples x_i using Eq. 4.4. We can also parameterize the distribution of samples x_i as follows:

$$\tilde{q_E}(z) = \frac{1}{N} \sum_i^N \delta(z - x_i) \tag{4.9}$$

Therefore,

$$\begin{aligned} KL(q_E||q_R) &= \int_z q_E(z) \ln \left(\frac{q_E(z)}{q_R(z)} \right) dz \\ &= \int_z \tilde{q_E}(z) \ln \left(\frac{q_E(z)}{q_R(z)} \right) dz \\ &= \sum_i^N \left\{ \frac{1}{N} \ln \left(\frac{q_E(x_i)}{q_R(x_i)} \right) \right\} \\ &= \sum_i^N \rho_G(x_i) \end{aligned} \tag{4.10}$$

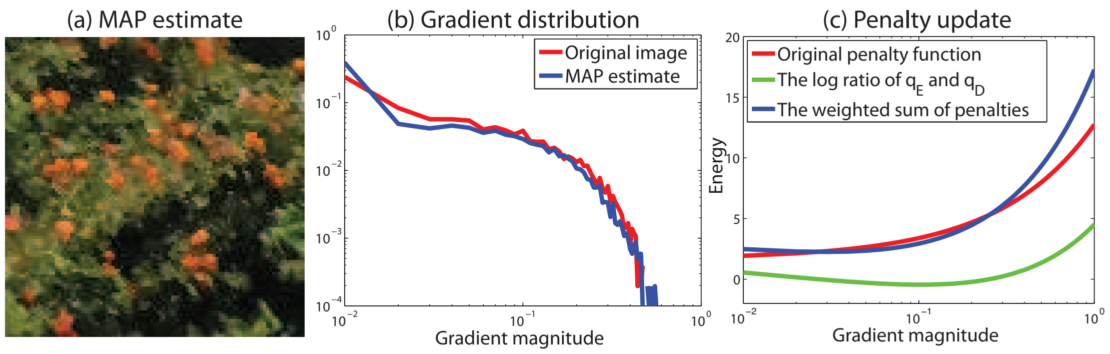


Figure 4.2: *This figure illustrates Algorithm 3. Suppose we deconvolve a degraded image using a MAP estimator. (b) shows that the x-gradient distribution of the MAP estimate in (a) does not match that of the original image. (c) Our algorithm adds the log ratio of q_E and q_R to the original penalty (i.e. $\lambda_R \|\nabla I\|^{\gamma_R}$) such that the weighted sum of the two penalty terms encourages a better distribution match in the following iteration.*

□

Algorithm analysis The behavior of Algorithm 3 depends on the value of w_2. When w_2 is small, the reconstructed image is similar to the MAP estimate. On the other hand, when w_2 is large, the algorithm oscillates around the desired solution (Fig. 4.3): the algorithm "ping-pong's" between a noisy solution and a piecewise smooth solution. For instance, suppose the current image estimate is piecewise smooth. The algorithm would then encourage more pixels with larger derivatives in the next iteration, which makes the subsequent solution noisier. In the following iteration, to reduce the derivative magnitudes to smooth noise, the algorithm penalizes gradients more severely to better match the reference distribution, in which case the image becomes piecewise smooth again, exhibiting an oscillatory behavior. In fact, when w_2 is very large, the linearized system (in Appendix A, Eq. 15) becomes indefinite, in which case the minimum residual method [70] cannot be used to solve the linearized system. To mitigate the reliability issue and to damp possible oscillations around the desired solution, we develop an iterative distribution reweighting algorithm.

■ 4.4.2 The iterative distribution reweighting (IDR)

We extend Algorithm 3 to reduce oscillations around the correct solution and to reduce sensitivity to parameter values. We achieve this by modifying the regularization function ρ_G in Algorithm 3. Our technique is motivated by perceptron algorithms that iteratively adjust a decision

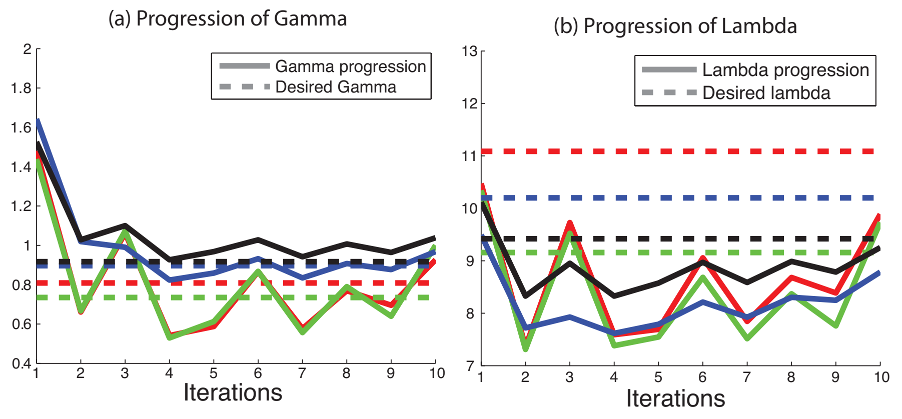


Figure 4.3: *We illustrate the operation of Algorithm 3 in terms of the γ_E, λ_E progressions. Different colors correspond to different gradient filters. Oftentimes, Algorithm 3 does not converge to a stable point, but oscillates around the desired solution.*

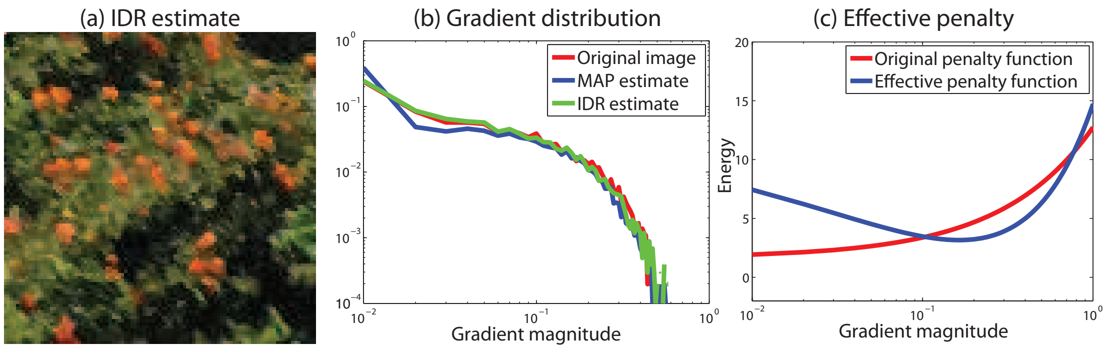


Figure 4.4: *The IDR deconvolution result. (a) shows the deconvolved image using IDR, and (b) compares the gradient distribution of images reconstructed using the MAP estimator and IDR. (c) The effective penalty after convergence (i.e. $w_1 \lambda_R \|\nabla I\|^{\gamma_R} + w_2 \sum_{l=1}^{10} \frac{1}{N} \ln\left(\frac{q_E(\nabla I)}{q_R(\nabla I)}\right)$) penalizes gradients with small and large magnitude more than gradients with moderate magnitude.*

boundary to minimize classification error. In our case, we iteratively adjust the regularization function to match the empirical gradient distribution to the reference gradient distribution.

To do so, instead of using the KL divergence as a regularization term ρ_G as in Algorithm 3, we set ρ_G as the *sum* of the KL divergences over previous iterations. Algorithm 4 shows the pseudocode for IDR.

IDR iteratively adjusts the penalty function ρ_G by the ratio of distributions q_E and q_R, thus the name the *iterative distribution reweighting (IDR)*. The benefit of IDR is that it reaches

```
% Initial image estimate to start iterative minimization
$\hat{I}^0 = \text{argmin}_I \left\{ \frac{\|B - k \otimes I\|^2}{2\eta^2} + w_1 \lambda_R \|\nabla I\|^{\gamma_R} \right\}$
Update ${q_E}^0$ using Eq. 4.4
% Iterative minimization
for l = 1 ... 10 do
    % Accumulating the KL divergence
    $\rho_G^l(\nabla I) = \rho_G^{(l-1)}(\nabla I) + \frac{1}{N} \ln \left( \frac{{q_E}^{(l-1)}(\nabla I)}{q_R(\nabla I)} \right)$
    % Image reconstruction
    $\hat{I}^l = \text{argmin}_I \left\{ \frac{\|B - k \otimes I\|^2}{2\eta^2} + w_1 \lambda_R \|\nabla I\|^{\gamma_R} + w_2 \rho_G^l(\nabla I) \right\}$
    Update ${q_E}^l$ using Eq. 4.4
end for
$\hat{I} = \hat{I}^{10}$
```

Algorithm 4: `The iterative distribution reweighting (IDR)`

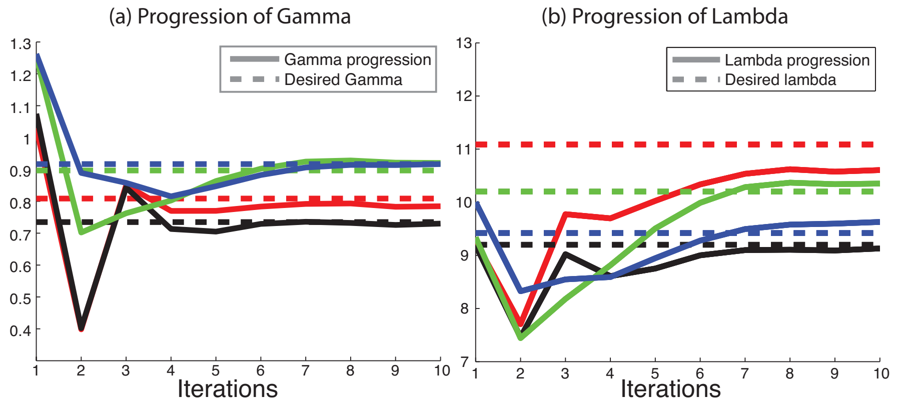


Figure 4.5: *This figure shows how the γ_E, λ_E progress from one iteration to the next. Different colors correspond to different gradient filters. We observe that the algorithm converges to a stable point in about 8 iterations.*

convergence when $q_E = q_R$. [2] We can also view the ρ_G update equation as damping the KL divergence with the sum of previous KL divergences, thereby smoothing oscillations. We can easily modify derivations in Appendix A to derive details for Algorithm 4. We illustrate the operation of IDR in Fig. 4.4, and show how γ_E, λ_E changes from one iteration to the next in Fig. 4.5. Observe that γ_E, λ_E no longer oscillate as in Fig. 4.3.

In Fig. 4.6, we test IDR for deblurring a single texture, assuming that the reference distri-

[2]This statement does not mean that the algorithm will converge only if $q_E = q_R$; the algorithm can converge to a local minimum.

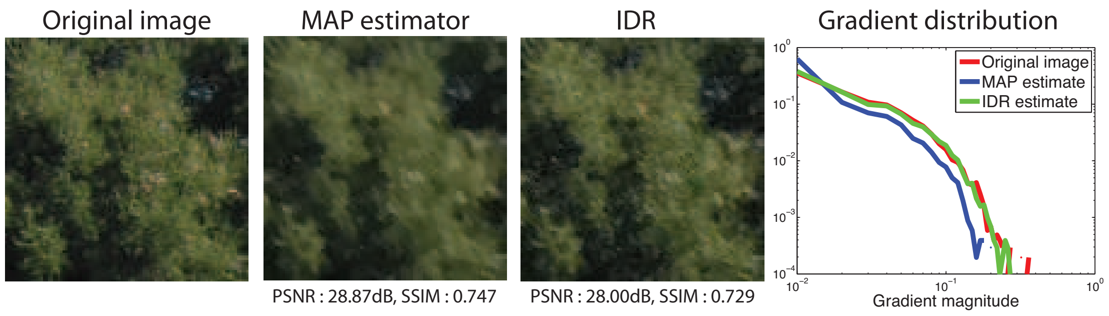


Figure 4.6: *We compare the deblurring performance of a MAP estimator and IDR. IDR reconstructs visually more pleasing mid-frequency textures compared to a MAP estimator.*

bution q_R is known a priori. We synthetically blur the texture using the blur kernel shown in Fig. 4.8 and add 5% Gaussian noise to the blurred image. We deblur the image using a MAP estimator and using IDR, and compare the reconstructions. For all examples in this paper, we use $w_1 = 0.025, w_2 = 0.0025$. We observe that the gradient distribution of the IDR estimate matches the reference distribution better than that of the MAP estimate, and visually, the texture of the IDR estimate better matches the original image's texture. Although visually superior, the peak signal-to-noise ratio (PSNR) / gray-scale SSIM [85] of the IDR estimate are lower than those of the MAP estimate. This occurs because IDR may not place the gradients at exactly the right position. Degraded images do not strongly constrain the position of gradients, in which case our algorithm disperses gradients to match the gradient distribution, which would lower the PSNR / SSIM.

Algorithm analysis IDR matches a *parametrized* gradient distribution, and therefore the algorithm is inherently limited by the accuracy of the fit. The behavior of IDR is relatively insensitive to the weighting term w_2, but a large w_2 can destabilize the minimum residual algorithm [70] that solves the linearized system in Eq. 15.

In most cases, IDR reliably reconstructs images with the reference gradient distribution. However, there are cases in which the algorithm settles at a local minimum that does not correspond to the desired texture. This usually occurs when the support of derivative filters is large and when we use many derivative filters to regularize the image. For instance, suppose we want to match the gradient histogram of a 3×3 filter. The algorithm needs to update 9 pixels to change the filter response at the center pixel, but updating 9 pixels also affects filter responses of 8 neighboring pixels. Having to match multiple gradient distributions at the same time increases the complexity. To control the complexity, we match four two-tap derivative filters. Adapting derivative filters to local image structures using steerable filters [20, 32, 67] may further improve the rendition of oriented textures, but it is not considered in this work.

■ 4.4.3 Reference distribution q_R estimation

We parameterize a reference distribution q_R using a generalized Gaussian distribution. Unfortunately, one often does not know *a priori* what q_R should be. Previous work estimates q_R from a database of natural images [31, 89] or hand-picks q_R through trial and error [48]. We adopt the image prior estimation technique introduced in Sec. 3.4 to estimate q_R directly from a degraded image, as we will now describe.

We first deconvolve a degraded image B using a MAP estimator (Eq. 4.1) with a hand-picked image prior, tuned to restore different textures reasonably well at the expense of a slightly noisy image reconstruction (i.e. a relatively small gradient penalty). In this paper, we set the parameters of the image prior as $[\gamma = 0.8, \lambda = 4, w_1 = 0.01]$ for all images. To reduce deconvolution noise, we down-sample the reconstructed image. We fit gradients from the down-sampled image to a generalized Gaussian distribution, as in Eq. 4.4, to estimate the reference distribution q_R. While fine details can be lost through down-sampling, empirically, the estimated reference distribution q_R is accurate enough for our purpose.

Our image reconstruction algorithm assumes that the texture is homogeneous (i.e. a single q_R). In the presence of multiple textures within an image, we segment the image and estimate separate reference distribution q_R for each segment: we use the EDISON segmentation algorithm [22] to segment an image into about 20 regions. Fig. 4.7 illustrates the image deconvolution process for spatially varying textures.

■ 4.5 Experiments

Deconvolution experiments We synthetically blur sharp images with the blur kernel shown in Fig. 4.8, add 2% noise, and deconvolve them using competing methods. We compare the performance of IDR against four other competing methods: (i) a MAP estimator with a sparse gradient prior [48], (ii) a MAP estimator with a sparse prior adapted to each segment (iii) a MAP estimator with a two-color prior [43] (iv) a MAP estimator with a content-aware image prior. We blur a sharp image using the kernel shown on the right, add 2% noise to it, and restore images using the competing methods. Fig. 4.8 shows experimental results. As mentioned in

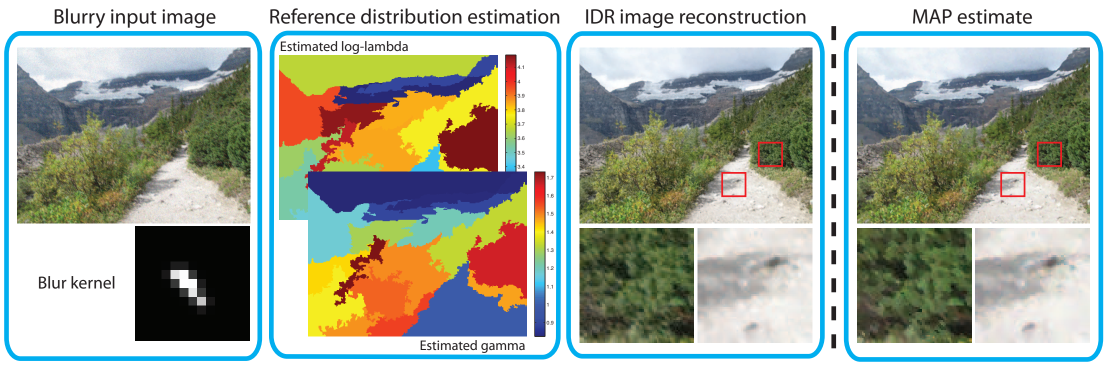


Figure 4.7: *For an image with spatially varying texture, our algorithm segments the image into regions of homogeneous texture and matches the gradient distribution in each segment independently. Compared to MAP estimators, our algorithm reconstructs visually more pleasing textures.*

Sec. 4.4.2, IDR does not perform the best in terms of PSNR / SSIM. Nevertheless, IDR reconstructs mid-frequency textures better, for instance fur details. Another interesting observation is that the content-aware image prior performs better, in terms of PSNR/SSIM, than simply adjusting the image prior to each segment's texture. By using segment-adjusted image prior, we observe segmentation boundaries that are visually disturbing. Another set of comparisons is shown in Fig. 4.9.

In Fig. 4.10, we compare the denoising performance of IDR to that of a marginal probability field (MPF) by Woodford *et al.* [89] at two noise levels (their implementation only handles grayscale, square images). Using MPF for denoising has two drawbacks. First, MPF quantizes intensity levels and gradient magnitudes to reduce computation. MPF quantizes 256 (8-bit) intensity levels to 64 intensity levels (6-bit), and it bins 256 (8-bit) gradient magnitudes to 11 slots. These quantizations would accentuate spotty noise in reconstructed images. IDR adopts a continuous optimization scheme that does not require any histogram binning or intensity quantization, therefore it does not suffer from quantization noise. Second, Woodford *et al.* [89] estimate the reference gradient distribution from a database of images, and use the *same* prior to denoise different images. This can be problematic because different images have different reference distributions q_R, but MPF would enforce the same gradient profile on them. Also, MPF does not adapt the image prior to the underlying texture, treating different textures the same way. Therefore, MPF distributes gradients uniformly across the image, even in smooth regions, which can be visually disturbing. IDR addresses these issues by estimating a reference

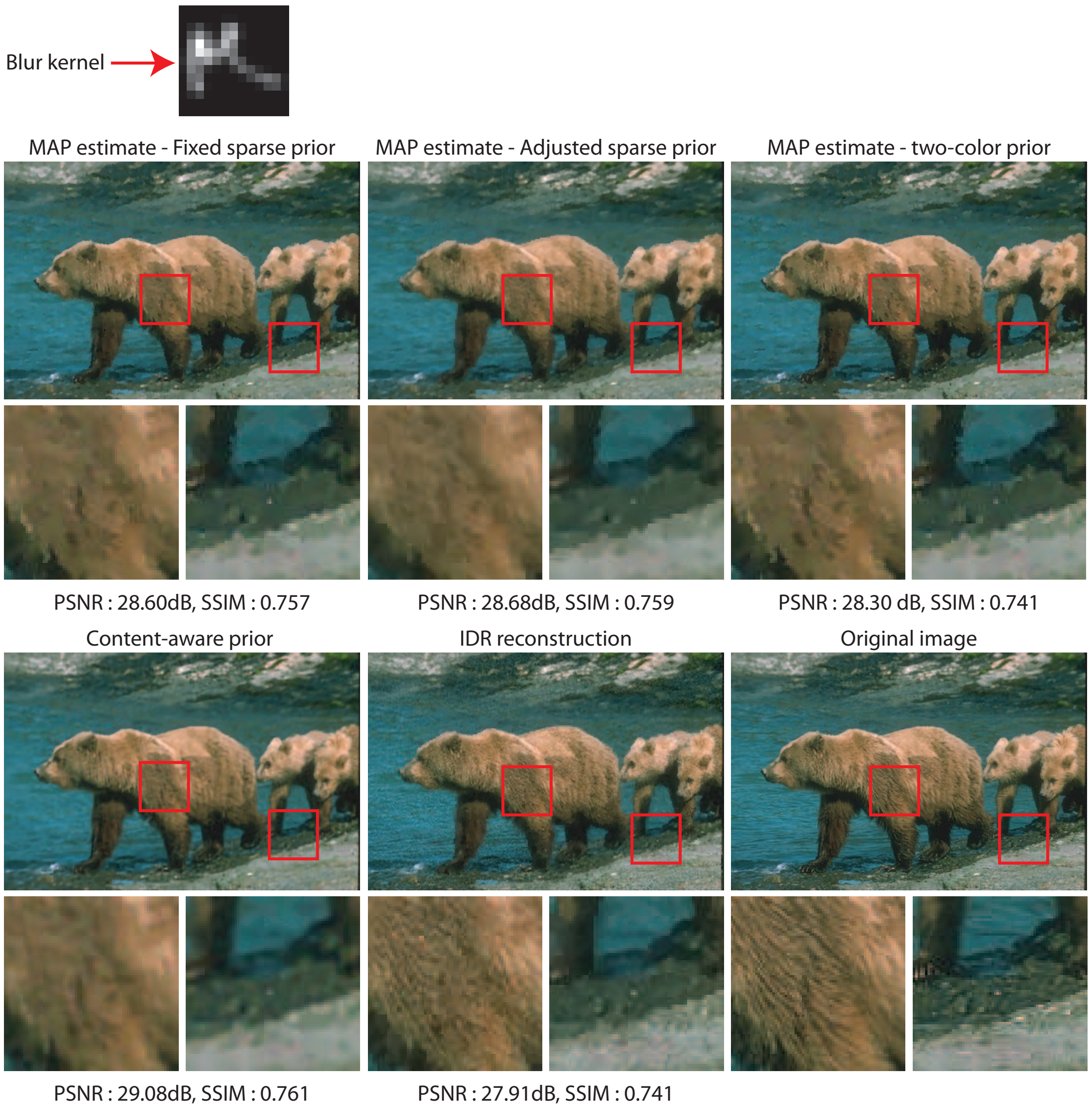

Blur kernel
MAP estimate - Fixed sparse prior
MAP estimate - Adjusted sparse prior
MAP estimate - two-color prior
PSNR : 28.60dB, SSIM : 0.757
PSNR : 28.68dB, SSIM : 0.759
PSNR : 28.30 dB, SSIM : 0.741
Content-aware prior
IDR reconstruction
Original image
PSNR : 29.08dB, SSIM : 0.761
PSNR : 27.91dB, SSIM : 0.741

Figure 4.8: *We compare the performance of IDR against four other competing methods: (i) a MAP estimator with a sparse gradient prior [48], (ii) a MAP estimator with a sparse prior adapted to each segment (iii) a MAP estimator with a two-color prior [43] (iv) a MAP estimator with a content-aware image prior. The red box indicate the cropped regions. Although the PSNR and the SSIM of our results are often lower than those of MAP estimators, IDR restores more visually pleasing textures (see bear furs).*

distribution q_R from an input image and by adapting q_R to spatially varying texture.

At a high degradation level, such as a noise level of 31.4%, our reference distribution es-

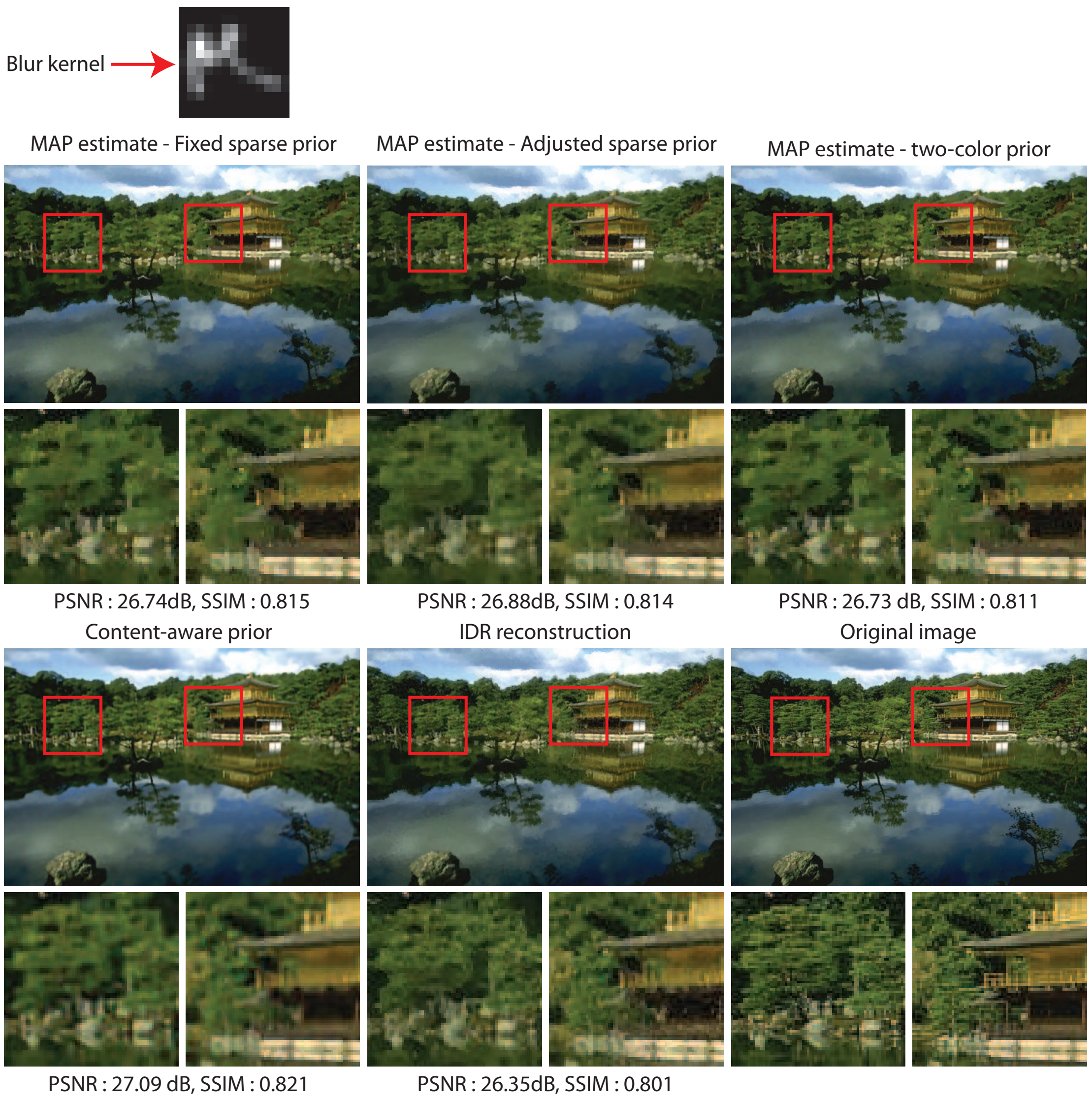
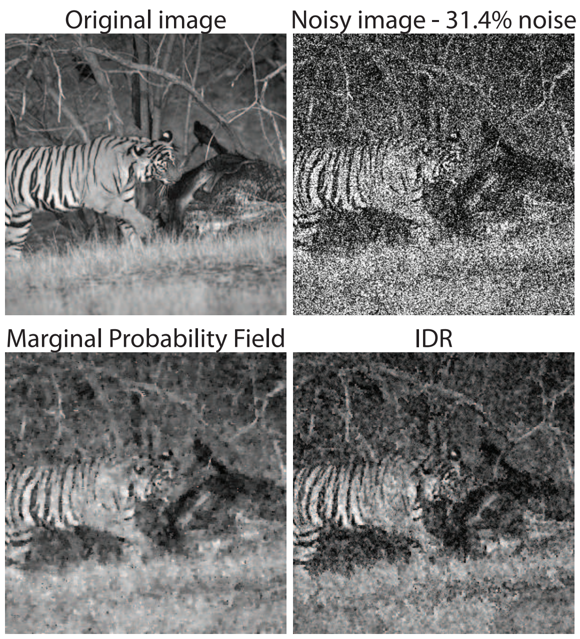

Blur kernel
MAP estimate - Fixed sparse prior
MAP estimate - Adjusted sparse prior
MAP estimate - two-color prior
PSNR : 26.74dB, SSIM : 0.815
PSNR : 26.88dB, SSIM : 0.814
PSNR : 26.73 dB, SSIM : 0.811
Content-aware prior
IDR reconstruction
Original image
PSNR : 27.09 dB, SSIM : 0.821
PSNR : 26.35dB, SSIM : 0.801

Figure 4.9: *This figure provides another set of deconvolution performance comparisons.*

timation algorithm can be unstable. In Fig. 4.10, our q_R estimation algorithm returns a distribution that has more "large" derivatives and less "small" derivatives (dotted line in Fig. 4.10), which manifests itself as a noisy IDR reconstruction. In contrast, MPF restores a plausible image, but this is somewhat coincidental in that the reference distribution that MPF imposes is quite similar to that of the original image.

At a more reasonable degradation level (15% noise), as shown in Fig. 4.11, our algorithm estimates a reference distribution that is very similar to that of the original image. Given a more



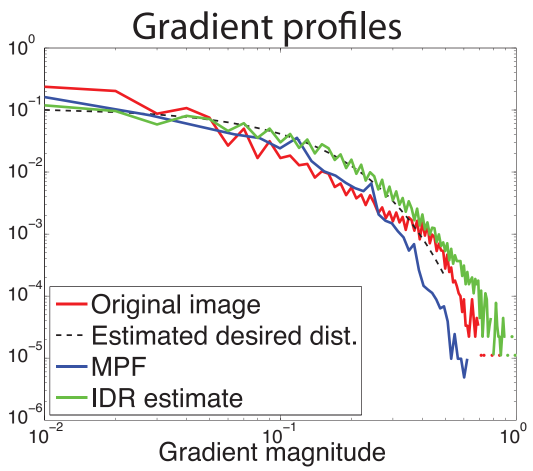


Figure 4.10: *We compare the denoising performance of IDR and the marginal probability field (MPF) [89] at a high noise level (31.4%). At such a high noise level, our distribution estimation algorithm is not reliable, thus IDR restores a noisy rendition compared to MPF.*

accurate reference distribution, IDR restores a visually pleasing image. On the other hand, MPF restores a noisy rendition because the reference distribution is quite different from that of the original image. Also note that the gradient distribution of the restored image is very similar to that of the restored image in Fig. 4.10, which illustrates our concern that using a single image prior for different images would degrade the image quality.

Segmenting images to regions and deconvolving each region separately may generate artificial texture boundaries, as in Fig. 4.12. While this rarely occurs, we could mitigate these artifacts using a texture-based segmentation algorithm rather than EDISON [22], which is a color-based segmentation algorithm.

User study IDR generates images with rich texture but with lower PSNR/SSIM than MAP estimates. To test our impression that images reconstructed by IDR are more visually pleasing, we performed a user study on Amazon Mechanical Turk.

We considered seven image degradation scenarios: noisy observations with 5%, 10%, 15% noise, blurry observations with a small blur and 2%, 5%, 7% noise, and a blurry observation with a moderate-size blur and 2% noise. For each degradation scenario, we randomly selected

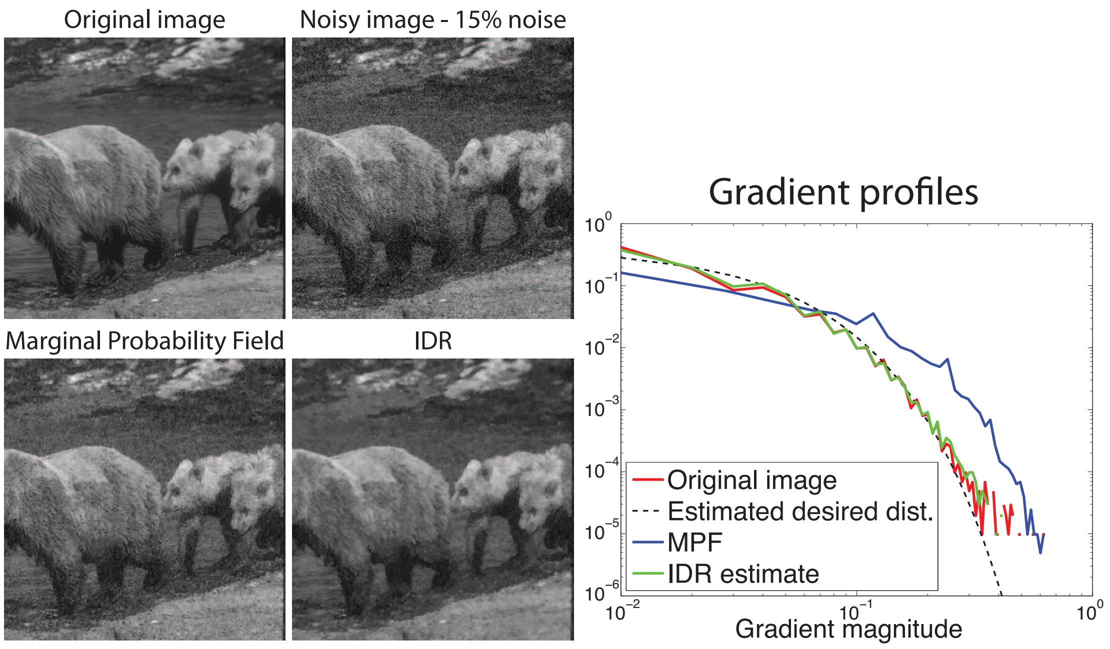


Figure 4.11: *In this figure, we compare the denoising performance of IDR and the MPF [89] at a moderate noise level (15%). At this noise level, the predicted gradient distribution matches the underlying image well, and the IDR restores a more natural image.*

4 images from a dataset of 13 images (roughly 700×500 pixels), and reconstructed images using a MAP estimator with a fixed sparse prior (i.e. the same sparse prior across the whole image), an adjusted sparse prior, and IDR.

As in Sec. 3.5, we showed users two images side-by-side, one reconstructed using our algorithm and another reconstructed using one of the two MAP estimators (i.e. fixed or adjusted). We asked users to select an image that is more visually pleasing and give reasons for their choice. Users were also given a "*There is no difference.*" option. We randomized the order in which we place images side by side.

We collected more than 25 user inputs for each comparison, and averaged user responses for each degradation scenario (Fig. 4.13). When the degradation level is low (5% noise or a small blur with 2% noise), users did not prefer a particular algorithm. In such cases, the observation term is strong enough to reconstruct visually pleasing images regardless of the prior and/or the reconstruction algorithm. When the degradation level is high, however, many users clearly favored our results. User comments pointed out that realistic textures in trees, grass, and even in seemingly flat regions such as gravel paths are important for visual realism. Users who favored

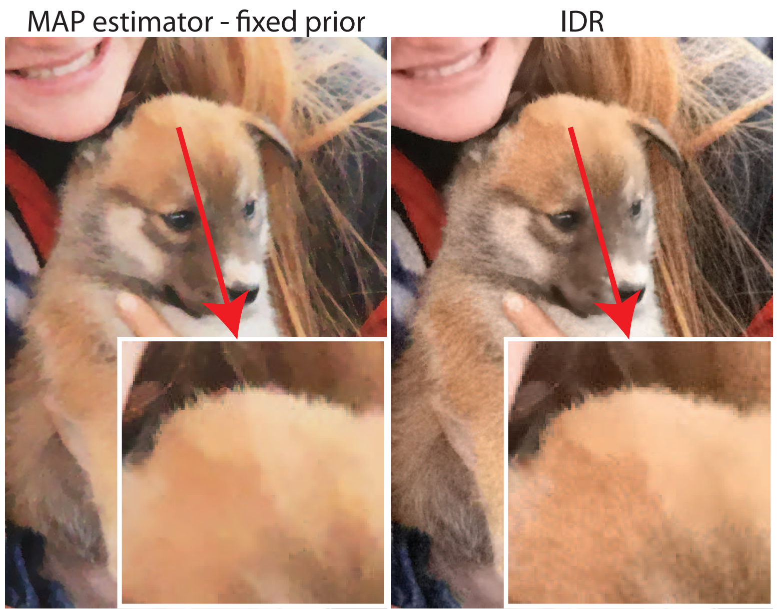


Figure 4.12: *We could observe an artificial boundary when the estimated prior is different in adjacent segments that have similar textures. While this rarely occurs, we could remove such artifacts using a texture segmentation algorithm instead of a color-based segmentation algorithm.*

MAP estimates preferred clean renditions of flat regions and were not disturbed by piecewise smooth textures (some even found it artistic.) Individual users consistently favored either our result or MAP estimates, suggesting that image evaluation is subjective in nature.

■ 4.6 Conclusion

We have developed an iterative deconvolution algorithm that matches the gradient distribution. Our algorithm bridges the energy minimization methods for deconvolution and texture synthesis. We show through a user study that matching derivative distribution improves the perceived quality of reconstructed images. The fact that a perceptually better image receives lower PSNR/SSIM suggests that there is a room for improvement in image quality assessment.

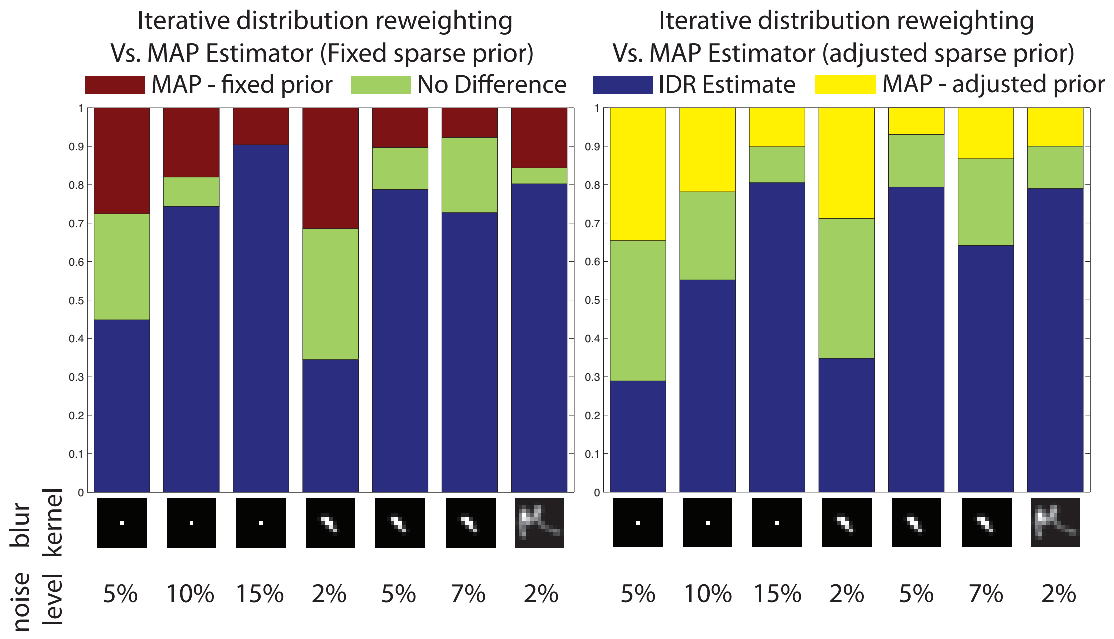


Figure 4.13: *We conducted a user study to test our impression that IDR reconstructions are visually more pleasing than MAP estimates. The blue region corresponds to the fraction of users that favored IDR over MAP estimators. When the image degradation level is small, users did not show a particular preference, but as the image degradation level increases, users favored images reconstructed using IDR.*

Chapter 5

Orthogonal parabolic exposures for motion deblurring

5.1 Introduction

IN previous chapters, we addressed spatially invariant blur due to handshake. This chapter addresses a different class of blur: a subject motion blur. Because subjects can move in different directions, images are often blurred in a spatially variant manner. Therefore, the subject motion blur removal often requires spatially variant motion estimation. Even if we could perfectly identify the motion, however, blur removal is still challenging because we lose high frequency information about the image due to blur.

This chapter provides a solution that addresses these challenges for a restricted class of subject motions. We assume that the scene consists of objects moving at a constant speed in arbitrary directions parallel to the image plane and that the camera is placed on a tripod. Spatially variant subject motion blur is therefore locally piecewise constant (in contrast to in-plane camera rotations that induce continuous spatially variant blur.)

Our solution takes two successive images using a moving sensor: one moving in a horizontal parabolic displacement path and another moving in a vertical parabolic path. From these two images, we recover one sharp image by locally estimating motions (i.e. blur kernels) (Sec. 5.4.2) and by deconvolving input images using a multi-image deconvolution algorithm (Sec. 5.4.1). We show that the kernel estimation error is negligible and that the image information loss due to motion blur is provably near minimal (Sec. 5.3).

■ 5.2 Related work

Some previous methods handle spatially variant blur by restricting the type of spatially variant blur [23, 24, 42, 47, 49, 73]: Levin [47] considers a piecewise constant spatially variant blur; Shan *et al.* [73] assume that the relative motion between the camera and the scene is purely rotational; Whyte *et al.* [88] and Gupta *et al.* [36] estimate a spatially variant blur by modeling camera shake as a rigid body motion. To aide spatially variant blur estimation, additional hardware could be used to record the relative movement between the camera and the scene during exposure, from which one can estimate the spatially variant blur [6, 79]. Levin *et al.* [49] introduce a new camera that makes the blur invariant to 1D subject motions. Users could assist spatially varying blur estimation by specifying blurred edges that should be sharp [41] or by specifying regions with different amount of blur [24]. Taking two images also helps estimate spatially variant blur [15, 76].

Most of aforementioned methods do not address information loss due to blur. Typical motion blur kernels correspond to box filters in the motion direction, therefore blurs attenuate high spatial frequencies and make the blur inversion ill-posed. One technique addressing this issue is a flutter shutter camera [63]. By opening and closing the shutter multiple times during exposure, one can significantly reduce the high frequency image information loss. Another method takes two images, each with different exposure lengths [90]. The short-exposure image contains high frequency information that supplements the missing information in the long-exposure, blurred image. Agrawal *et al.* [3] take multiple shots of a moving object, each with different exposures, and deconvolve the moving object using all the shots. The multi-shot strategy is beneficial because the information lost in one of the shots is captured by another. However, their strategy does not offer guarantees on the worst-case performance. Levin *et al.* [49] propose a parabolic motion camera to minimize the information loss for 1D constant velocity motions, but the solution is invalid if a 2D motion is present. Agrawal and Raskar [1] analyze the performance of a flutter-shutter camera and a parabolic camera and conclude that a flutter shutter camera performs better for handling a 2D constant velocity motion blur. Agrawal and Xu [2] introduce a new code for a flutter shutter camera with a better trade-off between blur estimation and information capture.

■ 5.3 Sensor motion design and analysis

Consider an object moving at a constant velocity and let $s_{x,y} = [s_x, s_y]$ be its 2D velocity vector. Suppose we capture J images $B^1, ..B^J$ of this object using J translating cameras. Locally, the blur is a convolution:

$$B^j = k^j_{s_{x,y}} \otimes I + n^j \tag{5.1}$$

where I is an ideal sharp image, n^j imaging noise, and $k^j_{s_{x,y}}$ a blur kernel (point spread function, PSF). $k^j_{s_{x,y}}$ is a function of the relative motion between the sensor and the scene. The convolution is a multiplication in the frequency domain:

$$\hat{B}^j(\omega_{x,y}) = \hat{k}^j_{s_{x,y}}(\omega_{x,y})\hat{I}(\omega_{x,y}) + \hat{n}^j(\omega_{x,y}) \tag{5.2}$$

where $\omega_{x,y} = [\omega_x, \omega_y]$ is a 2D spatial frequency, and the ˆ indicates the Fourier transform of the corresponding signal.

To deblur images successfully, we need to increase the spectral content of blur kernels $||\hat{k}^j_{s_{x,y}}(\omega_{x,y})||^2$. Qualitatively, a deblurring algorithm divides the Fourier transform $\hat{B}^j$ of the image by that of the blur kernel $\hat{k}^j_{s_{x,y}}$ at every spatial frequency. If $||\hat{k}^j_{s_{x,y}}(\omega_{x,y})||^2$ is small for all cameras, the deblurring algorithm amplifies noise and degrades the quality of restored image. We show in Sec. 5.4.1 that the reconstruction performance of the Wiener filter deconvolution method is inversely related to the summed spectra:

$$||\tilde{\hat{k}}_{s_{x,y}}(\omega_{x,y})||^2 = \sum_j ||\hat{k}^j_{s_{x,y}}(\omega_{x,y})||^2 \tag{5.3}$$

Therefore, we should maximize the joint spectrum $||\tilde{\hat{k}}_{s_{x,y}}(\omega_{x,y})||^2$ of an imaging device for every $\omega_{x,y}$ and for every $s_{x,y}$. This goal is formally stated as follows:

Given a time budget T, find a set of J camera motions that maximizes the minimum of the summed power spectrum $||\tilde{\hat{k}}_{s_{x,y}}(\omega_{x,y})||^2$ over every spatial frequency $\omega_{x,y}$ and every motion vector $||s_{x,y}|| < S_{obj}$.

We introduce the *first* solution that provides the worst-case spectral power guarantee for 2D constant velocity motions. To prove our claim, we start with a brief review of space time motion blur analysis. We show that a set of PSFs for all 2D constant velocity motions $||s_{x,y}|| < S_{obj}$

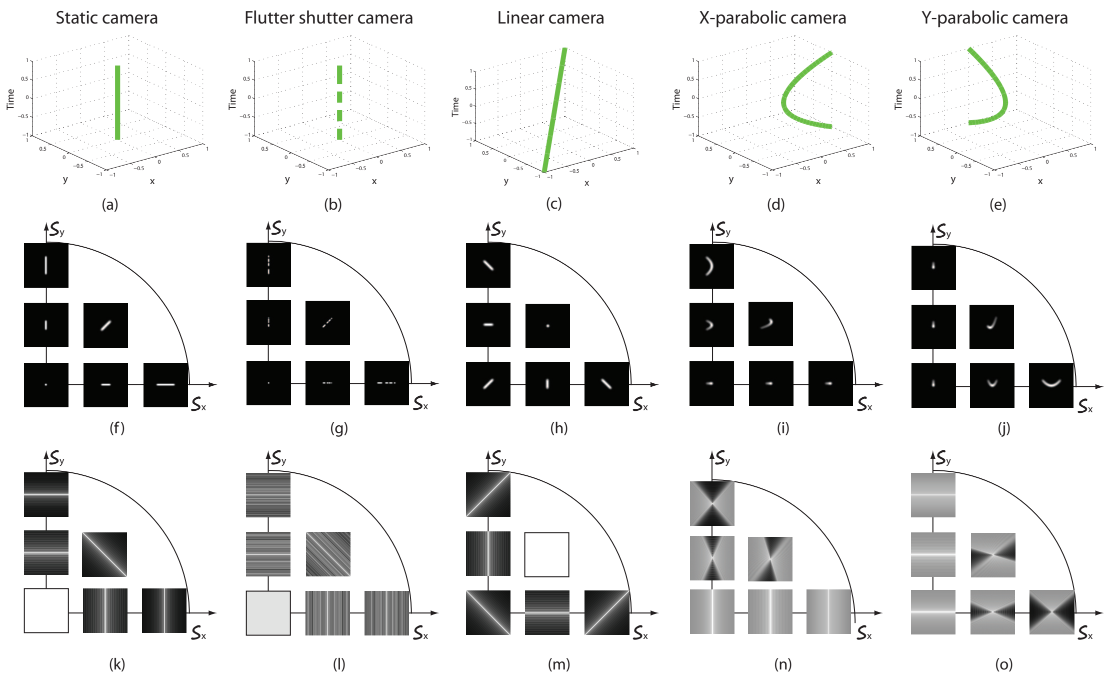


Figure 5.1: *The integration curves ϕ (a-e), the point spread functions $k_{s_{x,y}}$ (f-j) and their log-power spectra (k-o) for a few cameras. In (f-o), the outer axes correspond to x,y directional speed. In (f-j), the inner axes correspond to x–y coordinates, and in the spectra plots (k-o), the inner axes correspond to ω_x–ω_y coordinates. All spectra plots are normalized to the same scale.*

occupies the complementary volume of an inverted double cone in the Fourier domain, and that the camera motion design can be formulated as maximizing the spectral content in this volume. We show analytically that the best worst-case spectral coverage of any camera motions is bounded and that our design approaches the bound up to a constant multiplicative factor.

■ 5.3.1 Motion blur in the space-time volume

We represent light received by the sensor as a 3D space-time volume $L(x,y,t)$. That is, $L(x,y,t)$ denotes the light ray hitting the x,y coordinate of a static detector at a time instance t. A static camera forms an image by integrating these light rays over a finite exposure time T:

$$B(x,y) = \int_{-\frac{T}{2}}^{\frac{T}{2}} L(x,y,t)dt \tag{5.4}$$

Assume the camera is translating during exposure on the *x*–*y* plane, and let **f** be its displacement path:

$$\{\mathbf{f} : [x,y,t] = [f_x(t), f_y(t), t]\} \tag{5.5}$$

Then the rays hitting the detector are spatially shifted:

$$B(x,y) = \int_{-\frac{T}{2}}^{\frac{T}{2}} L\left(x + f_x\left(t\right), y + f_y\left(t\right), t\right) dt + n \tag{5.6}$$

where n is imaging noise. For example, for a static camera the integration curve is a vertical straight line $f_x(t) = f_y(t) = 0$ (Fig. 5.1(a)). The integration curve of a camera moving at a constant velocity is a slanted line $f_x(t) = s_x t,\ f_y(t) = s_y t$ (Fig. 5.1(c)). For horizontal parabolic motion, $f_x(t) = at^2,\ f_y(t) = 0$ and for a vertical parabola $f_x(t) = 0,\ f_y(t) = at^2$ (Fig. 5.1(d-e)). We can represent the integration curve **f** as a 3D integration kernel ϕ:

$$\phi(x,y,t) = \delta(x - f_x(t)) \cdot \delta(y - f_y(t)) \tag{5.7}$$

where δ is a Dirac delta function.

If an object motion is locally constant, we can express the integrated image as a convolution of a sharp image at one time instance (e.g. $L(x,y,0)$) with a point spread function $k_{s_{x,y}}$. The PSF $k_{s_{x,y}}$ of a constant velocity motion $s_{x,y} = [s_x, s_y]$ is a sheared projection of the 3D integration kernel ϕ:

$$k_{s_{x,y}}(x,y) = \int_t \phi(x - s_x t, y - s_y t, t) dt \tag{5.8}$$

Some PSFs of different integration kernels are shown in the second row of Fig. 5.1.

The Fourier transform $\hat{k}_{s_{x,y}}$ of the PSF $k_{s_{x,y}}$ is a slice from the Fourier transform $\hat{\phi}$ of the integration kernel ϕ [49, 56]:

$$\hat{k}_{s_{x,y}}(\omega_x, \omega_y) = \hat{\phi}(\omega_x, \omega_y, s_x\omega_x + s_y\omega_y) \tag{5.9}$$

$\hat{k}_{s_{x,y}}$ for several object motions for different integration kernels ϕ are shown in the bottom row of Fig. 5.1. Fig. 5.2(a) shows Fourier slices corresponding to horizontal object motions at varying velocities, the case considered in [49]. Slices occupy a 3D double wedge. When the motion direction changes (e.g. $s_x = s_y$ in Fig. 5.2(b)), slices occupy a rotated 3D double wedge. In general, 2D Fourier slices corresponding to all motion directions $\|s_{x,y}\| < S_{obj}$ lie in the

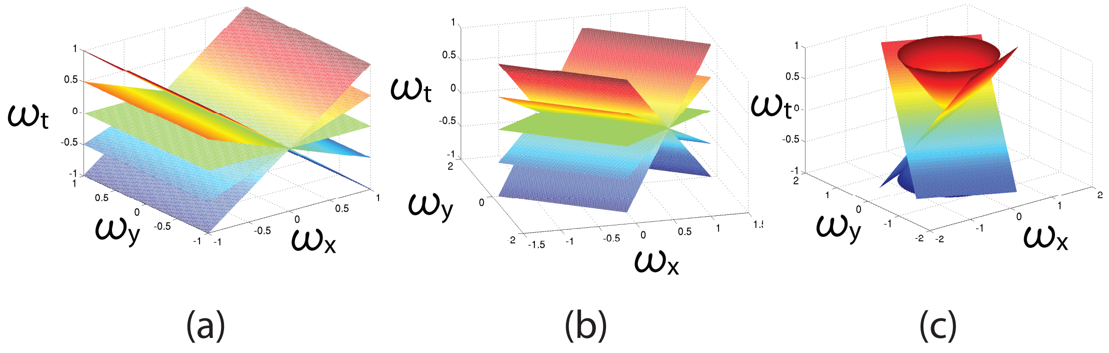


Figure 5.2: *(a) The union of horizontal motion PSF slices at all velocities $\hat{k}_s$ forms a 3D double wedge. (b) A union of diagonal motion PSF slices forms a rotated 3D double wedge. (c) The spectra of all 2D constant velocity PSF slices comprise a wedge of revolution.*

complementary volume of an inverted double cone (Fig. 5.2(c)). We refer to this volume as *a wedge of revolution*, defined as a set:

$$C \equiv \{(\omega_x, \omega_y, \omega_t) | \omega_t < S_{obj} \|\omega_{x,y}\|\} \tag{5.10}$$

To see this, note that the Fourier transform of a PSF is a slice from $\hat{\phi}$ at $\omega_t = s_x\omega_x + s_y\omega_y$, and if $\|s_{x,y}\| \le S_{obj}$, $s_x\omega_x + s_y\omega_y \le S_{obj}\|\omega_{x,y}\|$.

Bounding spectral content Suppose we capture J images of a scene and let $\|\tilde{\hat{\phi}}\|^2$ be the joint power spectrum $\|\tilde{\hat{\phi}}(\omega_x, \omega_y, \omega_t)\|^2 = \sum_j^J \|\hat{\phi}^j(\omega_x, \omega_y, \omega_t)\|^2$. As mentioned earlier, our goal is to design a set of camera motions that maximizes the joint kernel spectrum $\|\tilde{\hat{k}}_{s_{x,y}}\|^2$ (Eq. 5.3) for all object motions $\|s_{x,y}\| < S_{obj}$. Since PSFs of all bounded 2D linear motions occupy the wedge of revolution (Eq. 5.10), designing PSFs with high spectral power for all $s_{x,y} < S_{obj}$ is equivalent to maximizing the spectral content of $\|\tilde{\hat{\phi}}\|^2$ within the wedge of revolution.

We can derive an upper bound on the worst-case spectral content of any camera motions. The amount of photon energy collected by a camera within a fixed exposure time T is bounded. Therefore, by Parseval's theorem, the norm of every ω_{x_0,y_0} slice of $\tilde{\hat{\phi}}$ (i.e. $\tilde{\hat{\phi}}(\omega_{x_0}, \omega_{y_0}, \omega_t)$) is bounded [49]:

$$\int \|\tilde{\hat{\phi}}(\omega_{x_0}, \omega_{y_0}, \omega_t)\|^2 d\omega_t \le T \tag{5.11}$$

Every ω_{x_0,y_0}-slice intersects the wedge of revolution for a segment of length $2S_{obj}\|\omega_{x_0,y_0}\|$. To maximize the worst-case spectral power, the optimal camera would spread the captured energy uniformly in this intersection. Therefore, we can derive an upper bound on the worst-case spectral power by dividing the captured energy by the segment length:

$$\min_{\omega_t} \|\hat{\tilde{\phi}}(\omega_{x_0}, \omega_{y_0}, \omega_t)\|^2 \leq \frac{T}{2S_{obj}\|\omega_{x_0,y_0}\|} \,. \tag{5.12}$$

Since the PSFs spectra $\hat{k}^j_{s_{x,y}}$ are slices through $\hat{\phi}^j$, this bound also applies for the PSFs' spectral power:

$$\min_{s_{x,y}} \|\hat{\tilde{k}}_{s_{x,y}}(\omega_{x_0}, \omega_{y_0})\|^2 \leq \frac{T}{2S_{obj}\|\omega_{x_0,y_0}\|} \,. \tag{5.13}$$

The optimal bound Eq. 5.12 applies to any types of integration kernel ϕ regardless of the number of shots taken during the time budget T.

5.3.2 Orthogonal parabolic motions

We seek a motion path whose spectrum covers the wedge of revolution and approaches the bound in Eq. 5.12. We also seek to cover the spectrum with the fewest images, because as we take more images within the time budget, the delay between subsequent shots reduces the effective time budget, degrading the spectral performance.

We could compute the optimal camera motion by inverting the Fourier transform of the bound in Eq. 5.12. However, the inverse Fourier transform of this bound is not a physically valid motion of the form in Eq. 5.7. Our solution captures two images of a scene with two orthogonal parabolic motions. We show analytically that the orthogonal parabolic motions capture the wedge of revolution with the worst-case spectral power greater than $2^{-1.5}$ of the upper bound.

Camera motion Let ϕ_1, ϕ_2 be the 3D integration kernels of x and y parabolic camera motions. The kernels are defined by the integration curves f_1, f_2:

$$\begin{aligned} f_1(t) &= [a_x(t+T/4)^2, 0, t], \quad t = [-T/2...0] \\ f_2(t) &= [0, a_y(t-T/4)^2, t], \quad t = [0...T/2] \end{aligned} \tag{5.14}$$

At time t, the derivative of the x-parabolic camera motion $f_1(t)$ is $2a_x(t-T/4)$, therefore the camera essentially tracks an object moving with velocity $2a_x(t-T/4)$ along x axis. During exposure, the x-parabolic camera tracks *once* every moving object with velocity within the range $[-2a_xT/4...2a_xT/4]$. Similarly, the y-parabolic camera covers the velocity range

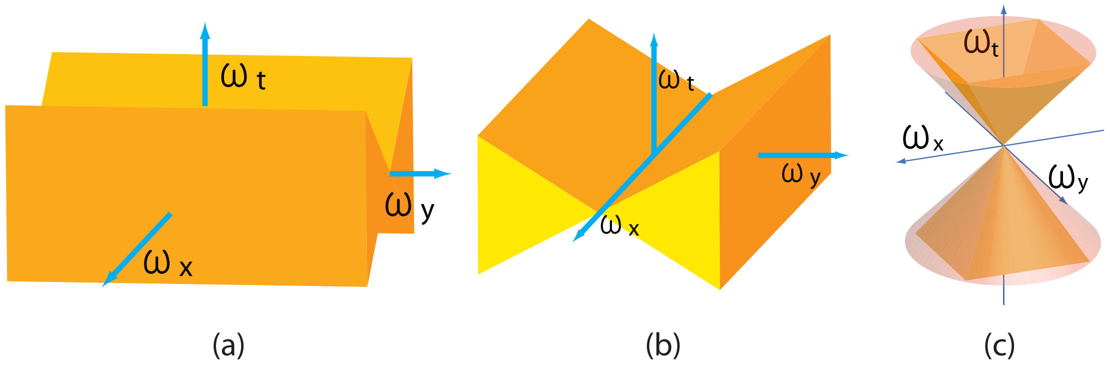


Figure 5.3: *(a) The spectrum $\hat{\phi}_1$ captured by a x-parabolic camera. (b) The spectrum $\hat{\phi}_2$ captured by a y-parabolic camera. (c) The sum of spectra captured by the two orthogonal parabolic cameras approximates the wedge of revolution.*

$[-2a_yT/4...2a_yT/4]$. For the reason that will be clarified below, we set

$$a_x = a_y = \frac{2\sqrt{2}S_{obj}}{T} \tag{5.15}$$

The maximal velocity of the sensor becomes $S_{sens} = \sqrt{2}S_{obj}$. That is, the velocity range covered by these parabolas is $[-S_{sens}...S_{sens}]$.

Fig. 5.1(i-j) show PSFs of different object motions captured by the orthogonal parabolic camera. PSFs take the form of a truncated and sheared parabola that depends on the object speed.

Optimality As mentioned earlier, to make the blur easily invertible, we want to maximize the spectral power of the camera motion paths within the wedge of revolution (Eq. 5.10). We show that the orthogonal parabolic motions capture the wedge of revolution with the worst-case spectral power greater than $2^{-1.5}$ of the optimal bound in Eq. 5.13.

We first derive the joint spectral coverage $||\tilde{\hat{\phi}}||^2$ of the two orthogonal parabolic motions. Levin *et al.* [49] show that a parabolic motion's spectrum is approximately a double wedge. Since a x-parabolic motion ϕ_1 is a Dirac delta along the y axis, the 3D kernel spectrum $||\hat{\phi}_1||^2$ spreads energy in a 3D double wedge and is constant along the ω_y axis (Fig. 5.3(a)). The y-parabolic motion spreads energy in the orthogonal 3D double wedge (Fig. 5.3(b)). Mathe-

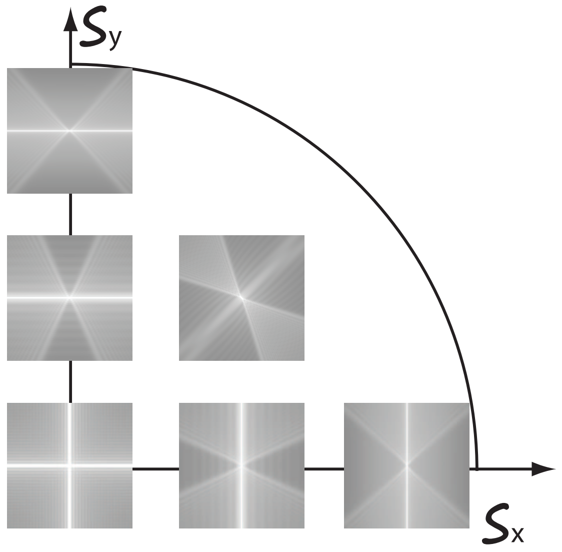


Figure 5.4: *The summed spectrum coverage of the two orthogonal parabolic motions for different object velocities $s_{x,y}$. While each parabolic motion has zeros in a range of spatial frequencies (see Fig. 5.1(n-o)), their summed spectrum does not have zeros in any spatial frequencies. The log-spectrum plots in this figure are normalized to the same scale as that of the log-spectrum plots in Fig. 5.1(k-o).*

matically speaking,

$$
\begin{aligned}
\|\hat{\phi}_1(\omega_x, \omega_y, \omega_t)\|^2 &\approx \frac{T}{4S_{sens}\|\omega_x\|} H(S_{sens}\|\omega_x\| - \|\omega_t\|) \\
\|\hat{\phi}_2(\omega_x, \omega_y, \omega_t)\|^2 &\approx \frac{T}{4S_{sens}\|\omega_y\|} H(S_{sens}\|\omega_y\| - \|\omega_t\|)
\end{aligned}
\tag{5.16}
$$

where $H(\cdot)$ is a Heaviside step function.

The 2D PSF spectra are slices from the 3D double wedge $\|\hat{\phi}_j\|^2$. Fig. 5.1 (n-o) show the log-spectrum of PSFs $\hat{k}_s^j$ for parabolic exposures as we sweep the object velocity. For x-directional motions ($s_y = 0$), the x-parabolic camera covers all spatial frequencies without zeros. This agrees with the 1D optimality argument in Levin *et al.* [49]. However, as y-directional motion increases, the x-parabolic camera fails to capture a double wedge of frequencies near the ω_y axis. In other words, the x-parabolic camera misses spectral contents in the presence of a y-directional motion, and the blur inversion is unstable. The y-parabolic camera, however, covers the frequencies missed by the x-parabolic camera, therefore the *sum* of these two spectra (Fig. 5.4) does not have any zeros in any spatial frequencies. Therefore, by taking two orthogonal parabolic exposures, we can reliably invert the blur for all 2D object motions.

Fig. 5.3(c) visualizes the joint spectrum covered by the orthogonal parabolic motions, suggesting that the sum of orthogonal 3D wedges is an approximation of the wedge of revolution. In fact, the sum of double wedges subsumes the wedge of revolution if the maximal sensor speed S_{sens} is set to $\sqrt{2}S_{obj}$.

Claim 3. *Let S_{sens} be the maximum sensor speed of the parabolic camera, and S_{obj} be the maximum object speed. If $S_{sens} \geq \sqrt{2}S_{obj}$, the joint power spectrum $\|\tilde{\hat{\phi}}\|^2$ of an orthogonal parabolic camera subsumes the wedge of revolution. When $S_{sens} = \sqrt{2}S_{obj}$, the worst-case spectral power of the orthogonal parabolic camera, at any frequency, is at least $\frac{1}{2\sqrt{2}}$ of the optimal bound.*

Proof. The power spectrum of each parabolic camera is given in Eq. 5.16. The joint power spectrum of the orthogonal parabolic camera is non-zero in the set $\{(\omega_x, \omega_y, \omega_t) | \omega_t \leq S_{sens} \max(\|\omega_x\|, \|\omega_y\|)\}$. If $(\omega_x, \omega_y, \omega_t)$ lies in the wedge of revolution, then $\omega_t \leq S_{obj}\|\omega_{x,y}\|$. Since $\|\omega_{x,y}\|^2 \leq 2\max(\|\omega_x\|^2, \|\omega_y\|^2)$,

$$\begin{aligned} \omega_t &\leq S_{obj}\|\omega_{x,y}\| \\ &\leq \sqrt{2}S_{obj} \max(\|\omega_x\|, \|\omega_y\|) \\ &\leq S_{sens} \max(\|\omega_x\|, \|\omega_y\|) \end{aligned} \tag{5.17}$$

In other words, the joint spectrum of the orthogonal parabolic cameras subsumes the wedge of revolution. This is illustrated in Fig. 5.5.

The spectral power of the joint spectrum at $(\omega_x, \omega_y, \omega_t)$ is at least the minimum of the 3D wedge spectra:

$$\min\left(\frac{T}{4S_{sens}\|\omega_x\|} + \frac{T}{4S_{sens}\|\omega_y\|}\right) \tag{5.18}$$

Since $\|\omega_{x,y}\| \geq max(\|\omega_x\|, \|\omega_y\|)$,

$$\begin{aligned} \min\left(\frac{T}{4S_{sens}\|\omega_x\|} + \frac{T}{4S_{sens}\|\omega_y\|}\right) &\geq \frac{T}{4S_{sens}\|\omega_{x,y}\|} \\ &= \frac{T}{4\sqrt{2}S_{obj}\|\omega_{x,y}\|} \end{aligned} \tag{5.19}$$

Therefore, the worst-case spectral power of the orthogonal parabolic camera is at least $2^{-1.5}$ of

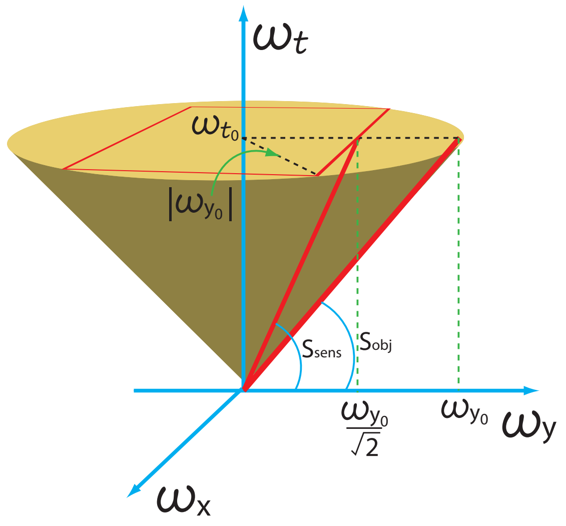


Figure 5.5: *The joint spectrum of orthogonal parabolic motions subsumes the wedge of revolution if $S_{sens} \geq \sqrt{2}S_{obj}$.*

the upper bound. □

Fig. 5.4 shows the log spectrum of the orthogonal parabolic cameras. Zeros present in one camera are compensated by its orthogonal counterpart for all object motions. At each velocity $s_{x,y}$, information for some spatial frequencies is better preserved than others, but Claim 3 guarantees that at each frequency, the spectral content is at least $2^{-1.5}$ of the optimal bound.

Discussions The orthogonal parabolic camera deblurs diagonally moving objects better than objects moving along the camera motion axis because x-parabolic and y-parabolic shots both capture information from diagonally moving objects. Note that if we know before the image capture that object motions are primarily x-directional, one could increase the exposure length of the x parabolic shot to improve the deblurring performance in expectation.

The spectral bound in Eq. 5.19 assumes that the image information at each spatial frequency is independent. Therefore, our bound holds only if the restoration method treats each spatial frequency as independent. One such restoration method is the Wiener filter (introduced in Eq. 5.26) that imposes a Gaussian prior on image gradients. In a strict sense, the use of a non-linear image reconstruction algorithm would require a different analysis method, which takes into account correlations between different spatial frequencies. However, our framework still provides a concrete construction for comparing different camera designs, which we present below.

5.3.3 Discussion of other cameras

We compare the performance of the orthogonal parabolic camera to those of other designs available in literature.

A static camera The integration kernel of a static camera is $\phi^{static}(t) = [0,0,t]$, for $t \in [-T/2...T/2]$ (Fig. 5.1(a)). Since the integration curve does not vary along the x or y axis, the power spectrum is constant along ω_x and ω_y:

$$||\hat{\phi}^{static}(\omega_x, \omega_y, \omega_t)||^2 = T^2 sinc^2(\omega_t T) \tag{5.20}$$

The Fourier transform of the PSF is a slice of the 3D spectrum $\hat{\phi}$, and is a sinc whose width depends on the object velocity $||\hat{k}^{static}_{s_{x,y}}||^2 = T^2 sinc^2((s_x\omega_x + s_y\omega_y)T)$. For fast object motions this sinc highly attenuates high frequencies. In fact, if the object motion is fast it is better to reduce the exposure time (this increases the width of the sinc) despite reducing the total amount of energy collected.

A flutter shutter camera In a flutter shutter camera [63], the integration curve of a static camera is temporally modulated (Fig. 5.1(b)). Therefore, the spectrum of the integration curve $\phi^{flutter}$ is constant along ω_x, ω_y and is modulated along ω_t:

$$||\hat{\phi}^{flutter}(\omega_x, \omega_y, \omega_t)||^2 = ||\hat{m}(\omega_t)||^2 \tag{5.21}$$

where $\hat{m}$ is the Fourier transform of the shutter code. This code can be designed to be more broadband than the sinc function in a static camera. However, the spectrum is constant along ω_x, ω_y. Therefore, the worst-case spectral performance is bounded as follows:

$$\min_s ||\hat{k}^{flutter}_s(\omega_x, \omega_y)||^2 = T/(2S_{obj}\Omega) \tag{5.22}$$

for all (ω_x, ω_y) [49], where Ω is the spatial bandwidth of the camera. As a result, the flutter shutter poorly captures the low frequency image contents.

A linearly moving camera If the camera is moving at a constant velocity (Fig. 5.1 third column), the integration curve is a slanted straight line $\phi^{linear}(t) = [s_x t, s_y t, t]$ (Fig. 5.1(c)). By linearly moving the camera, we can track the object that moves at the camera's speed, but we still suffer from a sinc fall-off for objects whose velocities are different from the camera's velocity.

Two shots Taking two images with a static camera, a linearly moving camera, or a flutter shutter camera can simplify the kernel estimation task, but it does not substantially enhance the spectral coverage of these cameras. Optimizing the exposure lengths of each shot [3], and in the case of a flutter shutter camera also optimizing the random codes in each shot, do not eliminate their fundamental limitations: their power spectra are constant along $\omega_{x,y}$ and hence they spend the energy budget outside the wedge of revolution. Previous two-image solutions to deblurring, such as [13, 15, 64, 90], fall into the category of taking two images with a static or a linearly moving camera. These methods can correctly find the motion kernels, but the image reconstruction quality is limited since the spectrum coverage is low.

A parabolic camera with a single exposure Blur kernels from a single-exposure parabolic camera are invariant to 1D constant-velocity motions, and can be shown to approach the optimal bound for a set of 1D linear motions with bounded speed [49]. A single parabolic camera, however, is neither motion invariant nor optimal for 2D motions. When an object moves in a direction orthogonal to the camera movement axis (i.e. a x-parabolic camera imaging an object moving in the y direction), the spectral coverage along the orthogonal frequencies (i.e. ω_y) is poor. We have shown in Fig. 5.1 (n-o) several Fourier slices in which the captured spectra contain zeros.

Synthetic simulation We compare the deblurring performance of (i) a pair of static cameras, (ii) a pair of flutter shutter cameras, (iii) a single parabolic camera and (iv) an orthogonal parabolic camera through simulations (Fig. 5.6). For all cameras, we fix the total exposure time T and assume that the object motion is known. The orthogonal parabolic camera is setup to deblur objects moving at speed less than S_{obj}. To give previous solutions the favor of doubt, we optimized their parameters for *each* motion independently: for a pair of static camera, we use the optimal split of the exposure time T into two shots; for a pair of flutter shutter camera, we use the optimal split of the exposure time T *and* the optimal code combinations. In a realistic scenario we cannot optimize the split of the exposure time T nor flutter-shutter codes because the object motion is not known a priori.

We render images of a moving object seen by these cameras: zero-mean Gaussian noise with standard deviation $\eta = 0.01$ is added to the rendered blurry images. We deblur rendered images using the Wiener deconvolution and compare the reconstruction performance. Fig. 5.6 shows deconvolution results and their peak signal-to-noise ratio (PSNR). Each row corresponds to a different object velocity. When the object is static, a pair of static camera restores visually

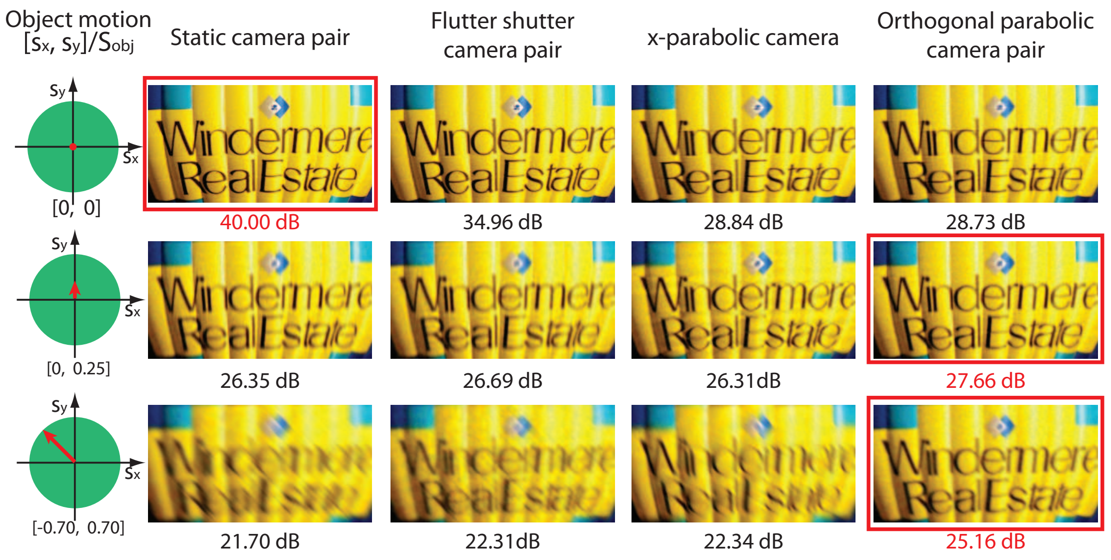


Figure 5.6: *Synthetic visualizations of the reconstruction quality. We optimized the exposure lengths of each camera. First column: The object motion during the exposure. The green disc denotes the velocity range covered by the orthogonal parabolic camera, and the red arrow denotes the object velocity. Other columns show images deconvolved using the Wiener filter (Eq. 5.26). The orthogonal parabolic camera outperforms the other optimized solutions in deblurring the moving object.*

the most pleasing deconvolution result. This is intuitively satisfying since the static camera is optimized for static object motions. The image quality from a flutter shutter camera is slightly worse than that of a static camera due to the loss of light. For moving objects, the orthogonal parabolic camera restores visually the most pleasing deconvolution results. While the orthogonal parabolic camera deblurs moving objects better than other cameras, its performance degrades as the object moves faster. However, the worst-case spectral performance across all velocities $s_{x,y}$ of interest is at least $2^{-1.5}$ of the optimal bound.

We put the synthetic experiment in the context of previous blur removal techniques. The performance of previous two-image motion deblurring techniques, such as [13, 15, 64, 90] can be approximated by the deconvolution result of the static camera pair in Fig. 5.6. Even if these solutions correctly estimate the motion, inverting the blur kernel is still hard since high frequencies are attenuated. Blind motion deblurring solutions, such as [31, 74], attempt to resolve an even harder problem, since they estimate the blur kernel from a single input image.

5.4 Image reconstruction

We review a Bayesian method for image deconvolution and kernel estimation, and extend the result to accomodate two input images. We derive a closed form solution to estimate blur kernels from input images, and present an equivalent representation to estimate motion locally. Also, we experimentally show that an image reconstruction error due to kernel misclassification is small.

5.4.1 Non-blind deconvolution

A non-blind deconvolution algorithm recovers a blur-free image I from a blurry image B^j and an estimated blur kernel k^j. Let $\overline{B}, \overline{k}$ be $\overline{B} = [B^1, B^2], \overline{k} = [k^1, k^2]$. We recover the blur-free image by maximizing the posterior probability. Using Bayes rule:

$$\begin{aligned} \tilde{I} &= \underset{I}{\operatorname{argmax}}\, p(I|\overline{B}, \overline{k}) \\ &\propto \underset{I}{\operatorname{argmax}}\, p(I, \overline{B}|\overline{k}) \\ &= \underset{I}{\operatorname{argmax}}\, p(I) \prod_{j=1}^{2} p(B^j|k^j, I) \end{aligned} \tag{5.23}$$

where we can define each term as follows:

$$\log p(B^j|k^j, I) = -\frac{1}{\eta^2}|B^j - k^j \otimes I|^2 + C_1 \tag{5.24}$$

$$\log p(I) = -\beta \sum_i \left(\rho(|g_{x,i}(I)|) + \rho(|g_{y,i}(I)|)\right) + C_2 \tag{5.25}$$

η^2 is the imaging noise variance, $\beta = 0.002$ controls the variance of the gradient profile, C_1, C_2 are constants, $g_{x,i}, g_{y,i}$ are x, y directional gradient operators at pixel i, and $\rho(z) = z^{\alpha}$ is a robust norm. When $\alpha = 2$, we impose a Gaussian prior on the image gradients, and when $\alpha \leq 1$, we impose a sparse prior.

Eq. 5.23 is essentially a joint deconvolution model, stating that we seek an image $\tilde{I}$ that fits the convolution constraints of both B^1 and B^2. In other words, the deconvolved image $\tilde{I}$ should be able to synthesize the input images B^1 and B^2 using the pair of kernels that reconstructed $\tilde{I}$. Although not presented in Bayesian terms, Rav-Acha and Peleg [64] essentially deblur two input images by maximizing the likelihood term (Eq. 5.24), and Chen *et al.* [13] augment it with the prior term Eq. 5.25. Using a sparse prior leads to visually pleasing results with crisp edges, but it is worth considering a Gaussian prior because we can derive closed form solutions.

We can efficiently solve Eq. 5.23 using the Wiener filter (i.e. a Gaussian image prior) [35]:

$$\hat{\tilde{I}}(\omega_{x,y}) = \frac{\hat{\bar{k}}^*(\omega_{x,y})\hat{\bar{B}}(\omega_{x,y})}{\frac{1}{\eta^2}\|\hat{k}(\omega_{x,y})\|^2 + \sigma^{-2}(\omega_{x,y})}. \tag{5.26}$$

where * is a complex conjugate operator, $\sigma^2(\omega_{x,y})$ is the variance of the image prior in the frequency domain: $\sigma^{-2}(\omega_{x,y}) = \beta(\|\hat{G}_x\|^2 + \|\hat{G}_y\|^2)$ where $\hat{G}_x, \hat{G}_y$ are the Fourier transform of derivative filters. We use the Wiener filter to restore images for kernel estimation, but use a sparse deconvolution to restore the final blur-free image.

We can explicitly compute the expected reconstruction error using a Gaussian image prior by taking expectation over the space of natural images and over image noise:

$$\mathbf{E}_{I,\mathbf{n}}\left[\|\tilde{I} - I\|^2\right] = \sum_{\omega_x}\sum_{\omega_y} \frac{\eta^2}{\sum_j \|\hat{k}^j_{s_{x,y}}(\omega_x,\omega_y)\|^2 + \eta^2\sigma^{-2}(\omega_x,\omega_y)} \tag{5.27}$$

Eq. 5.27 highlights that the image reconstruction error decreases monotonically as the summed power spectrum $\sum_j \|\hat{k}^j_{s_{x,y}}(\omega_x,\omega_y)\|^2$ increases. This justifies our PSF design goal in Eq. 5.3.

5.4.2 Kernel estimation

A critical step in motion deblurring is estimating the correct blur kernel $\bar{k}$. For that we seek

$$\bar{k} = \underset{k}{\operatorname{argmax}}\, p(\bar{k}|\bar{B}) = \underset{k}{\operatorname{argmax}}\, p(\bar{B}|\bar{k})p(\bar{k}) \tag{5.28}$$

where $p(\bar{k})$ is a prior on blur kernels (which we assume uniform). We derive the likelihood $p(\bar{B}|\bar{k})$ by marginalizing over all latent images I:

$$p(\bar{B}|\bar{k}) = \int p(\bar{B}, I|\bar{k})dI \tag{5.29}$$

where $p(\bar{B}, I|\bar{k})$ is defined in Eq. 5.24,5.25. If the prior $p(I)$ is Gaussian, $p(\bar{B}|\bar{k})$ is also Gaussian. If $p(I) \sim \mathbb{N}(0,\sigma^2)$, we can evaluate $p(\bar{B}|\bar{k})$ explicitly in the Fourier domain:

$$\log p(\hat{\bar{B}}|\hat{\bar{k}}) = C_3 - \frac{1}{2N}\sum_{\omega_{x,y}}^{N}\left(\frac{\|\hat{B}^1\hat{k}^{2*} - \hat{B}^{2*}\hat{k}^1\|^2 + \eta^2\sigma^{-2}\left(\|\hat{B}^1\|^2 + \|\hat{B}^2\|^2\right)}{\|\hat{B}^1\|^2 + \|\hat{B}^2\|^2 + \eta^2\sigma^{-2}}\right) \tag{5.30}$$

We have omitted the dependence on $\omega_{x,y}$ for clarity. When there is only one observed image (i.e. $\hat{k}^2 = 0, \hat{B}^2 = 0$), Eq. 5.30 reduces to a zero-frequency test which favors kernels with similar zero patterns as that of the blurry image $\hat{B}^1$ [48]. When there are two observed images, the difference term $\|\hat{B}^1\hat{k}^{2*} - \hat{B}^{2*}\hat{k}^1\|^2$ supplements the zero frequency test: this term favors a pair of kernels that satisfies the commutative property of convolution. This phase term drastically improves the reliability of the kernel estimation measure. While Rav-Acha and Peleg [64], Chen *et al.* [13] and Agrawal *et al.* [3] introduce kernel estimation methods that explicitly instantiate the commutative property of convolution, what we introduce here is a Bayesian kernel estimation method that balances the contribution of the commutative property of convolution and the image prior.

We can rewrite $\log p(\bar{\hat{B}}|\bar{\hat{k}})$ in an equivalent representation that is more attractive for computational reasons. This involves solving for the latent image $\tilde{I}$ using Eq. 5.23, and expressing $p(\bar{B}|\bar{k})$ as follows [1]:

$$\log p(\bar{B}|\bar{k}) = \log p(\tilde{I}, \bar{B}|\bar{k}) + \tilde{\Psi} + C_4 \tag{5.31}$$

where $\tilde{\Psi} = \sum_\omega \log \Psi_\omega$, and $\Psi_\omega = \frac{1}{\eta^2}\sum_j \|\hat{k}^j_\omega\|^2 + \sigma_\omega^{-2}$ is the variance of $p(\bar{\hat{B}}_\omega|\bar{\hat{k}}_\omega)$. This variance term plays a critical role in distinguishing Eq. 5.31 from a standard MAP score $p(I, \bar{k}|\bar{B})$ since Eq. 5.31 accounts for the overall probability volume around the mode and not only the mode itself [50].

Qualitatively, $\log p(\tilde{I}, \bar{B}|\bar{k})$ penalizes kernel pairs $\bar{k}$ that restore an image $\tilde{I}$ which would not fit the convolution constraints in Eq. 5.24. To satisfy the convolution constraints, the kernel pair $\bar{k}$ should "undo" the blur present in input images $\bar{B}$ *and* respect the spatial shift between input images (i.e. satisfy the commutative property of convolution.) Fig. 5.7 shows a synthetic example. We blur a sharp image with a pair of blur kernels, and deconvolve the blurred images using the correct/incorrect kernel pair. When we use an incorrect kernel pair, we observe ghosting artifacts due to an incompatible spatial shift. Therefore, this image is unable to regenerate the input images, and $\log p(\tilde{I}, \bar{B}|\bar{k})$ penalizes that. On the other hand, ghosting artifacts are not visible when we use the correct kernel to deblur the input images.

Most PSF estimation algorithms [31, 74] are designed to estimate blur kernels that are uniform across the image, and are not well suited to subject motion blur because these algorithms search over the full space of possible motions. In our scenario, object motions are assumed to

[1] This is a Laplace approximation of $\log p(\bar{\hat{B}}|\bar{\hat{k}})$, which is equivalent to $\log p(\bar{\hat{B}}|\bar{\hat{k}})$ since $\log p(\bar{\hat{B}}|\bar{\hat{k}})$ is a Gaussian distribution.

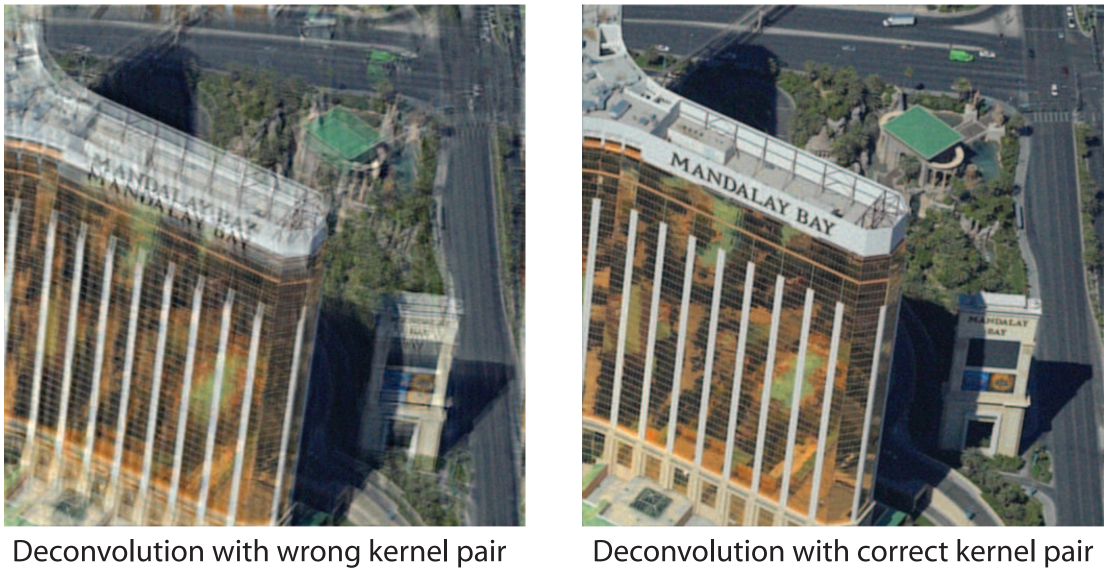

Deconvolution with wrong kernel pair
Deconvolution with correct kernel pair

Figure 5.7: *An image is synthetically blurred, and is deconvolved using the correct blur kernel pair and an incorrect blur kernel pair. An incorrect blur kernel pair has a spatial shift incompatible with input images, leading to ghosting and ringing in the restored image, whereas the correct kernel pair restores a sharp, artifact free image.*

be constant velocity. Since constant velocity motions comprise only a small subset of general motions, we can constrain the motion search space (i.e. the blur kernel search space) to constant velocity motions. This subsequently reduces the kernel estimation error. In this work, we estimate $\bar{k}$ by evaluating the log likelihood Eq. 5.31 on a set of PSF pairs that correspond to discretized 2D constant velocity motions, and by choosing the pair with the highest log likelihood.

5.4.3 Local kernel estimation

If there are multiple motions in the scene, we need to locally estimate the motion. Let $\tilde{I}_s$ be images generated by deconvolving $\bar{B}$ with the blur kernel pair $\bar{k}_s$, and let $\tilde{B}_s^j = k_s^j \otimes \tilde{I}_s$ be a re-convolved image. The log-likelihood $log(p(\bar{B}|\bar{k}_s))$ at pixel i is:

$$\begin{aligned} \log p(\bar{B}(i)|\bar{k}_s) \approx & -\frac{1}{2\eta^2}\sum_{j=1}^{2}|B^j(i) - \tilde{B}_s^j(i)|^2 \\ & - \rho(g_{x,i}(\tilde{I}_s)) - \rho(g_{y,i}(\tilde{I}_s)) + \frac{1}{N}\tilde{\Psi} \end{aligned} \tag{5.32}$$

```
% Variable definitions
$\bar{B} \equiv$ Blurred input images.
$S_k \equiv$ A set of blur kernel candidates.
$i \equiv$ A pixel index.
$f(\bar{B}, \bar{k}) \equiv$ Eq. 5.32

$C \Leftarrow 0.15$ % penalty for single-image explanations
$\bar{B}_i \Leftarrow 15 \times 15$ window around the pixel $i$ in $\bar{B}$.
% Compute the log-likelihood
for every kernel candidate $\bar{k} \in S_k$ do
    if $k^2 = 0$ then
        cost($\bar{k}$) $\Leftarrow f(\bar{B}_i, \bar{k}) + C$ % A single-image explanation
    else
        cost($\bar{k}$) $\Leftarrow f(\bar{B}_i, \bar{k})$
    end if
end for
% The kernel estimate maximizes the log-likelihood
Kernel estimate at i $\Leftarrow \arg\min$ cost($\bar{k}$)
```

Algorithm 5: `Blur kernel estimation at pixel` i

where $N = 15 \times 15$ is the size of the local window centered around the pixel i.

Handling motion boundaries Because we take two images sequentially, there are motion boundaries that are visible in one image but not in the other. In such regions the observation model (Eq. 5.24) is inconsistent and the joint deconvolution leads to visual artifacts. Therefore, we use an image deblurred using only one of the two input images to fill in motion boundaries. We can automatically detect where to use a single-image explanation by also considering kernel candidates that consist of a single image observation (i.e. $B^2 = 0, k^2 = 0$). We add an additional fixed penalty C (set to 0.15 for all experiments, determined through cross validation) to those kernel candidates; otherwise, the log-likelihood (Eq. 5.32) always favors a single image solution. Algorithm 5 provides a pseudocode for blur kernel estimation at pixel i.

Multi-scale blur kernel estimation The quality of restored images depends on how finely we sample the space of 2D constant velocity motions. With our current camera setup, we discretize the space into 4500 samples. We quantize the space such that a step in the velocity space results in roughly a one-pixel blur at the maximum object velocity. Searching over 4500 velocity samples to find the correct kernel pair at the full image resolution is computationally expensive. We resort to a coarse-to-fine strategy to mitigate the computational burden. We

```
% Variable definitions
$k_j \equiv$ the kernel estimate at $j^{th}$ scale. $j$ indexes the scale 1 to 3, from coarse to fine.
$\overline{B}^j \equiv$ Input image pyramids at $j^{th}$ scale.
$S_k^1 \equiv 2 \times 4500/4^2$ kernel candidates at the coarsest scale.

Generate a 3-level image pyramid down-sampled in octaves.
% Estimate the blur at the coarsest scale
for every pixel $i$ do
    $k_1(i) \Leftarrow$ EstimateBlurPixel($\overline{B}^1, S_k^1, i$)
end for
% Regularize the estimate using MRF
$k_1 \Leftarrow$ MRFRegularize($k_1$)
% Loop over scales
for $j = 2:3$ do
    for every pixel $i$ do
        % Velocity candidate reduction
        $S_k^j(i) \Leftarrow$ 9 velocity neighbors of $k_{j-1}(i)$
        $k_j(i) \Leftarrow$ EstimateBlurPixel($\overline{B}^j, S_k^j(i), i$)
    end for
    $k_j \Leftarrow$ MRFRegularize($k_j$)
end for
```

Algorithm 6: `Multi-scale blur estimation`

first down-sample the input images $\overline{B}$ by a factor of 4 in both width and height to reduce the number of pixels and also the motion search space: blur kernels from two adjacent velocity samples are essentially identical at a coarser resolution. At the coarsest scale, we search through $2 \times 4500/(4^2)$ velocity samples (single-image explanations incur the factor of 2). We then propagate the estimated motion to a finer resolution to refine the estimates.

At each spatial scale, we regularize the log-likelihood in Eq. 5.32 using a Markov random field (MRF). Algorithm 6 provides a pseudocode for our multi-scale kernel estimation strategy. We use the regularized kernel map to reconstruct the blur free image $\tilde{I}$. First, we deconvolve input images $\overline{B}$ using all blur kernels $\overline{k}_{s_{x,y}}$ that appear in the estimated kernel map. Given the set of deconvolved images $\tilde{I}_{s_{x,y}}$, we reconstruct the blur free image from $\tilde{I}_{s_{x,y}}$ by selecting, at each pixel, the pixel value from the image deblurred using the estimated blur kernel. We blend different motion layers using a Poisson blending method [61] to reduce artifacts at abutting motion layers.

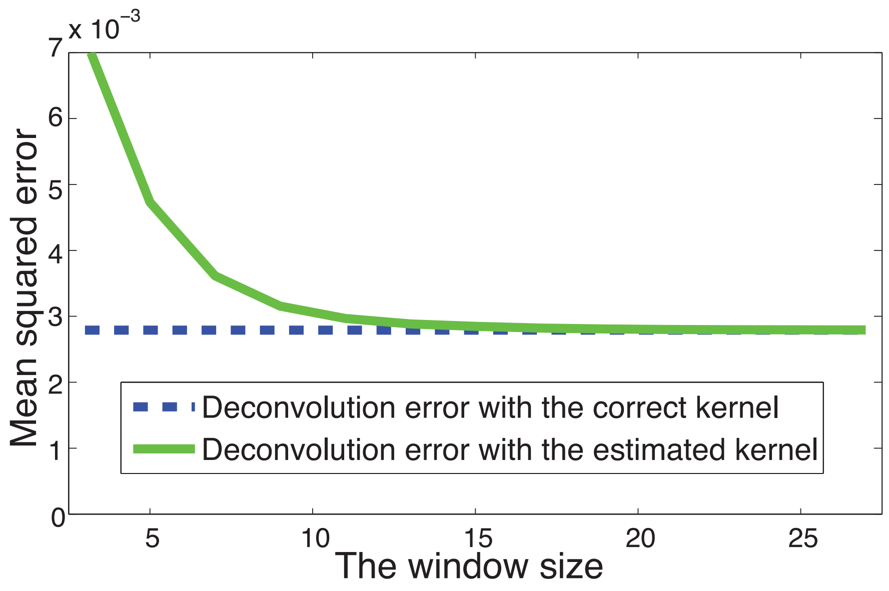
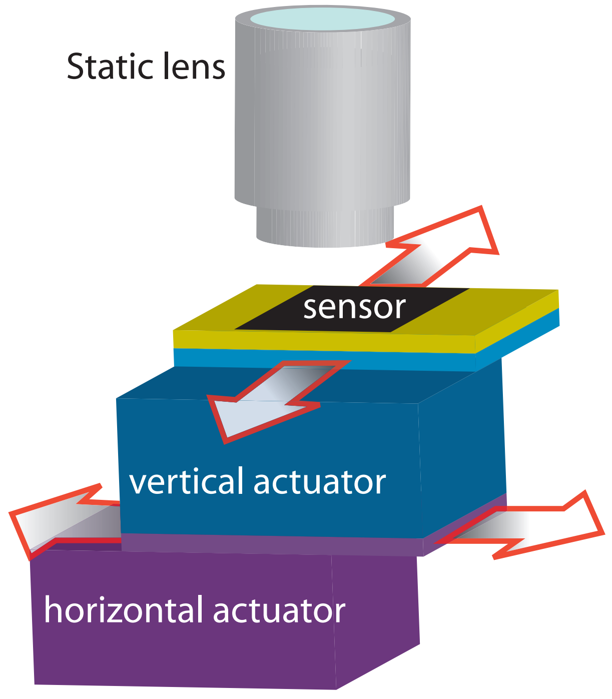


Figure 5.8: *This figure evaluates the amount of deconvolution error contributed by the local kernel estimation algorithm. When the local window is larger than* 15×15 *pixels, the deconvolution error from kernel estimation negligible.*

Quantifying the kernel estimation error Fig. 5.8 quantifies the image reconstruction error introduced by kernel estimation. We blur a sharp natural image using a blur kernel pair, and we deblur the rendered images using the correct kernel pair. Then we compute the base-line mean-squared error (MSE). The mean-squared error is not zero because we lose information through the blurry observation process, thus the restored image is not exactly the same as the original image. We also deblur the rendered images using kernels locally estimated by maximizing the log-likelihood in Eq. 5.32, and compute its MSE. For this experiment, we compute the MSE as we increase the window size, shown as a green curve in Fig. 5.8. On the same plot, we show the base-line MSE (a dotted blue curve). The base-line MSE is *independent* of the kernel estimation error, therefore the difference between the green curve and the dotted blue curve is the deconvolution error from kernel misidentification. We observe that the additional error from kernel estimation is negligible when the window size is greater than 15×15. This result suggests that it is reasonable to focus on finding a camera motion that maximizes the spectral power of blur kernels.



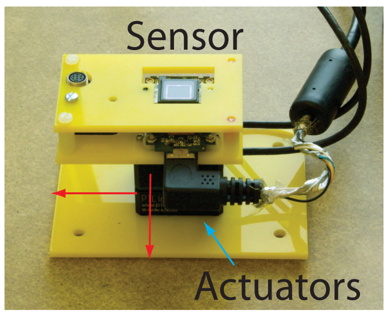


Figure 5.9: *(a) A diagram of our prototype. (b) A photograph of the actuators and the camera sensor.*

5.5 Experiments

5.5.1 Prototype camera

We built a prototype camera consisting of a sensor, two motion stages and their controllers. We mounted a light-weight camera sensor (Point Grey Research Flea 2 Camera) on two motion stages (Physik Instrumente M-663 pair), where each can move the camera sensor along orthogonal axes (See Fig. 5.9(a)). In each image capture, one of the motion stages undergoes parabolic motion, approximated by 19 segments of constant velocity due to control constraints. In practice, we could replace the motion stages with an image stabilization hardware. The camera lens is affixed to the camera lid, and does not move during exposure. The total exposure time for taking two images is 500ms: 200ms for each image, with a delay of 100ms between exposures. 100ms delay is incurred by switching the control from one motion stage to another, and can be reduced by using an improved hardware.

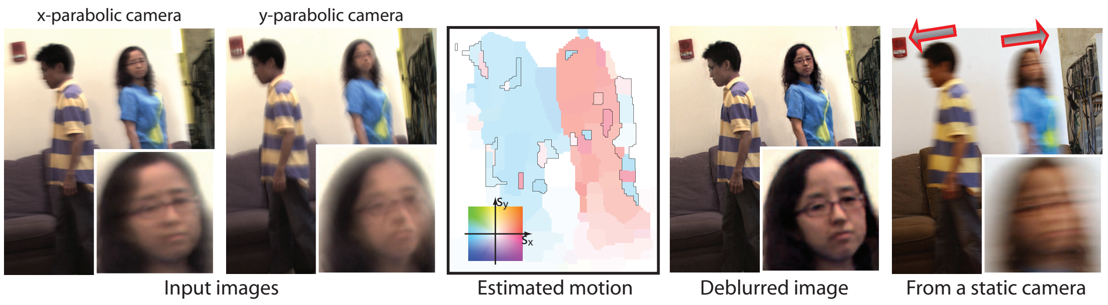


Figure 5.10: *This figure shows the pipeline of our system. We take two images using the orthogonal parabolic camera, and we locally estimate motion. The estimated motion is shown with the color coding scheme in the inset, and the detected motion boundaries are represented with black bounding boxes. We deblur the captured image pair using the estimated motion map. For reference, we also show the image taken with a synchronized static camera with a 500ms exposure.*

We rendered PSFs of our imaging system for different object speed using a parameterized actuator motion model, and used them for deconvolution. We validated the accuracy of rendered PSFs by physically calibrating blur kernels at several object velocities and by comparing them to rendered kernels. For calibration, we place a high-frequency calibration pattern on a motion rail, take a sharp image of the static calibration pattern with a static camera, and take an image of the moving pattern with a camera undergoing a parabolic motion. We solve for the kernel k that minimizes $||B - k \otimes I||^2$, where I is the sharp image of the static calibration pattern, and B is the image of the moving pattern taken with a parabolic camera.

5.5.2 Results

Fig. 5.10 illustrates the deblurring pipeline. First, we capture two images successively while the sensor undergoes parabolic motions in two orthogonal directions. From the two images, we locally estimate the motion and restore the blur-free image using blur kernels that correspond to the estimated motion. Automatically detected motion boundaries are shown by black bounding boxes. Our kernel estimation algorithm sometimes misclassifies motions in un-textured regions, but this does not lead to visual artifacts. For reference we show an image taken with a static camera with 500ms exposure, synchronized to the first shot of the orthogonal parabolic camera. This reference image reveals the object motion during exposure.

In Fig. 5.11, we compare the deconvolution performance of a two-shot static camera and an orthogonal parabolic camera. A toy train is moving at a constant velocity, assumed known

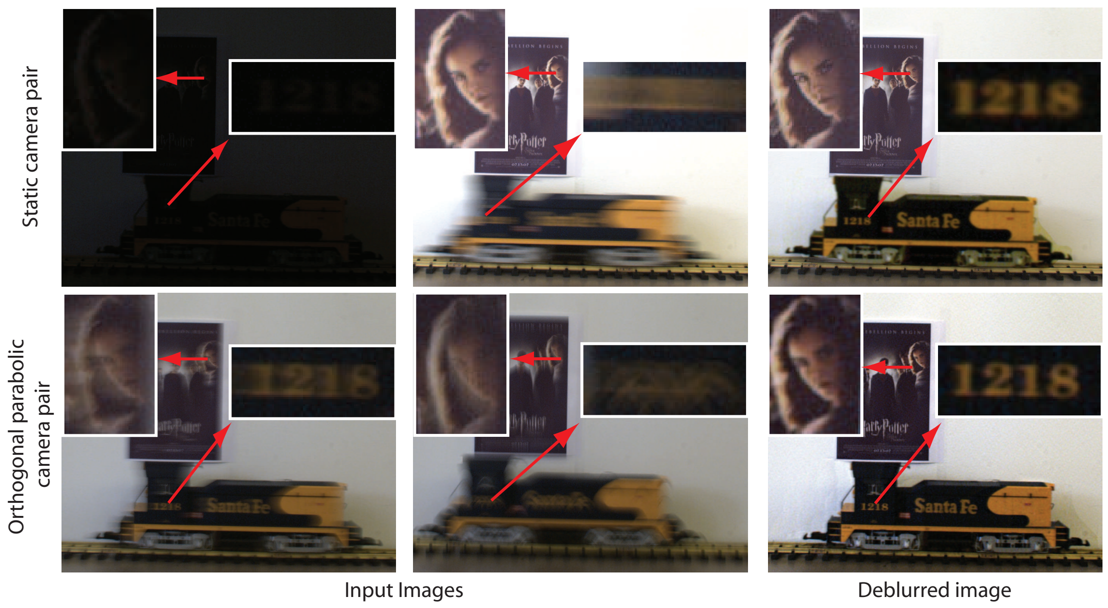


Figure 5.11: *We compare the deblurring performance of a two-shot static camera and an orthogonal parabolic camera. We optimize the split of the exposure for the static camera, assuming that we know the object motion: 40ms for the first shot and 360ms for the second shot. The blur kernel is estimated manually to compare just the amount of information captured by these cameras. The static camera reconstructs static objects well, but at the expense of a degraded rendition of the moving object, whereas the orthogonal parabolic camera restores a reasonable rendition of both the static and moving parts.*

for this comparison. For the static camera, we optimize the split of the exposure for this known train motion: 40ms for the first shot, and 360ms for the second shot. Using the static camera, we can reliably reconstruct the static part of the scene at the expense of degraded renditions of moving parts. On the other hand, our camera enables reliable reconstructions of both static and moving parts, although static regions are slightly more degraded compared to static regions restored using the static camera. An orthogonal parabolic camera spreads the energy budget over all velocities of interest, whereas a static camera concentrates the energy budget for the static motion.

We present more deblurring results on human motions in Fig. 5.12, using parabolic exposure to capture motions in non-horizontal directions [2]. Images from the static camera (500ms exposure) reveal the motions during exposure, shown by red arrows. We can observe some artifacts at motion boundaries at which the joint convolution model does not hold. In general,

[2]The camera body is tilted for this purpose.

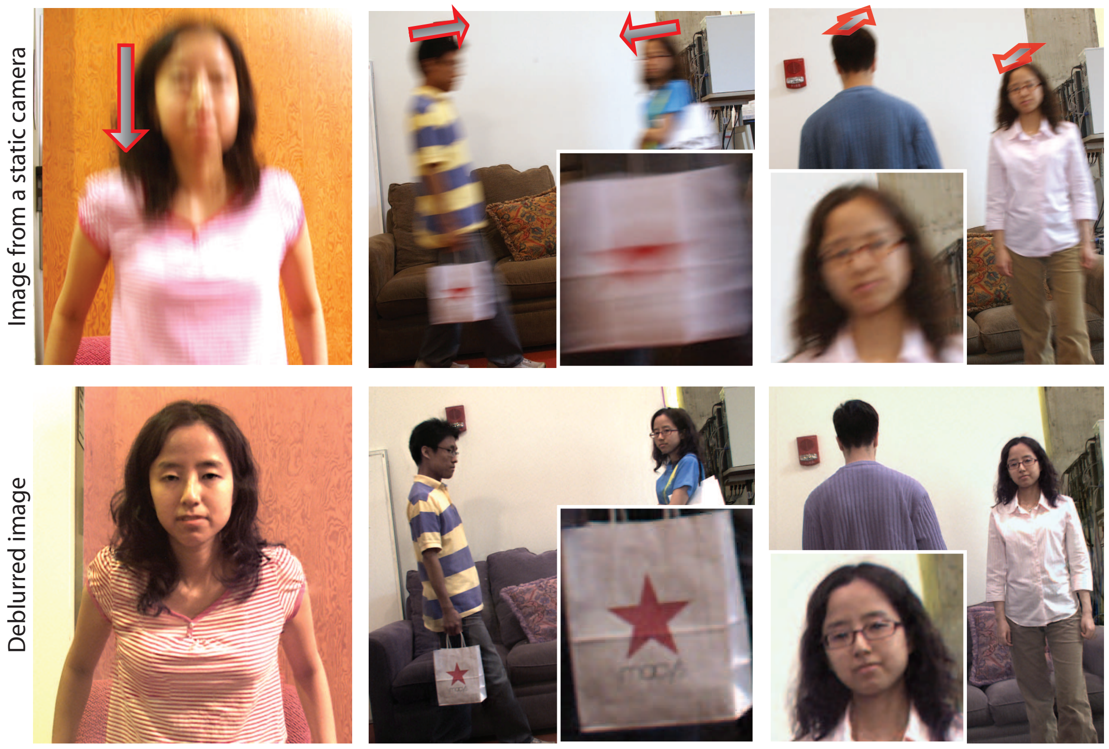


Figure 5.12: *We show the deblurring performance of the orthogonal parabolic camera. Images from a static camera with 500ms exposure are shown for reference. Arrows on reference images show the direction and magnitude of motion.*

however, the reconstructions are visually pleasing. In the third column of Fig. 5.12, we show how an orthogonal parabolic camera handles a perspective motion. While a perspective motion does not conform to our assumption on object motions, our system still recovers a reasonably sharp image.

Our image reconstruction algorithm treats an occluded region as a motion boundary. When a moving object is seen only in one of the two images due to occlusion, as in Fig. 5.13, an image deblurred using only one of the input images is used to fill in the occluded region.

5.5.3 Discussion

Kernel estimation takes 30 min - 1 hour on a single, serial machine: the running time depends on the size of the image. A by-product of kernel estimation is a blur free image deblurred using the Wiener filter. The running time of the sparse deconvolution algorithm is roughly 6 hours.

We assume that objects move at a constant velocity within the exposure time, which is

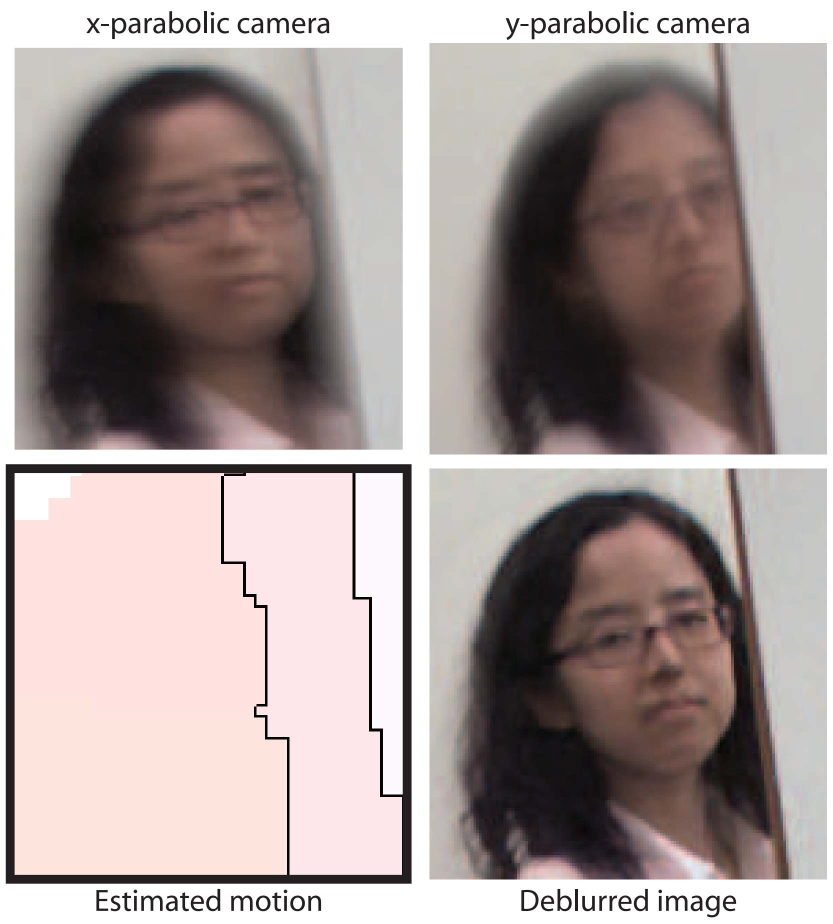


Figure 5.13: *Our image reconstruction algorithm handles occlusion boundaries in a manner similar to motion boundaries. In the occluded region, an image deblurred using only one of the two input images is used.*

a limitation shared by most previous work that deals with object motion [47, 49]. Camera shake, which typically exhibits complex kernels, needs to be handled separately. Our camera design captures image information almost optimally, but it does not provide guarantees for kernel estimation performance. While taking two images certainly helps kernel estimation, designing a sensor motion that optimizes both kernel estimation and information capture is an open problem. Our image reconstruction takes into account occlusions by allowing some pixels to be reconstructed from a single images, but a full treatment of occlusion for deconvolution remains an open challenge.

■ 5.6 Conclusion

This chapter presented a two-exposure solution to removing constant velocity object motion blur. We showed that the union of PSFs corresponding to 2D linear motions occupy a wedge of revolution in the Fourier space, and that the spectral content of the orthogonal parabolic camera approaches the optimal bound up to a multiplicative constant within the wedge of revolution.

Chapter 6

Conclusions and future work

This thesis investigated solutions to a long-standing problem in photography: motion blur removal. Motion blur removal is challenging because many blur – image pairs could have generated the blurry photograph and we need to pick the correct pair from them. The challenge is aggravated since the blur can be spatially variant depending on the relative motion between the camera and the scene.

This thesis proposed both hardware and software solutions to address a few aspects of these challenges. Chapter 2 introduced the idea of analyzing blurred edge profiles for estimating a blur kernel from a single image. We showed that (i) it is possible to estimate blur kernel projections by analyzing blurred edge profiles and that (ii) we can reconstruct a blur kernel from these blur kernel projections using the inverse Radon transform. This method is conceptually simple and computationally attractive, but is applicable only to images with many edges in different orientations. To address this concern, we also proposed an alternative technique that integrates kernel projection constraints in a MAP blur kernel estimation framework. We experimentally showed that this technique is stable even for images without many isolated edges in different orientations.

Even if we could perfectly estimate the blur kernel, blur removal is still challenging because a blur attenuates high frequency information about the original image. To hallucinate the lost information, a sparse gradient profile of natural images has been extensively exploited as an image prior. We showed in Chapter 3, however, that a sparse gradient prior is not a good model for textures. Our result showed that we should adapt the image prior to local textures. While this technique improves the quality of restored images, restored textures were not as rich as the original texture. To address this issue, we developed a new image restoration technique called iterative distribution reweighting (IDR) (in Chapter 4). IDR matches the gradient distribution of restored images to their reference distribution. We experimentally demonstrated that IDR restores visually more pleasing images compared to MAP restoration methods.

Chapter 5 addressed subject motion blur. Removing subject motion blur is challenging because the blur is spatially variant. However, we can simplify the problem by assuming that the subject is moving at a constant velocity *locally*. This assumption not only simplified the kernel estimation problem, but also allowed us to modify the imaging system to reduce the image information loss due to blur.

There are two concrete ideas to extend ideas presented in this thesis. The following sections explore these ideas in detail.

■ 6.1 Blur kernel estimation from blurred line profiles

In Chapter 2, we showed that we can recover a blur kernel by analyzing blurred edge profiles. One of the assumptions is that an image consists of isolated step edges oriented in different directions, and this assumption limits the algorithm's applicability. In fact, even piecewise smooth logos sometimes have thin bands at boundaries, making our algorithm inappropriate.

We would like to extend our method to incorporate blurred line profiles as well as blurred edge profiles. Because there are generally more lines than step edges, we foresee that using information from lines would improve the kernel estimation accuracy and stability.

There are two ways to incorporate blurred line profiles. The first method models a line profile as a box filter of unknown width. If the blur kernel was known, we can estimate the line widths; if the line widths were known, we can estimate the blur kernel from the blurred line profiles using a modified inverse Radon transform (see Eq. 6.6). Therefore, we iteratively try to estimate the line width and the blur kernel from blurred line profiles. The second method is more general: we assume that a line profile is a more general signal. We attempt to recover both the blur kernel and the *deblurred* line profiles from blurry line profiles. We explain both ideas in detail.

■ 6.1.1 Modeling lines as a box filter

A line of finite width z parallel to the y axis can be represented $L^z(x,y) = S(x) - S(x-z)$, where $S(x)$ is a step function along the x axis. $L^z(x,y)$ is equivalent to a convolution of an ideal line $\delta(x)$ with a box filter of width z. A step edge is a special case of a line where $z = \infty$.

We can show that a blurred profile of L^z is a projection of a blur kernel *blurred* by a box filter of width z. To show this, we note that a blurred edge profile is an integral of a blur kernel projection:

$$B_E(\rho_x, \rho_y) = B_E(\rho) = \int_{-\infty}^{\rho} \phi_\theta^k(\tau) d\tau \tag{6.1}$$

where $\phi_\theta^k(\rho)$ is a blur kernel projection along orientation θ and $\rho = \sqrt{\rho_x^2 + \rho_y^2}$(these variables are defined more rigorously in Chapter 2). Since L^z is a difference between edges separated by z and since these operations are linear, the blurred profile of a line of width z along orientation θ is:

$$\chi_\theta^z(\rho) = \int_{-\infty}^{\rho} (\phi_\theta^k(\tau) - \phi_\theta^k(\tau - z)) d\tau \tag{6.2}$$

We can rewrite Eq. 6.2 in a different form:

$$\chi_\theta^z(\rho) = \int_{-\infty}^{\rho} \left\{ \int_{-\infty}^{\infty} \phi_\theta^k(\zeta)(\delta(\tau - \zeta) - \delta(\tau - z - \zeta)) d\zeta \right\} d\tau \tag{6.3}$$

Now, by changing the order of integration,

$$\begin{aligned} \chi_\theta^z(\rho) &= \int_{-\infty}^{\infty} \phi_\theta^k(\zeta) \left\{ \int_{-\infty}^{\rho} (\delta(\tau - \zeta) - \delta(\tau - z - \zeta)) d\tau \right\} d\zeta \\ &= \int_{-\infty}^{\infty} \phi_\theta^k(\zeta) \{ S(\rho - \zeta) - S(\rho - z - \zeta) \} d\zeta \\ &= \phi_\theta^k(\rho) \otimes (S(\rho) - S(\rho - z)) \end{aligned} \tag{6.4}$$

Therefore, a blurred profile of L^z in orientation θ is a blur kernel projection $\phi_\theta^k(\rho)$ convolved with a box filter of width z.

This finding implies that if we know the width of the line, we can use the blurred line profiles for kernel estimation. To do so, we need to re-define the likelihood term $p(B|k)$ in the posterior probability $p(k|B) \propto p(B|k)p(k)$. In Sec. 2.2.2, we defined $p(B|k)$ as follows:

$$\begin{aligned} p(B|k) &= \prod_{i=1}^{N} p(\phi_{\theta_i}|k) \\ &\propto \prod_{i=1}^{N} \left\{ \exp\left(-\frac{\|\phi_{\theta_i} - R_{\theta_i} k\|^2}{2\eta_p^2} \right) \right\} \end{aligned} \tag{6.5}$$

where R_{θ_i} is a projection operator along θ_i. We redefine this likelihood in terms of blurred line profiles χ:

$$\begin{aligned} p(B|k) &= \prod_{i=1}^{N} p(\chi_{\theta_i}, z_i|k) \\ &= \prod_{i=1}^{N} p(\chi_{\theta_i}|k, z_i)p(z) \\ &\propto \prod_{i=1}^{N} \left\{ \exp\left(-\frac{\|\chi_{\theta_i} - W(z_i)R_{\theta_i}k\|^2}{2\eta_p^2} \right) \right\} p(z) \end{aligned} \tag{6.6}$$

where $W(z_i)$ is a convolution matrix that blurs the kernel projection by the width of the line z_i. Because z_i's are unknown, we also need to infer z_i during kernel estimation. z_i's have a structural regularity: edges from the same line are likely to have the same width. This can be represented as a smoothness prior $p(z)$. Given this model, we can either perform (i) an alternating maximization with respect to k and z or (ii) an Expectation-Maximization by treating line widths z as a hidden variable.

■ 6.1.2 Recovering both the line profiles as well as the blur kernel from the blurred line profiles

The idea in the previous section is conceptually simple, but is still restrictive since it makes a strong assumption that line profiles are box filters. Lines may have complex structures that cannot be adequately represented using a box-filter model. Therefore, as a more general method, we eliminate such assumptions on line profiles. Instead, we attempt to recover both line profiles L_i's as well as the blur kernel k from blurred line profiles χ_{θ_i}'s extracted from the blurred image. In other words, we want to recover k and L_i's that maximize the following probability:

$$\prod_{i=1}^{N} p(L_i, k|\chi_{\theta_i}) = \prod_{i=1}^{N} p(\chi_{\theta_i}|L_i, k)p(L_i)p(k) \tag{6.7}$$

We can model the likelihood term $p(\chi_{\theta_i}|L_i, k)$ as follows:

$$p(\chi_{\theta_i}|L_i, k) \propto \exp\left(-\frac{\|\chi_{\theta_i} - s_{\theta_i}(L \otimes k)\|^2}{2\eta_p^2} \right) \tag{6.8}$$

where s_{θ_i} is a slicing operator orthogonal to orientation θ_i. To learn the line prior $p(L_i)$, we can sample 1D slices of lines from natural images and investigate structural regularities. Since analyzing 1-dimensional structures should be easier than analyzing 2-dimensional images, this should be a feasible task. Given these models, we can maximize the posterior probability $\prod_{i=1}^{N} p(L_i, k|\chi_{\theta_i})$ with respect to k and L_i's to recover both the blur kernel k and the line profiles

L_i.

6.2 Blur kernel estimation using phase information

While Chapter 2 introduced a computationally attractive kernel estimation method, another way to reduce the computational complexity is to develop a closed form solution to kernel estimation. Levin *et al.* [48] derive a closed form solution for kernel estimation by maximizing the posterior probability $p(k|B)$ that assumes a Gaussian image prior $p(I) \sim \mathbb{N}(0, \sigma^2)$ and a uniform prior on blur kernels. Levin *et al.* [48, 50] show that $p(k|B) \sim \mathbb{N}(0, K\sigma^2 K^T + \eta^2 I)$, where K is a convolution matrix of k and I is an identity matrix. Therefore, we can find the blur kernel estimate by minimizing the log-posterior as follows:

$$\hat{k} = \underset{K}{\operatorname{argmin}}\, \tilde{B}^T (K\sigma^2 K^T + \eta^2 I)^{-1} \tilde{B} \tag{6.9}$$

where $\tilde{B}$ is a rasterized version of B. We can rewrite this expression in the Fourier domain:

$$\tilde{\hat{k}}(\omega_{x,y}) = \underset{\hat{k}}{\operatorname{argmin}} \frac{\|\hat{B}(\omega_{x,y})\|^2}{\|\hat{k}(\omega_{x,y})\|^2 \|\sigma(\omega_{x,y})\|^2 + \eta^2} \tag{6.10}$$

This kernel estimation strategy is simple and enables a rapid blur kernel estimation.

One drawback of Eq. 6.10 is that it ignores the phase information, and the blur kernel tends to be smooth [50]. It's well known that a Gaussian prior cannot disambiguate a signal $f(t)$ from its time-inverted version $f(-t)$. The importance of phase has been recognized early in the signal processing community [60]. Also, in Sec. 5.4.2, we have observed how phase information from two input images helps kernel estimation. We expect that integrating phase information by using a non-Gaussian image prior would improve the kernel estimation performance.

We foresee two directions to incorporate the phase information.

- **Using higher-order spectra**

  Higher-order spectra (HOS) analysis techniques analyze a signal's phase information [57]. HOS techniques have been successfully applied to various computer vision tasks: removing a gamma correction [28], removing a radial lens distortion [30], and estimating planar surface orientation [29]. HOS could provide similar benefits to blur kernel estimation.

- **Using local phase coherence**

  Wang and Simoncelli [84] present an interesting observation that a natural image has locally coherent phase across scale. Wang and Simoncelli [84] present a theoretical justification of the local phase coherence using scale-invariance of a multiscale Wavelet/steerable pyramid. We could use this information as an image prior to develop a kernel estimation algorithm. In other words, we can formulate the blur kernel estimation problem as finding a blur kernel that would maximally correlate the local phase of images across scale.

Appendix A: Algorithm details of IDR

We derive the details of the KL divergence penalty algorithm in Chapter 4.4.1. We can rewrite the image reconstruction optimization function as follows:

$$\frac{\|B-k\otimes I\|^2}{2\eta^2}+w_1\lambda_R\|\nabla I\|^{\gamma_R}+w_2\left(\lambda_R\|\nabla I\|^{\gamma_R}-\lambda_E\|\nabla I\|^{\gamma_E}\right)+\ln\left(\frac{\gamma_E {\lambda_E}^{1/\gamma_E}}{2\Gamma(1/\gamma_E)}\frac{2\Gamma(1/\gamma_R)}{\gamma_R {\lambda_R}^{1/\gamma_R}}\right) \quad (11)$$

The shape parameters of the empirical distribution q_E are functions of I, but dependences are omitted to reduce clutter.

First three terms Eq. 11 are similar in form to the ordinary MAP estimator, therefore they can be minimized using a gradient descent technique. If we can compute the derivative of $\ln\left(\frac{\gamma_E {\lambda_E}^{1/\gamma_E}}{2\Gamma(1/\gamma_E)}\frac{2\Gamma(1/\gamma_R)}{\gamma_R {\lambda_R}^{1/\gamma_R}}\right)$ with respect to I, we can minimize the entire function in Eq. 11 using a gradient descent method. We show that it indeed is the case.

Let $\tilde{I}$ be a rasterized vector of the image I. The derivative of $\ln\left(\frac{\gamma_E {\lambda_E}^{1/\gamma_E}}{2\Gamma(1/\gamma_E)}\frac{2\Gamma(1/\gamma_R)}{\gamma_R {\lambda_R}^{1/\gamma_R}}\right)$ with respect to $\tilde{I}$ takes the following form:

$$\frac{\partial}{\partial \tilde{I}}\ln\left(\frac{\gamma_E {\lambda_E}^{1/\gamma_E}}{2\Gamma(1/\gamma_E)}\frac{2\Gamma(1/\gamma_R)}{\gamma_R {\lambda_R}^{1/\gamma_R}}\right)=\alpha\frac{\partial\gamma_E}{\partial\tilde{I}}+\beta\frac{\partial\lambda_E}{\partial\tilde{I}} \quad (12)$$

where

$$\begin{aligned}\alpha &= \left(\frac{1}{\gamma_E}-\frac{\ln(\lambda_E)}{{\gamma_E}^2}+\frac{\Psi\left(\frac{1}{\gamma_E}\right)}{{\gamma_E}^2}\right)\\ \beta &= \left(\frac{1}{\gamma_E\lambda_E}\right)\end{aligned} \quad (13)$$

Ψ is a digamma function. $\frac{\partial\gamma_E}{\partial\tilde{I}}$ and $\frac{\partial\lambda_E}{\partial\tilde{I}}$ can be derived as follows:

$$\begin{aligned}\frac{\partial\gamma_E}{\partial\tilde{I}} &= \frac{{\gamma_E}^2{\lambda_E}^{(\frac{2}{\gamma_E})}\Gamma(\frac{1}{\gamma_E})}{N\Gamma(\frac{3}{\gamma_E})\left(\Psi(\frac{1}{\gamma_E})-3\Psi(\frac{3}{\gamma_E})+2\ln(\lambda_E)\right)}2G^TG\tilde{I}\\ \frac{\partial\lambda_E}{\partial\tilde{I}} &= -\frac{\Gamma(1/\gamma_E)\gamma_E{\lambda_E}^{(1+2/\gamma_E)}}{N\Gamma(3/\gamma_E)}G^TG\tilde{I}\end{aligned} \quad (14)$$

We show the proofs in following subsections. Since $\ln\left(\frac{\gamma_E {\lambda_E}^{1/\gamma_E}}{2\Gamma(1/\gamma_E)}\frac{2\Gamma(1/\gamma_R)}{\gamma_R {\lambda_R}^{1/\gamma_R}}\right)$ is differentiable, we can optimize Eq. 11 using a gradient descent technique. Furthermore, at fixed $[\gamma_E,\lambda_E]$, Eq. 12

is linear in $\tilde{I}$, suggesting that we can use an iterative reweighted least squares (IRLS) method to minimize Eq. 11.

Let $\tilde{B}$ be a rasterized vector of the observed image B, and K be the convolution matrix of the blur kernel k. We take the derivative of the optimization function Eq. 11 with respect to $\tilde{I}$:

$$\begin{aligned} &-\frac{K^T(\tilde{B}-K\tilde{I})}{\eta^2} + 2w_1\lambda_R\gamma_R G^T\|G\tilde{I}\|^{\gamma_R-1} + w_2\Big(2\lambda_R\gamma_R G^T\|G\tilde{I}\|^{\gamma_R-1} - 2\lambda_E\gamma_E G^T\|G\tilde{I}\|^{\gamma_E-1} \\ &+\big(\alpha - \lambda_E|G\tilde{I}|^{\gamma_E}\ln(|G\tilde{I}|)\big)\circ\frac{\partial\gamma_E}{\partial\tilde{I}} + \big(\beta - |G\tilde{I}|^{\gamma_E}\big)\circ\frac{\partial\lambda_E}{\partial\tilde{I}}\Big) = 0 \end{aligned} \tag{15}$$

where G is a gradient operator, and $\circ$ is a Hadamard element-wise matrix multiplication operator.

IRLS algorithm approximates the solution of a non-linear equation Eq. 15 by iteratively solving a linear equation that approximates Eq. 15. We approximate $\gamma G^T\|G\tilde{I}\|^{\gamma-1}$ as follows:

$$\gamma G^T\|G\tilde{I}\|^{\gamma-1} = \gamma G^T W G\tilde{I} \tag{16}$$

where W is a reweighting matrix. We update W iteratively such that minimizing $\gamma G^T\|G\tilde{I}\|^{\gamma-1}$ matches minimizing $\gamma G^T W G\tilde{I}$.

We handle the non-linearity due to $\lambda_E|G\tilde{I}|^{\gamma_E}\ln(|G\tilde{I}|)$ and $|G\tilde{I}|^{\gamma_E}$ by evaluating them once with the image reconstructed from the previous iteration, and *fixing these coefficients* during the actual minimization with respect to $\tilde{I}$. We iterate this process until convergence. We use a minimum residual method to solve the *linear* system in Eq. 15.

We can easily modify this algorithm to derive the IDR algorithm details.

The derivative of $\ln\left(\frac{\gamma_E{\lambda_E}^{1/\gamma_E}}{2\Gamma(1/\gamma_E)}\frac{2\Gamma(1/\gamma_R)}{\gamma_R{\lambda_R}^{1/\gamma_R}}\right)$

We note that γ_R, λ_R are constants, so we can focus on taking the derivative of γ_E, λ_E. We can rewrite $\ln\left(\frac{\gamma_E{\lambda_E}^{1/\gamma_E}}{2\Gamma(1/\gamma_E)}\right)$ as follows:

$$\ln\left(\frac{\gamma_E{\lambda_E}^{1/\gamma_E}}{2\Gamma(1/\gamma_E)}\right) = \ln(\gamma_E) + \frac{1}{\gamma_E}\ln(\lambda_E) - \ln\left(2\Gamma(\frac{1}{\gamma_E})\right) \tag{17}$$

There exists a relationship between the Gamma function Γ and the digamma function Ψ:

$$\frac{d\Gamma(z)}{dz} = \Gamma(z)\Psi(z) \tag{18}$$

We can use this relationship to show that

$$\begin{aligned}
&\frac{\partial}{\partial \tilde{I}} \ln\left(\frac{\gamma_E {\lambda_E}^{1/\gamma_E}}{2\Gamma(1/\gamma_E)} \frac{2\Gamma(1/\gamma_R)}{\gamma_R {\lambda_R}^{1/\gamma_R}} \right) \\
&= \frac{1}{\gamma_E}\frac{\partial \gamma_E}{\partial \tilde{I}} + \frac{1}{\gamma_E \lambda_E}\frac{\partial \lambda_E}{\partial \tilde{I}} - \frac{1}{{\gamma_E}^2}\ln(\lambda_E)\frac{\partial \gamma_E}{\partial \tilde{I}} + \frac{1}{{\gamma_E}^2}\Psi(\frac{1}{\gamma_E})\frac{\partial \gamma_E}{\partial \tilde{I}} \\
&= \alpha \frac{\partial \gamma_E}{\partial \tilde{I}} + \beta \frac{\partial \lambda_E}{\partial \tilde{I}}
\end{aligned} \tag{19}$$

The derivative of λ_E with respect to $\tilde{I}$

We show that

$$\frac{\partial \lambda_E}{\partial \tilde{I}} = -\frac{\Gamma(1/\gamma_E)\gamma_E {\lambda_E}^{(1+2/\gamma_E)}}{N\Gamma(3/\gamma_E)} G^T G\tilde{I} \tag{20}$$

where N is the total number of samples.

We can compute the second moment m_2 of gradient samples of $\tilde{I}$ as follows:

$$m_2 = \frac{1}{N}\tilde{I}^T G^T G\tilde{I} \tag{21}$$

where G is a gradient operator, and we assume that the mean of gradients $G\tilde{I}$ is zero.

The second moment m_2 is related to generalized Gaussian shape parameters γ_E, λ_E as follows:

$$m_2 = \frac{\Gamma(3/\gamma_E)}{{\lambda_E}^{\frac{2}{\gamma_E}}\Gamma(1/\gamma_E)} \tag{22}$$

We take the derivative of m_2 with respect to $\tilde{I}$. From Eq. 21,

$$\frac{\partial m_2}{\partial \tilde{I}} = \frac{2}{N} G^T G\tilde{I} \tag{23}$$

For tractability, we assume that γ_E is independent of $\tilde{I}$. From, Eq. 22,

$$\frac{\partial m_2}{\partial \tilde{I}} = \frac{\Gamma(3/\gamma_E)}{\Gamma(1/\gamma_E)}\frac{2}{\gamma_E}{\lambda_E}^{-\frac{2}{\gamma_E}-1}\frac{\partial \lambda_E}{\partial \tilde{I}} \tag{24}$$

From Eq. 23 and Eq. 24, we can show that

$$\frac{\partial \lambda_E}{\partial \tilde{I}} = -\frac{\Gamma(1/\gamma_E)\gamma_E {\lambda_E}^{(1+2/\gamma_E)}}{N\Gamma(3/\gamma_E)} G^T G\tilde{I} \tag{25}$$

The derivative of γ_E with respect to $\tilde{I}$

We show that

$$\frac{\partial \gamma_E}{\partial \tilde{I}} = \frac{{\gamma_E}^2 {\lambda_E}^{(\frac{2}{\gamma_E})} \Gamma(\frac{1}{\gamma_E})}{N\Gamma(\frac{3}{\gamma_E}) \left(\Psi(\frac{1}{\gamma_E}) - 3\Psi(\frac{3}{\gamma_E}) + 2\ln(\lambda_E) \right)} 2G^T G\tilde{I} \tag{26}$$

where N is the total number of samples.

Again, we use the relationship:

$$m_2 = \frac{\Gamma(3/\gamma_E)}{{\lambda_E}^{\frac{2}{\gamma_E}} \Gamma(1/\gamma_E)} \tag{27}$$

We take the derivative of m_2 with respect to $\tilde{I}$ assuming that m_2 is independent of λ_E.

$$\frac{\partial m_2}{\partial \tilde{I}} = \frac{1}{\left({\gamma_E}^{\left(\frac{2}{\gamma_E}\right)} \Gamma\left(\frac{1}{\gamma_E}\right) \right)^2} \times \left\{ \Gamma(\frac{3}{\gamma_E}) \Psi(\frac{3}{\gamma_E})(-\frac{3}{{\gamma_E}^2}) {\lambda_E}^{\frac{2}{\gamma_E}} \Gamma(\frac{1}{\gamma_E}) \; - \Gamma(\frac{3}{\gamma_E}) \left(\frac{\partial}{\partial \gamma_E} \left({\lambda_E}^{\frac{2}{\gamma_E}} \Gamma(\frac{1}{\gamma_E}) \right) \right) \right\} \tag{28}$$

We can show that

$$\left(\frac{\partial}{\partial \gamma_E} \left(\lambda^{\frac{2}{\gamma_E}} \Gamma(\frac{1}{\gamma_E}) \right) \right) = -{\lambda_E}^{\frac{2}{\gamma_E}} \Gamma(\frac{1}{\gamma_E}) \left(\frac{1}{{\gamma_E}^2} \right) \left(\Psi(\frac{1}{\gamma_E}) + 2\ln(\lambda_E) \right) \tag{29}$$

Using above relationships and the derivative of m_2 with respect to $\tilde{I}$ (Eq. 23), we can show that

$$\frac{\partial \gamma_E}{\partial \tilde{I}} = \frac{{\gamma_E}^2 {\lambda_E}^{(\frac{2}{\gamma_E})} \Gamma(\frac{1}{\gamma_E})}{N\Gamma(\frac{3}{\gamma_E}) \left(\Psi(\frac{1}{\gamma_E}) - 3\Psi(\frac{3}{\gamma_E}) + 2\ln(\lambda_E) \right)} 2G^T G\tilde{I} \tag{30}$$

Bibliography

[1] Amit Agrawal and Ramesh Raskar. Optimal single image capture for motion deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[2] Amit Agrawal and Yi Xu. Coded exposure deblurring: Optimized codes for PSF estimation and invertibility. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[3] Amit Agrawal, Yi Xu, and Ramesh Raskar. Invertible motion blur in video. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2009.

[4] Michal Aharon, Michael Elad, and Alfred Bruckstein. The K-SVD: an algorithm for designing of overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, Nov. 2006.

[5] Danny Barash. A fundamental relationship between bilateral filtering, adaptive smoothing, and the nonlinear diffusion equation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(6):844 – 847, June 2002.

[6] Moshe Ben-Ezra and Shree K. Nayar. Motion-based motion deblurring. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:689 – 698, 2004.

[7] Tom E. Bishop, Rafael Molina, and James R. Hopgood. Nonstationary blind image restoration using variational methods. In *Proceedings of the IEEE International Conference in Image Processing (ICIP)*, 2007.

[8] Michael J. Black, Guillermo Sapiro, David H. Marimont, and David Heeger. Robust anisotropic diffusion. *IEEE Transactions on Image Processing*, Mar. 1998.

[9] Jian-Feng Cai, Hui Ji, Chaoqiang Liu, and Zuowei Shen. High-quality curvelet-based motion deblurring from an image pair. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[10] Jian-Feng Cai, Hui Ji, Chaoqiang Liu, and Zuowei Shen. Blind motion deblurring from a single image using sparse approximation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[11] Emmanuel J. Candes and David L. Donoho. Curvelets - a surprisingly effective nonadaptive representation for objects with edges, 1999.

[12] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001.

[13] Jia Chen, Lu Yuan, Chi-Keung Tang, and Long Quan. Robust dual motion deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[14] Sunghyun Cho and Seungyong Lee. Fast motion deblurring. *ACM Transactions on Graphics (Proc. of SIGGRAPH ASIA)*, 28(5):article no. 145, 2009.

[15] Sunghyun Cho, Yasuyuki Matsushita, and Seungyong Lee. Removing non-uniform motion blur from images. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2007.

[16] Taeg Sang Cho, William T. Freeman, and Hensin Tsao. A reliable skin mole localization scheme. In *Proceedings of the IEEE Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA)*, 2007.

[17] Taeg Sang Cho, Moshe Butman, Shai Avidan, and William T. Freeman. The patch transform and its applications to image editing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[18] Taeg Sang Cho, Shai Avidan, and William T. Freeman. A probabilistic image jigsaw puzzle solver. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[19] Taeg Sang Cho, Shai Avidan, and William T. Freeman. The patch transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[20] Taeg Sang Cho, Neel Joshi, C. Lawrence Zitnick, Sing Bing Kang, Rick Szeliski, and William T. Freeman. A content-aware image prior. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[21] Taeg Sang Cho, Anat Levin, Frédo Durand, and William T. Freeman. Motion blur removal with orthogonal parabolic exposures. In *Proceedings of the IEEE International Conference on Computational Photography (ICCP)*, 2010.

[22] Christopher M. Christoudias, Bogdan Georgescu, and Peter Meer. Synergism in low level vision. In *IEEE International Conference on Pattern Recognition*, 2002.

[23] Shengyang Dai and Ying Wu. Motion from blur. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[24] Shengyang Dai and Ying Wu. Removing partial blur in a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[25] Stanley Roderick Deans. *The Radon Transform and some of its applications*. Krieger Publishing Company, 1992.

[26] Minh N. Do and Martin Vetterli. The contourlet transform: an efficient directional multiresolution image representation. *IEEE Transactions on Image Processing*, Dec. 2005.

[27] Michael Elad. On the origin of the bilateral filter and ways to improve it. *IEEE Transactions on Image Processing*, 11(10):1141 – 1151, Oct. 2002.

[28] Hany Farid. Blind inverse gamma correction. *IEEE Transactions on Image Processing*, 10(10):1428 – 1433, Oct. 2001.

[29] Hany Farid and Jana Kosecka. Estimating planar surface orientation using bispectral analysis. *IEEE Transactions on Image Processing*, 16(8):2154 – 2160, Aug. 2007.

[30] Hany Farid and Alin C. Popescu. Blind removal of lens distortions. *Journal of the optical society of America*, 18(9):2072 – 2078, Sept. 2001.

[31] Rob Fergus, Barun Singh, Aaron Hertzmann, Sam Roweis, and William T. Freeman. Removing camera shake from a single photograph. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2006.

[32] William T. Freeman and Edward H. Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1991.

[33] William T. Freeman, Egon C. Pasztor, and Owen T. Carmichael. Learning low-level vision. *International Journal of Computer Vision*, 40(1):25 – 47, 2000.

[34] Gonzalez and Woods. *Digital image processing*. Prentice Hall, 2008.

[35] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing (3rd Edition)*. Prentice Hall, 2007.

[36] Ankit Gupta, Neel Joshi, C. Lawrence Zitnick, Michael Cohen, and Brian Curless. Single image deblurring using motion density functions. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2010.

[37] Yoav HaCohen, Raanan Fattal, and Dani Lischinski. Image upsampling via texture hallucination. In *Proceedings of the IEEE International Conference on Computational Photography (ICCP)*, 2010.

[38] David K. Hammond and Eero P. Simoncelli. Image denoising with an orientation-adaptive Gaussian scale mixture model. In *Proceedings of the IEEE International Conference in Image Processing (ICIP)*, 2006.

[39] David J. Heeger and James R. Bergen. Pyramid-based texture analysis/synthesis. In *ACM SIGGRAPH*, 1995.

[40] Lens Group Canon Incorporated. *EF LENS WORK III, The Eyes of EOS.* Canon Inc, 1993.

[41] Jiaya Jia. Single image motion deblurring using transparency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[42] Neel Joshi, Rick Szeliski, and David J. Kriegman. PSF estimation using sharp edge prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[43] Neel Joshi, C. Lawrence Zitnick, Rick Szeliski, and David J. Kriegman. Image deblurring and denoising using color priors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[44] Johannes Kopf, Chi-Wing Fu, Daniel Cohen-Or, Oliver Deussen, Dani Lischinski, and Tien-Tsin Wong. Solid texture synthesis from 2D exemplars. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 26(3), 2007.

[45] Jefferey C. Lagarias, James A. Reeds, Margaret H. Wright, and Paul E. Wright. Convergence properties of the Nelder-Mead simplex method in low dimensions. *SIAM Journal of Optimization*, 1998.

[46] Ann B. Lee, David Mumford, and Jinggang Huang. Occlusion models for natural images: a statistical study of a scale-invariant dead leaves model. *International Journal of Computer Vision*, 41:35–59, 2001.

[47] Anat Levin. Blind motion deblurring using image statistics. In *Proceedings of Neural Information Processing Systems (NIPS)*, Dec. 2006.

[48] Anat Levin, Rob Fergus, Fredo Durand, and William T. Freeman. Image and depth from a conventional camera with a coded aperture. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2007.

[49] Anat Levin, Peter Sand, Taeg Sang Cho, Frédo Durand, and William T. Freeman. Motion-invariant photography. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2008.

[50] Anat Levin, Yair Weiss, Frédo Durand, and William T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[51] Yuanzhen Li and Edward H. Adelson. Image mapping using local and global statistics. In *SPIE EI*, 2008.

[52] Yunpeng Li, Sing Bing Kang, Neel Joshi, Steve Seitz, and Daniel Huttenlocher. Generating sharp panoramas from motion-blurred videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[53] Ce Liu, William T. Freeman, Rick Szeliski, and Sing Bing Kang. Noise estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[54] L. B. Lucy. An iterative technique for the rectification of observed distributions. *The astronomical journal*, 79:745 –754, 1974.

[55] G. Matheron. *Random Sets and Integral Geometry*. John Wiley and Sons, 1975.

[56] Ren Ng. Fourier slice photography. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2005.

[57] Chrysostomos L. Nikias and Athina P. Petropulu. *Higher-order spectra analysis - A non-linear signal processing framework*. Prentice Hall, 1993.

[58] Mila Nikolova. Model distortions in Bayesian MAP reconstruction. *Inverse Problems and Imaging*, 1(2):399–422, 2007.

[59] Ido Omer and Michael Werman. Color lines: Image specific color representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[60] Alan V. Oppenheim and Jae Soo Lim. The importance of phase in signals. *Proceedings of the IEEE*, 69(5):529 – 541, May 1981.

[61] Patrick Perez, Michel Gangnet, and Andrew Blake. Poisson image editing. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2003.

[62] Javier Portilla and Eero P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal of Computer Vision*, 40(1):49 – 71, Oct. 2000.

[63] Ramesh Raskar, Amit Agrawal, and Jack Tumblin. Coded exposure photography: Motion deblurring using fluttered shutter. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2006.

[64] Alex Rav-Acha and Shmuel Peleg. Two motion-blurred images are better than one. *Pattern Recognition Letters*, 26:311 – 317, 2005.

[65] William Hadley Richardson. Bayesian-based iterative method of image restoration. *Journal of the optical society of America*, 62:55 – 59, 1972.

[66] Stefan Roth and Michael J. Black. Fields of experts: a framework for learning image priors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2005.

[67] Stefan Roth and Michael J. Black. Steerable random fields. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2007.

[68] Daniel L. Ruderman. The statistics of natural images. *Network: Computation in Neural Systems*, 5(4):517–548, 1999.

[69] Bryan Russell, Antonio Torralba, Kevin Murphy, and William T. Freeman. LabelMe: a database and web-based tool for image annotation. *International Journal of Computer Vision*, 2008.

[70] Youcef Saad and Martin H. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM JSSC*, 1986.

[71] Suhail S. Saquib, Charles A. Bouman, and Ken Sauer. ML parameter estimation for Markov random fields with applications to Bayesian tomography. *IEEE Transactions on Image Processing*, 7(7):1029, 1998.

[72] Uwe Schmidt, Qi Gao, and Stefan Roth. A generative perspective on MRFs in low-level vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[73] Qi Shan, Wei Xiong, and Jiaya Jia. Rotational motion deblurring of a rigid object from a single image. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2007.

[74] Qi Shan, Leo Jiaya Jia, and Aseem Agarwala. High-quality motion deblurring from a single image. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2008.

[75] Eero P. Simoncelli and Edward H. Adelson. Noise removal via bayesian wavelet coring. In *Proceedings of the IEEE International Conference in Image Processing (ICIP)*, 1996.

[76] Michal Sorel and Filip Sroubek. Space-variant deblurring using one blurred and one underexposed image. In *Proceedings of the IEEE Conference on Image Processing (ICIP)*, 2009.

[77] A. Srivastava, A. B. Lee, E. P. Simoncelli, and S-C. Zhu. On advances in statistical modeling of natural images. *Journal of Math Imaging and Vision*, 18(1):17–33, 2003.

[78] Charles V. Stewart. Robust parameter estimation in computer vision. *SIAM Reviews*, 41 (3):513 – 537, Sept. 1999.

[79] Yu-Wing Tai, Hao Du, Michael S. Brown, and Stephen Lin. Image/video deblurring using a hybrid camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[80] Marshall F. Tappen, Ce Liu, Edward H. Adelson, and William T. Freeman. Learning Gaussian conditional random fields for low-level vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[81] Peter Toft. *The Radon Transform - Theory and Implementation*. PhD thesis, Technical University of Denmark, 1996.

[82] Carlo Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 1998.

[83] Martin Wainwright and Eero P. Simoncelli. Scale mixtures of Gaussians and the statistics of natural images. In *Proceedings of Neural Information Processing Systems (NIPS)*, 2000.

[84] Zhou Wang and Eero P. Simoncelli. Local phase coherence and the perception of blur. In *Advances in Neural Information Processing Systems*, 2003.

[85] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 2004.

[86] Yair Weiss and William T. Freeman. What makes a good model of natural images? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[87] Max Welling, Geoffrey Hinton, and Simon Osindero. Learning sparse topographic representations with products of student-t distributions. In *Proceedings of Neural Information Processing Systems (NIPS)*, 2002.

[88] Oliver Whyte, Josef Sivic, Andrew Zisserman, and Jean Ponce. Non-uniform deblurring for shaken images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[89] Oliver J. Woodford, Carsten Rother, and Vladimir Kolmogorov. A global perspective on MAP inference for low-level vision. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2009.

[90] Lu Yuan, Jian Sun, Long Quan, and Heung-Yeung Shum. Image deblurring with blurred/noisy image pairs. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2007.

[91] Song Chun Zhu, Yingnian Wu, and David Mumford. Filters, random fields and maximum entropy (FRAME): Towards a unified theory for texture modeling. *International Journal of Computer Vision*, 1998.