Blur Kernel Estimation using Straight Edges

Abstract

Camera shake induces unsightly blur in photographs, and is a common source of quality degradation in images. We describe a method to recover the blur kernel from a single image. We analyze edges in the blurry photograph to estimate projections *of the blur kernel. These projections are also known as the* Radon Transform. *We apply the Inverse Radon Transform to recover a kernel suitable for non-blind deconvolution. We impose sparsity priors in the Inverse Radon Transform to account for the sparse nature of blur kernels.*

Our method does not require the latent image estimation step to iteratively refine the kernel estimation, therefore the algorithm is computationally attractive. Although our approach applies only to images with a sufficient number of straight edges in many orientations, these encompass a broad range of scenes including many man-made scenes. We show through experiments that our algorithm performs favorably compared to prior art.

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.

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 [16, 6, 22, 4, 3, 13]. Taking images with modified cameras [1, 20] 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*. [8] 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 [17, 14]. Shan *et al*. [18] introduce a local prior, in addition to a sparse gradient prior of natural images, to detect and smooth surfaces. Cai *et al*. [2] 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 [9] 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*. [10] 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*. [5] extend Joshi *et al*. [10] in a multi-scale manner to estimate more general blur kernels with multiple modes. Cho *et al*. [5] reduce the computation by deblurring only edges in the gradient domain: their GPU implementation runs in near real-time. Levin *et al*. [12] compare the performance of several single-image blind deconvolution algorithms, and empirically show that the algorithm introduced by Fergus *et al*. [8] is the state-of-the-art in single-image blur kernel estimation [1].

2. Kernel estimation from edges

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

$$B = k \otimes I + n \tag{1}$$

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.

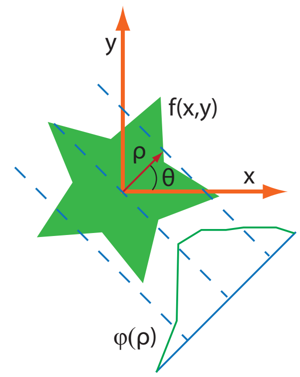

Figure 1: *The Radon transform $\phi_\theta^f(\rho)$ of a signal f (i.e. the star) is an integral of the signal along the line $\rho = x\cos(\theta) + y\sin(\theta)$ (i.e the dotted line).*

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 [7, 21]. The Radon transform of a signal $f(x, y)$ is an integral of the signal along a straight line:

$$\phi_\theta^f(\rho) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta\left(\rho - x\cos(\theta) - y\sin(\theta)\right) dxdy \tag{2}$$

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. 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 [21].

Interestingly, we can relate the Radon transform to our imaging model in Eq. (1). The imaging model in Eq. (1) can be expressed in the continuous domain:

$$B(\rho_x, \rho_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} k(x,y) I(\rho_x - x, \rho_y - y) dxdy \tag{3}$$

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

$$\begin{aligned} B_L(\rho_x, \rho_y) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} k(x,y)\delta\left(\rho - x\cos(\theta) - y\sin(\theta)\right) dxdy \\ &= \phi_\theta^k(\rho) \end{aligned} \tag{4}$$

[1] Levin *et al*. [12] do not consider Cho *et al*. [5].

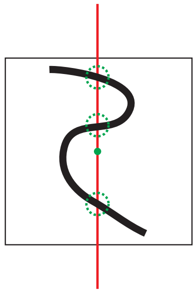

Figure 2: *The value of the convolved image at the green dot is a sum of intersections of the blur kernel (the black line) and the line (the red line). The dotted green circles indicate the intersections.*

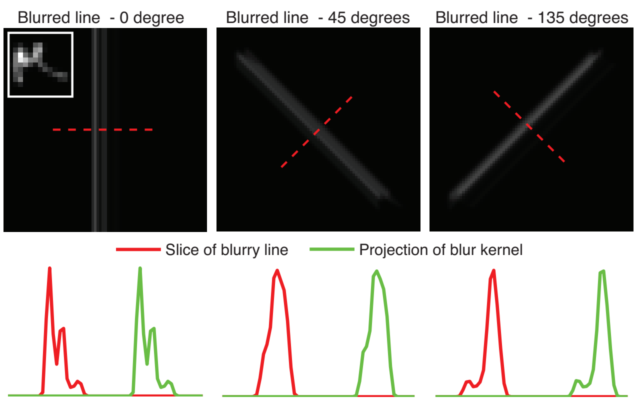

Figure 3: *We show experimentally that a slice (shown dotted-red) orthogonal to a blurred line is the same as an explicit projection of the blur kernel along the line.*

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 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 orthogonal slices of blurred lines to explicit projections of the blur kernel in those orientations. Fig. 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

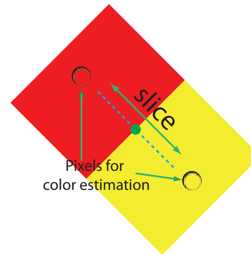

Figure 4: *To find two dominant colors on either side of an edge, we average the pixels separated from the edge by $3/4$ the assumed size of the blur kernel.*

[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:

$$e(\rho) = \int_{-\infty}^{\rho} \delta\left(\tau - x\cos(\theta) - y\sin(\theta)\right) d\tau \qquad (5)$$

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

$$\begin{aligned}
&B_E(\rho_x, \rho_y) \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} k(x,y) \int_{-\infty}^{\rho} \delta\left(\tau - x\cos(\theta) - y\sin(\theta)\right) d\tau dx dy \\
&= \int_{-\infty}^{\rho} \left\{ \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} k(x,y)\delta\left(\tau - x\cos(\theta) - y\sin(\theta)\right) dx dy \right\} d\tau \\
&= \int_{-\infty}^{\rho} \phi_\theta^k(\tau) d\tau
\end{aligned} \qquad (6)$$

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 [15] 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. 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:

$$c_i = \alpha_i W + (1 - \alpha_i) Z \qquad (7)$$

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

[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.

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 [7, 21]. 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,

$$p(k|B) \propto p(B|k)p(k) \tag{8}$$

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:

$$\begin{aligned} p(B|k) &= \prod_{i=1}^{N} p(\phi_{\theta_i}|k) \\ &\propto \exp\left(-\frac{\sum_{i=1}^{N} \|\phi_{\theta_i} - R_{\theta_i} k\|^2}{2\eta_p^2}\right) \end{aligned} \tag{9}$$

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. (6)) and α models orientation estimation error, which increases with the image noise level. The algorithm is robust to the value of α; we set $\alpha = 1$ through cross validation.

The number of edge samples (i.e. N in Eq. (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:

$$\sum_{i=1}^{N} \|\phi_{\theta_i} - R_{\theta_i} k\|^2 \approx \sum_{j=1}^{360} w_j \|\tilde{\phi}_{\theta_j} - R_{\theta_j} k\|^2 \tag{10}$$

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:

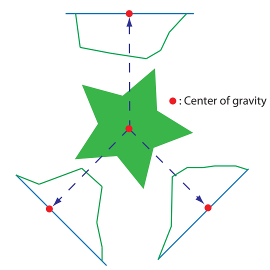

Figure 5: *The center of gravity within an object's projection is equivalent to the projection of the object's center of gravity.*

$$p(k) \propto \exp\{-(\lambda_1\|k\|^{\gamma_1} + \lambda_2\|\nabla k\|^{\gamma_2})\} \tag{11}$$

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:

$$\hat{k} = \text{argmin}_k \left\{ \frac{\sum_{j=1}^{360} w_j \|\tilde{\phi}_{\theta_j} - R_{\theta_j} k\|^2}{2\eta_p^2} + \lambda_1\|k\|^{\gamma_1} + \lambda_2\|\nabla k\|^{\gamma_2} \right\} \tag{12}$$

We use an iterative reweighted least squares method [11, 19] to minimize the energy in Eq. (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 projection is equivalent to the projection of the object's center of gravity, as shown in Fig. 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. 6. We blur this test pattern using a blur kernel shown at the top of Fig. 6, and add 0.5% Gaussian noise to the blurred pattern.

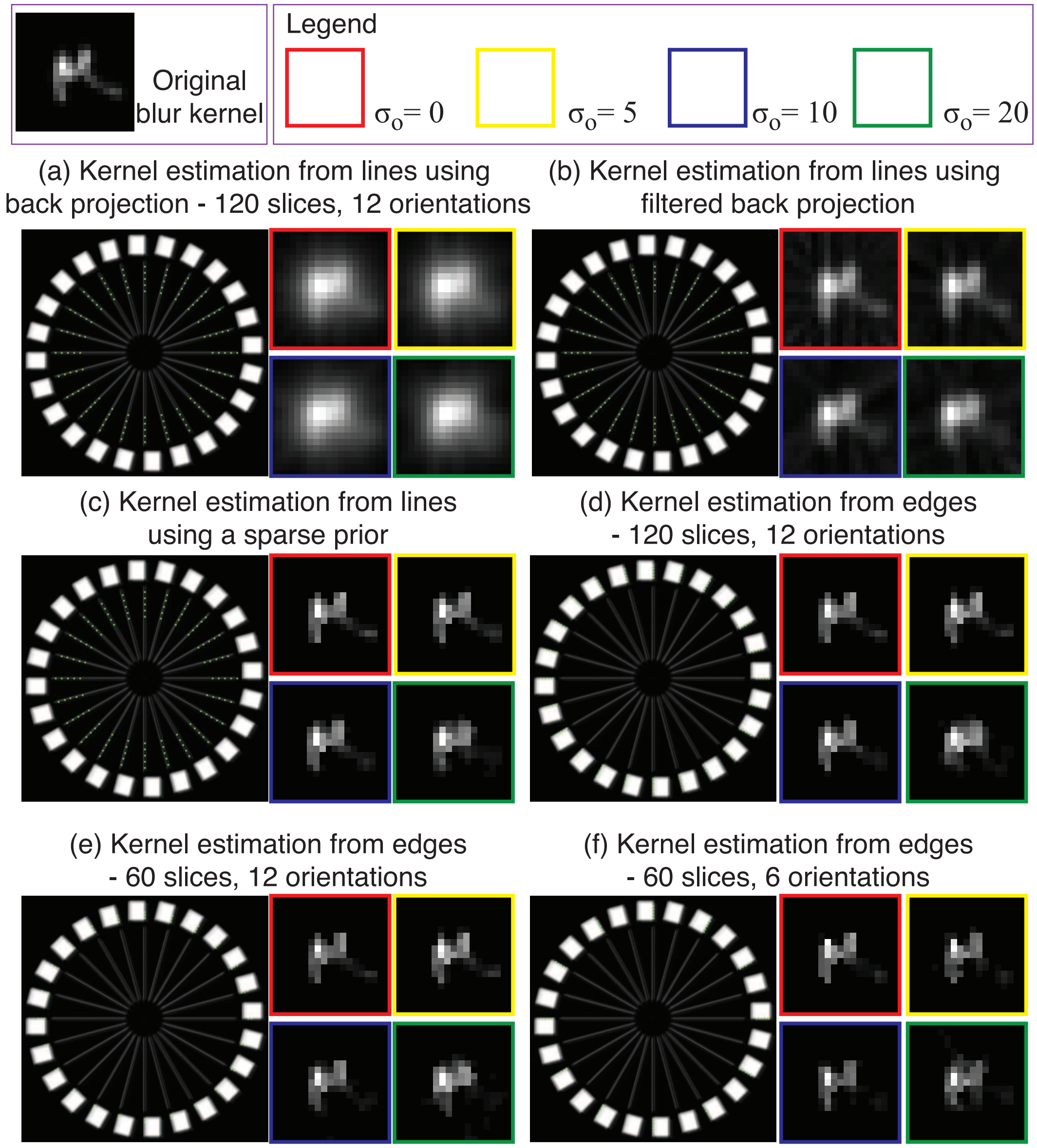

Figure 6: *We estimate a blur kernel from a synthetically blurred test pattern. We blur the test pattern using the blur kernel shown on the top left. As a first experiment, we compare three different inverse Radon transform algorithms: (a) the back projection algorithm (b) the filtered back projection algorithm (c) our algorithm in Eq. (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.*

As a first experiment, shown in Fig. 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. 6(a), a filtered back projection algorithm in Fig. 6(b), and our algorithm in Fig. 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. 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 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. 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. 6(f) while using 60 slices as in Fig. 6(e), however, our algorithm is less stable. This experiment shows that, if possible, we should take many edge samples in many orientations.

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 (Section 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. (7)) should lie between 0 and 1. An edge sample with a slice that lies outside of $0-\epsilon$ and $1+\epsilon$, where $\epsilon = 0.03$, 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:

$$\frac{\| \sum_{j \in N(l)} \exp(-i2\theta_j) \|}{\sum_{j \in N(l)} 1} \tag{13}$$

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.

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*. [8], Shan *et al*. [18], and Cho *et al*. [5]. In order to compare just the kernel estimation performance, we used the same deconvolution algorithm [11] to restore images.

Fig. 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. 8 shows more comparisons.

Our algorithm sometimes recover blur kernels with spurious "islands", as in Fig. 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. 9(a), and/or when there are not enough edges in different orientations, as shown in Fig. 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

Figure 7: *This figure compares our algorithm's kernel estimation performance to three previous work: Fergus* et al. *[8], Shan* et al. *[18], and Cho* et al. *[5]. In most examples, our algorithm compares favorably to prior art.*

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. 10, we do not need to recover the original blur kernel to adequately remove the blur. We can remove the blur from

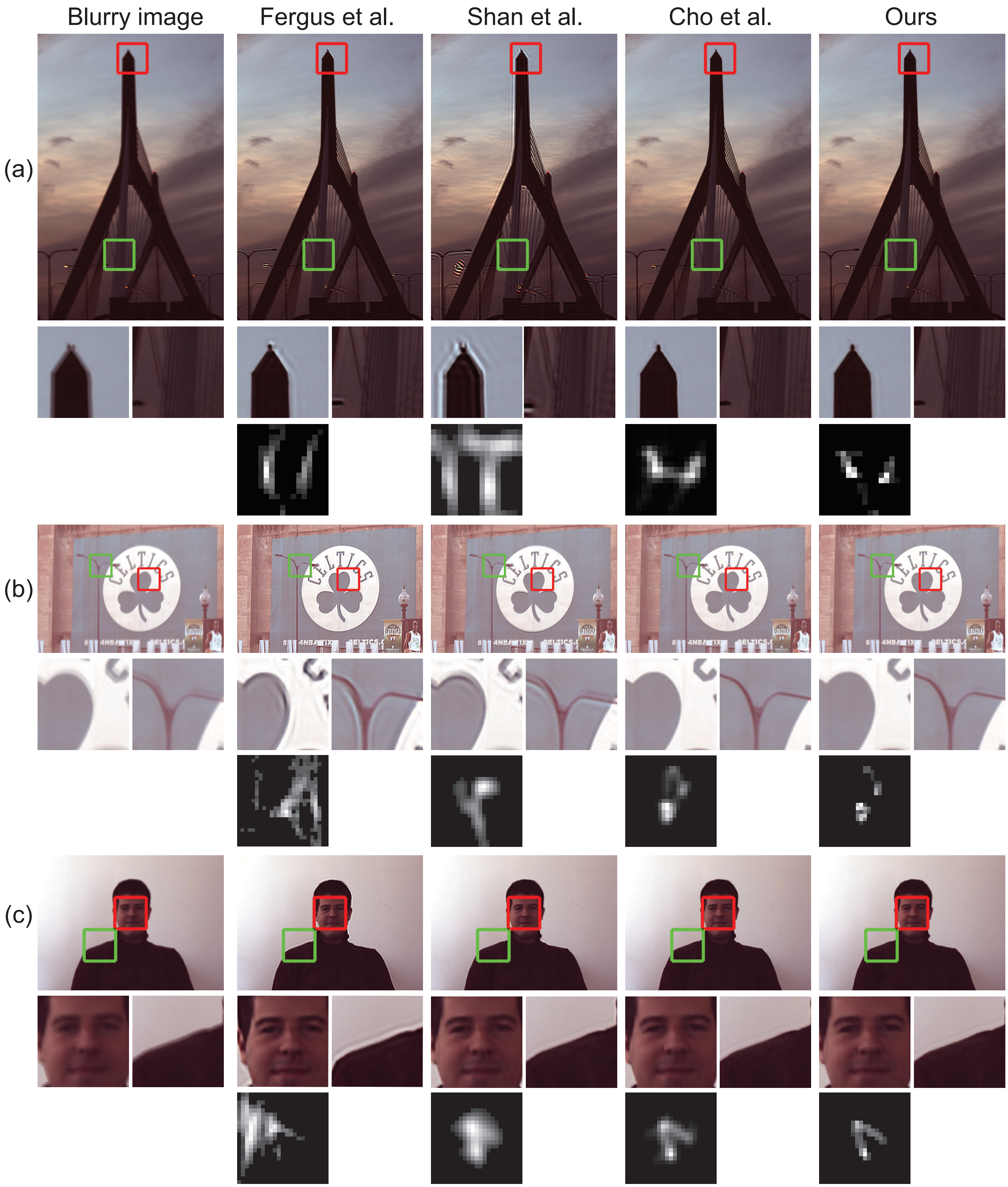

Figure 8: *This figure shows more comparisons of our kernel estimation algorithm and prior art.*

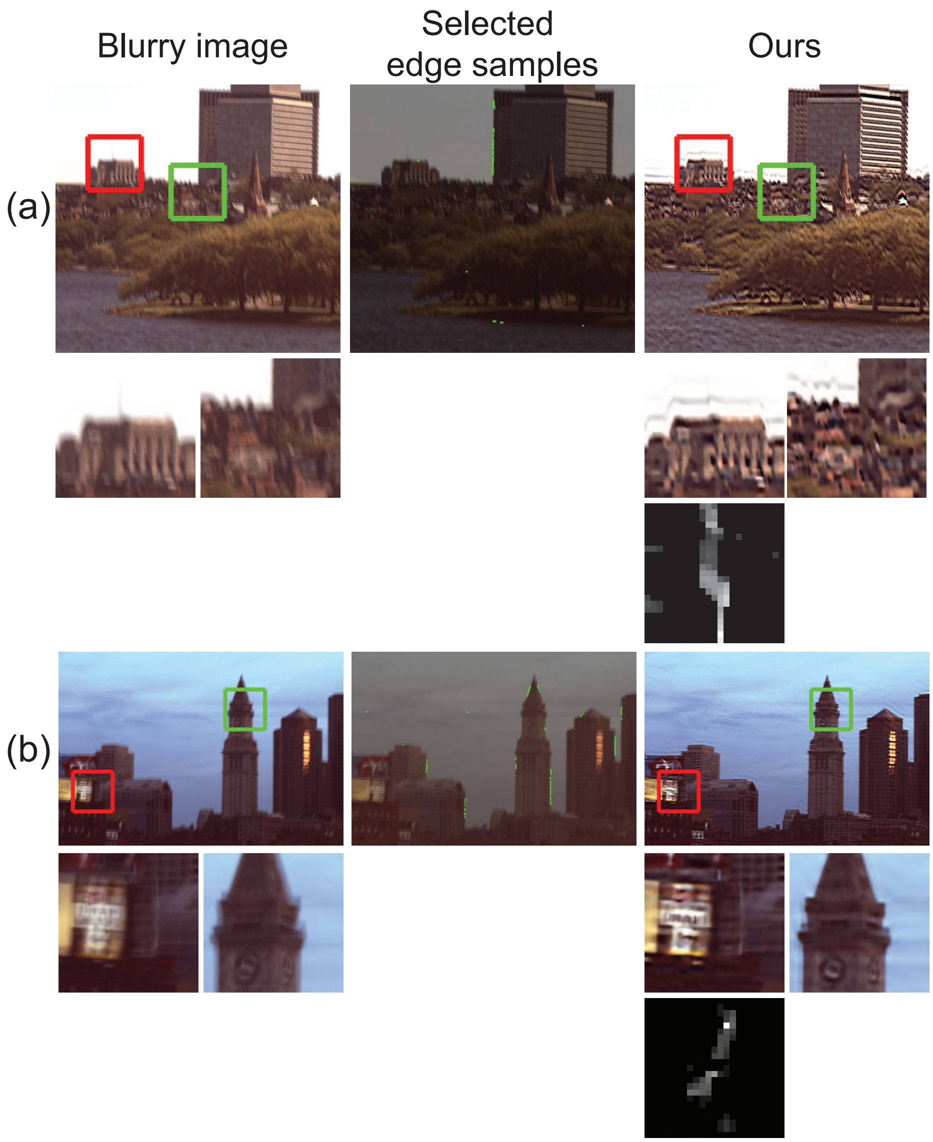

Figure 9: *Our kernel estimation algorithm is sensitive (a) when there are not enough edges and/or (b) when there are not enough edges in different orientations. This figure illustrates these failure modes.*

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. (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-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 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 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 (Section 2.1).

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

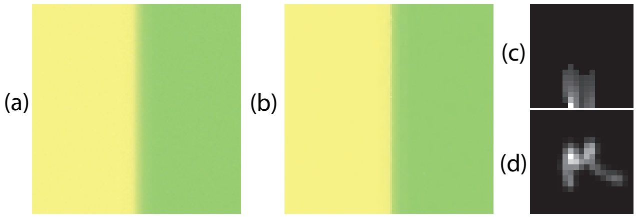

Figure 10: *(a) A blurred stripe (b) The deblurred stripe using the kernel estimated from our algorithm (c) Estimated blur kernel (d) The ground-truth blur kernel. Our kernel estimation algorithm only recovers the "horizontal" component of the ground-truth blur kernel, but the deblurred image is still crisp and is free of ringing.*

enough isolated 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 [18, 5]:

$$\begin{aligned}[\hat{k}, \hat{I}] &= \text{argmax}_{k,I}\, p(k, I|B) \\ &= \text{argmax}_{k,I}\, p(B|k, I)p(k)p(I) \end{aligned} \tag{14}$$

$[\hat{k}, \hat{I}]$ 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. (1)):

$$p(B|k, I) \propto \exp\left(-\frac{\|B - k \otimes I\|^2}{2\eta_n^2}\right) \tag{15}$$

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

$$p(I) \propto \exp\left(-\lambda\|\nabla I\|^\gamma\right) \tag{16}$$

The blur kernel prior $p(k)$ favors blur kernels with sparse intensity profiles as well as sparse gradient profiles (Eq. (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. (14): we first 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*. [12] argue that the joint estimation of the kernel and the sharp image is not a good idea because the joint probability Eq. (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. (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 [12].

Algorithm 1 The RadonMAP blur estimation algorithm

% Initial kernel estimation
$\hat{k} \Leftarrow \text{argmin}_k$ Eq. (12)
for l = 1 to 5 **do**
  $\hat{I} \Leftarrow \text{argmax}_I\, p(\hat{k}, I|B)$ *% Latent image estimation*
  $\hat{\mathbf{I}} \Leftarrow$ bilateralFiltering($\hat{I}$)
  $\hat{k} \Leftarrow \text{argmax}_k\, p(k, \hat{\mathbf{I}}|B)$ *% Kernel re-estimation*
end for
$\hat{I} \Leftarrow \text{argmax}_I\, p(\hat{k}, I|B)$

To resolve this issue, we augment the likelihood term in Eq. (15) using the Radon transform constraint in Eq. (10):

$$p(B|k, I) \propto \exp\left(-\left\{\frac{\|B - k \otimes I\|^2}{2\eta_n^2} + \frac{\sum_{j=1}^{360} w_j\|\tilde{\phi}_{\theta_j} - R_{\theta_j}k\|^2}{2\eta_p^2}\right\}\right) \tag{17}$$

The Radon transform term bases on a strong assumption that natural images consist of step edges and that every detected edge should be an ideal step edge. It effectively penalizes the no-blur solution, and steers the joint distribution $p(k, I|B)$ to favor the correct solution. Algorithm 1 shows the pseudocode for the joint estimation algorithm. We name this algorithm RadonMAP.

Notice that we filter the latent image estimate $\hat{I}$ using a bilateral filter before re-estimating the kernel, as in [5]. The bilateral filter step is important for improving the kernel estimation performance. $\hat{I}$ usually contains visually disturbing ringing and noise because the initial kernel estimate is inaccurate. If we directly use $\hat{I}$ to refine the blur kernel, we would be seeking a blur kernel that would reconstruct $\hat{I}$ from B. To improve the blur kernel, we bilateral-filter the latent image so that the refined blur kernel restores an image with less ringing and noise.

Experimental results Fig. 11(a-b) show how the RadonMAP algorithm improves the failure cases shown in Fig. 9. The images deblurred using the MAP kernel estimation algorithm 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. 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 the no-blur solution. The negative log-posterior

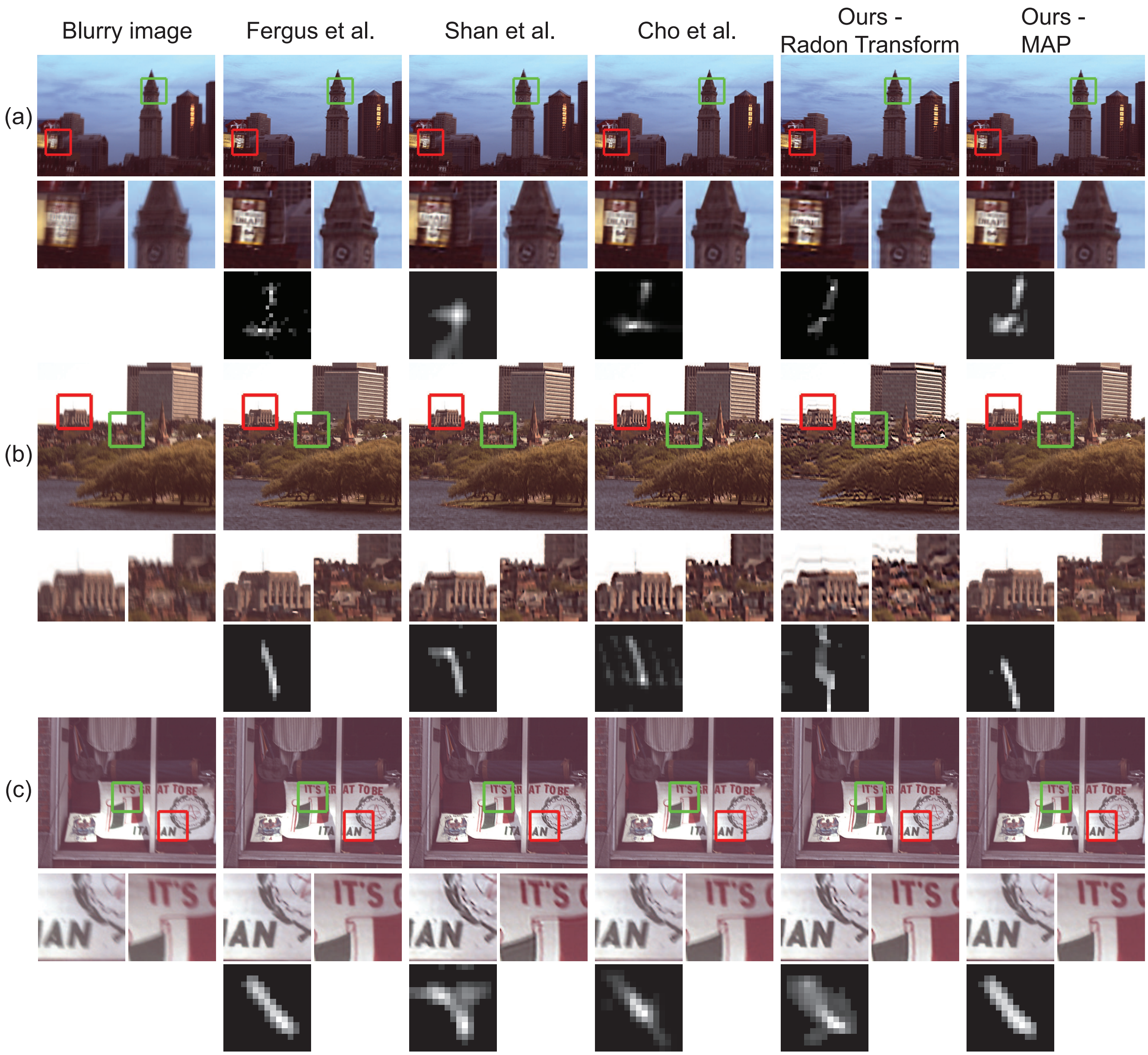

Figure 11: *By integrating kernel projection constraints to a MAP kernel estimation method, we can improve the kernel estimation performance. We name this algorithm RadonMap. (a-b) show that even when there are not enough edges in different orientations (as shown in Fig. 9), the RadonMAP algorithm can reliably reconstruct the kernel.*

of our solution in Fig. 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.

4.1. Quantitative evaluation

We quantify the performance of blur estimation algorithms using cumulative error ratio [12]. Error ratio (ER) measures the deconvolution error of using the estimated blur kernel compared to the deconvolution error of using the ground-truth kernel. In particular, ER is defined as follows:

$$ER = \frac{\|I - D_{est}\|^2}{\|I - D_{gt}\|^2} \tag{18}$$

where D_{est} is the image restored using the estimated blur kernel, and D_{gt} is the image restored using the ground-truth blur kernel. Levin *et al.* [12] provide a set of test images and blur kernels for comparisons. However, the test images are too small (255×255 pixels) and do not have salient step

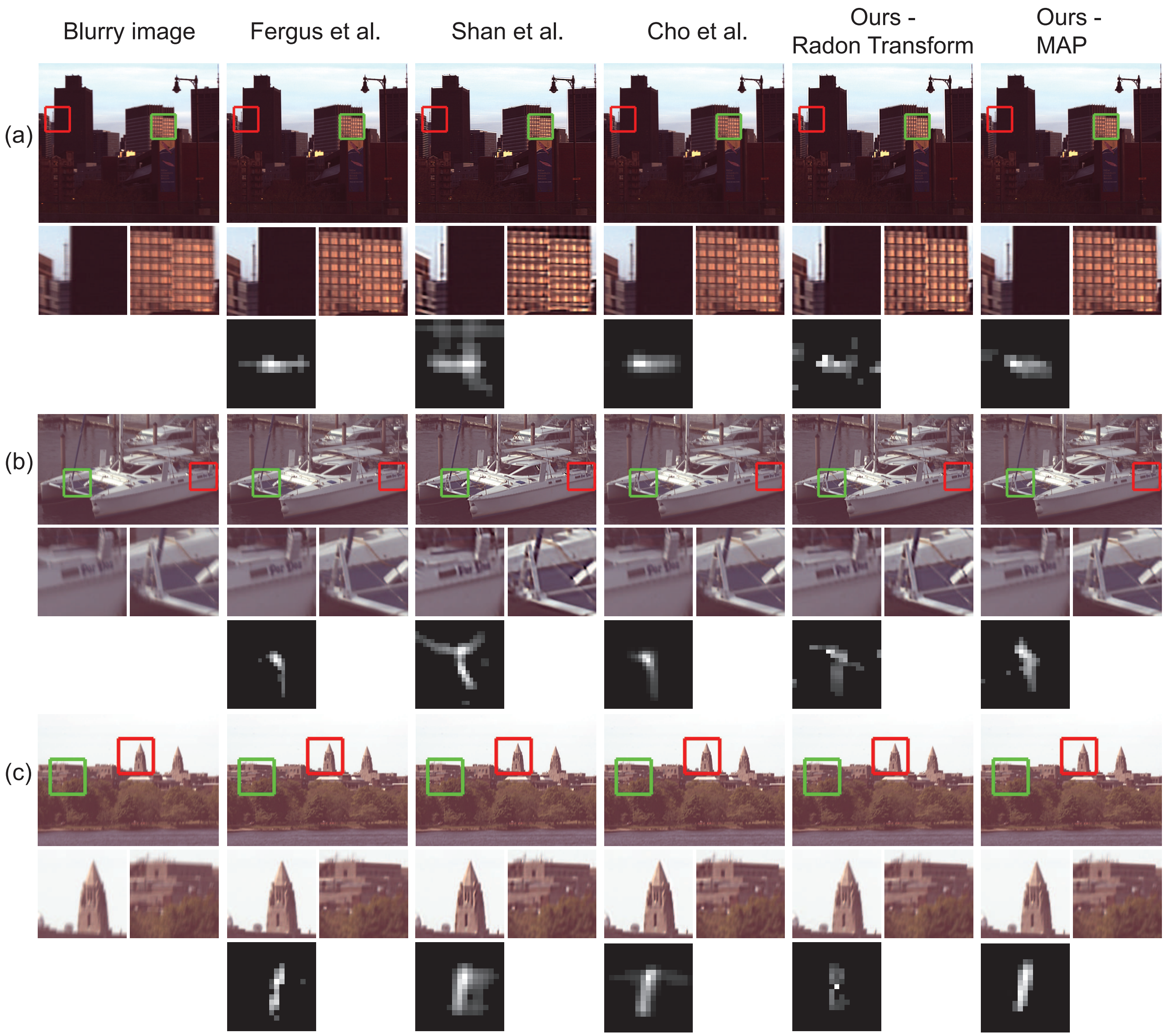

Figure 12: *We show more examples in which the MAP kernel estimation algorithm improves the estimated blur kernels.*

edges. To address these issues, we have hand-selected 6 images (each with about 1 mega pixels) of different contents, and computed ER for each blur kernel provided in Levin *et al*.'s dataset. Figure Fig. 13 shows the test images and blur kernels: each algorithm is tested with 48 blurred images. Images 1–3 contain many edges in different orientations, whereas images 4–6 do not. Therefore, we can conjecture that our algorithms would perform better on images 1–3 than on images 4–6.

Figure Fig. 14 shows the cumulative error ratio for each deblurring algorithm. The inverse Radon transform of blur projections performs better than the algorithms presented in Fergus *et al*. [8] and Shan *et al*. [18], but performs worse than the algorithm in Cho *et al*. [5]. Augmenting the blur kernel projection constraints in a MAP framework (the RadonMAP algorithm) improves the performance of our algorithm, but it still falls short of Cho *et al*.'s algorithm.

To gain more insight, we have plotted the cumulative error ratio for images 1–3 and images 4–6 separately in Fig. 15. This figure shows an interesting trend that for piecewise smooth images with enough edges in many orientations (i.e. images 1–3), the RadonMAP algorithm outperforms all existing algorithms, but if images lack such edges (i.e. images 4–6), Cho's algorithm performs the best. Still, even in

Figure 13: *We evaluate the blur estimation performance of five different blur estimation algorithms. We synthetically blur each image with each blur kernel shown in this figure, and estimate blur kernels from each of them using five competing algorithms. The algorithms' performance is measured using cumulative error ratio.*

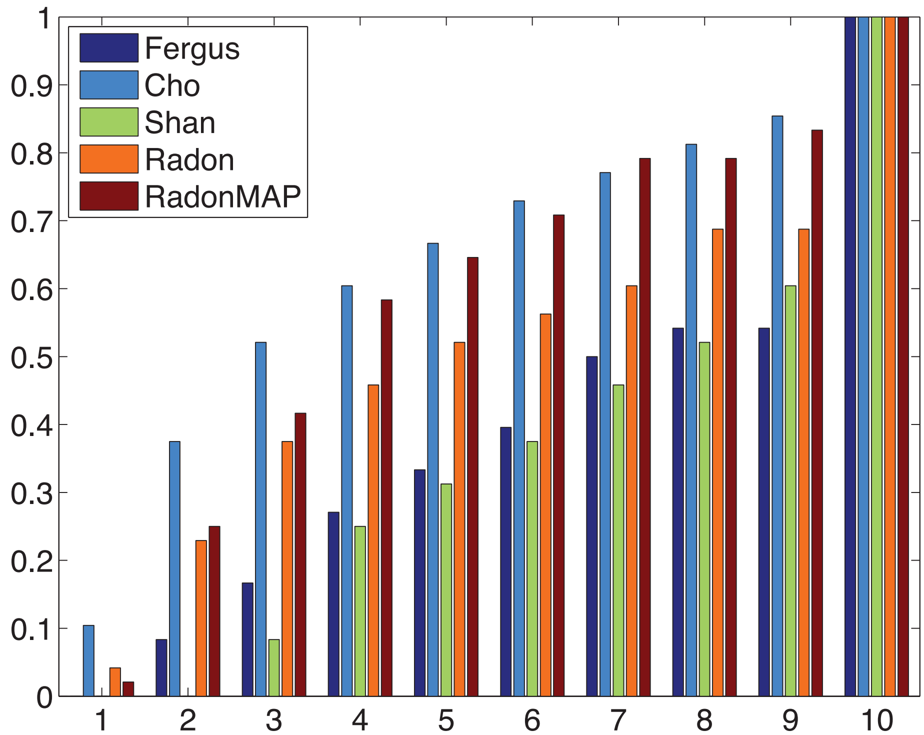

Figure 14: *This figure shows the cumulative error ratio for five blur estimation algorithms: Fergus* et al. *[8], Shan* et al. *[18], Cho* et al. *[5], the inverse Radon transform based blur estimation in Section 2.2 (named Radon in the legend), and the MAP algorithm augmented with the Radon transform constraints in this section (named RadonMAP in the legend). From the cumulative error ratio, the algorithm in Cho* et al. *performs the best, closely followed by the MAP algorithm augmented with the Radon transform.*

such scenarios our algorithms compare favorably to Fergus *et al*.'s and Shan *et al*.'s algorithms.

We have also observed that the size of the blur also affects the performance of our algorithm. Therefore, we have plotted the cumulative error ratio for blur kernel 1–4 and 5–7 in Fig. 13 separately in Fig. 16. Blur kernels 5–8 have larger spatial supports compared to blur kernels 1–4. Interestingly, when the blur kernel support is small, the Radon transform based algorithms perform slightly better than Cho's algorithm, but when the blur kernel support is large, our algorithms suffer. This issue can be attributed to edge detection: (i) the number of stable edges decreases as the blur kernel support increases because more edges are contaminated by otherwise isolated neighboring edges (ii) the stable edge detection becomes more challenging because a single isolated edge is often interpreted as two edges that starts at one end of the blurred profile and stops at the other end. We could reduce such blur size dependencies by extending our algorithms in a multi-scale manner.

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.

References

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

[2] J.-F. Cai, H. Ji, C. Liu, and Z. 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.

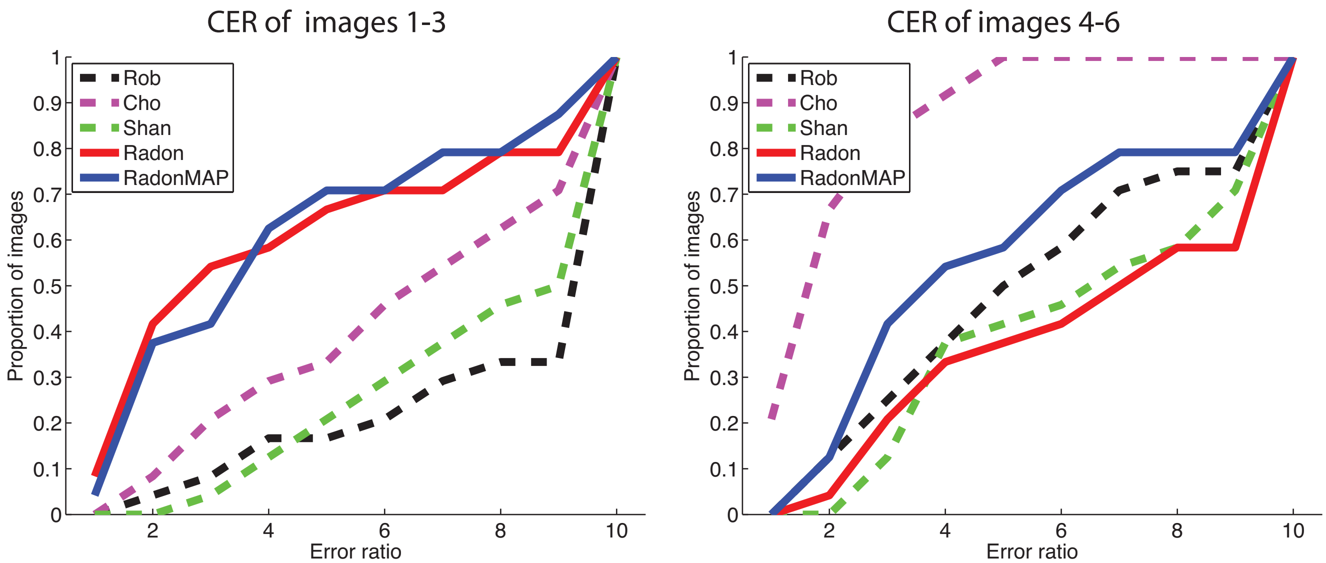

Figure 15: *In this figure, we plot the cumulative error ratio for images 1–3 and images 4–6 separately. The inverse Radon transform of kernel projections (named Radon) and the MAP estimation with kernel projection constraints (named RadonMAP) both perform well when images contain many edges in different orientations as in images 1–3, but their performance drops drastically when images do not contain enough edges as in images 4–6.*

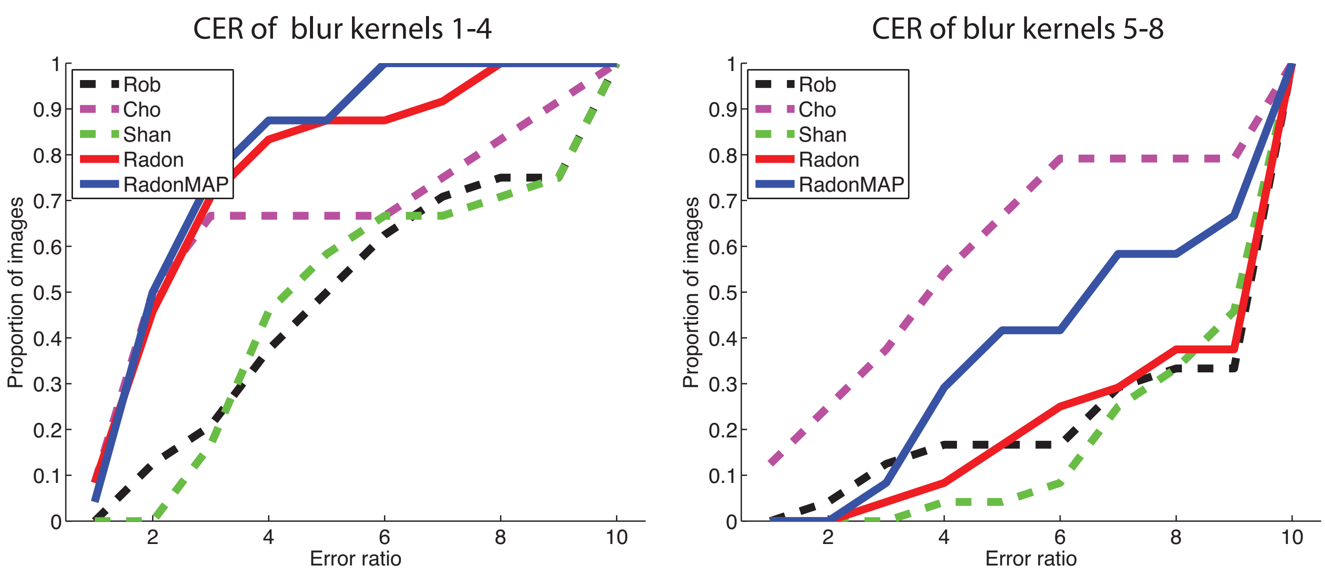

Figure 16: *The performance of the inverse Radon transform of kernel projections (named Radon) and the MAP estimation with kernel projection constraints (named RadonMAP) depend on the size of the blur. In this figure, we plot the cumulative error ratio for kernels 1–4 and kernels 5–8 in Fig. 13 separately. We observe that both Radon and RadonMAP are state-of-the-art when blur kernels are small, but their performance drops drastically when blur kernels are large.*

[3] J.-F. Cai, H. Ji, C. Liu, and Z. 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.

[4] J. Chen, L. Yuan, C.-K. Tang, and L. Quan. Robust dual motion deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

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

[6] S. Cho, Y. Matsushita, and S. Lee. Removing non-uniform motion blur from images. In *Proceedings of International*

Conference on Computer Vision (ICCV), 2007.

[7] S. R. Deans. *The Radon Transform and some of its applications*. Krieger Publishing Company, 1992.

[8] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single image. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2006.

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

[10] N. Joshi, R. Szeliski, and D. J. Kriegman. PSF estimation using sharp edge prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[11] A. Levin, R. Fergus, F. Durand, and W. T. Freeman. Image and depth from a conventional camera with a coded aperture. *ACM TOG (Proc. SIGGRAPH)*, 2007.

[12] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[13] Y. Li, S. B. Kang, N. Joshi, S. Seitz, and D. Huttenlocher. Generating sharp panoramas from motion-blurred videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

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

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

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

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

[18] Q. Shan, L. J. Jia, and A. Agarwala. High-quality motion deblurring from a single image. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2008.

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

[20] Y.-W. Tai, H. Du, M. S. Brown, and S. Lin. Image/video deblurring using a hybrid camera. In *CVPR*, 2008.

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

[22] L. Yuan, J. Sun, L. Quan, and H.-Y. Shum. Image deblurring with blurred/noisy image pairs. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 2007.