Edge-based Blur Kernel Estimation Using Patch Priors

Libin Sun*
Brown University
lbsun@cs.brown.edu

Sunghyun Cho
Adobe Research
scho@adobe.com

Jue Wang
Adobe Research
juewang@adobe.com

James Hays
Brown University
hays@cs.brown.edu

Abstract

Blind image deconvolution, i.e., estimating a blur kernel k and a latent image x from an input blurred image y, is a severely ill-posed problem. In this paper we introduce a new patch-based strategy for kernel estimation in blind deconvolution. Our approach estimates a "trusted" subset of x by imposing a patch prior specifically tailored towards modeling the appearance of image edge and corner primitives. To choose proper patch priors we examine both statistical priors learned from a natural image dataset and a simple patch prior from synthetic structures. We show that our patch prior prefers sharp image content to blurry ones. Based on the patch priors, we iteratively recover the partial latent image x and the blur kernel k. A comprehensive evaluation shows that our approach achieves state-of-the-art results for uniformly blurred images.

1. Introduction

Image blur caused by camera shake is a common problem in consumer photography. A motion blurred image y capturing a static scene is often modeled as:

$$y = k * x + n, \quad (1)$$

where k is the blur kernel, x is the latent image, n is noise, and $*$ is the convolution operator. For blind deconvolution, the goal is to recover both x and k from y, which is an ill-posed problem.

To overcome the ill-posedness of blind deconvolution, previous works place strong assumptions or prior knowledge on k and x. Regarding k, it is often assumed that k should be sparse [7, 16] and continuous [3]. For x, it is often assumed that image gradients are heavy-tailed [7, 16, 13, 14, 11]. However, we argue that this popular family of sparsity priors is not suitable for the task of kernel estimation for the following reasons. First, sparsity priors prefer blurry images to sharp ones [13], because blur reduces overall gradient magnitude. Hence, sparsity priors have limited capacity to steer the latent image towards a sharp solution. Second, they fundamentally suffer from the fact that the unit of representation is extremely limited: gradient filters often consider two or three pixels, hence ignoring longer-range dependencies which give rise to the most salient image structures and geometry. This is also why state-of-the-art image restoration methods often involve larger neighborhoods or image patches [2, 15, 18, 20].

Another family of blind deconvolution algorithms explicitly exploits edges for kernel estimation. Joshi *et al*. [8] and Cho *et al*. [5] directly restore sharp edges from blurry edges and rely on them to estimate the blur kernel. While these methods work well for small scale blur, they have difficulty dealing with large blur kernels, as directly restoring sharp edges from a severely blurred image is non-trivial. To handle large blur kernels, Cho and Lee [4] introduce an edge-based approach, which alternates between restoring sharp edges and estimating the blur kernel in a coarse-to-fine fashion. This framework has been further extended by Xu and Jia [19] and has proven to be effective [9]. However, these approaches heavily rely on heuristic image filters such as shock and bilateral filtering for restoring sharp edges, which are often unstable, as we will show in Sec. 3.3.

In this paper, we propose a new edge-based approach using *patch* priors on edges of the latent image x. Patches can model image structures better than filter responses. In our approach, we estimate a "trusted" subset of x by imposing patch priors specifically tailored towards modeling the appearance of image edge and corner primitives. We only restore these primitives since other image regions, *e.g*. flat or highly-textured ones, do not carry much useful blur information for kernel estimation. Furthermore, restoring textures often results in hallucinated high frequency content, which corrupts the subsequent kernel estimation steps. We illustrate how to incorporate the patch prior into an edge-based iterative blind deconvolution framework through an optimization process, where we iteratively recover the partial latent image x and the blur kernel k. Experimental results show that our approach achieves state-of-the-art results (Sec. 5).

The main question we address in this paper is *what is*

*Part of the work was done while the first author was an intern at Adobe Resarch.

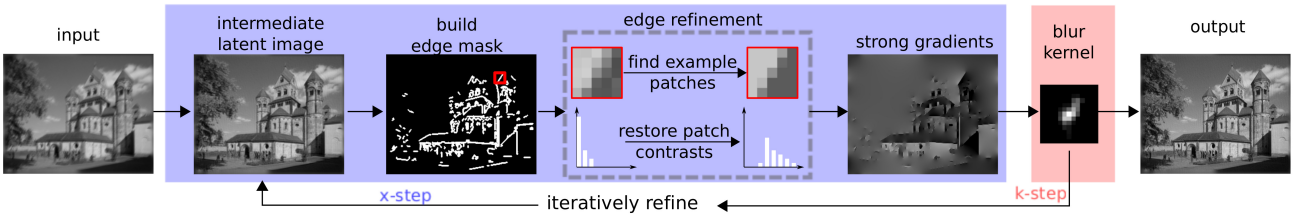

Figure 1. Algorithm pipeline. Our algorithm iterates between x-step and k-step with the help of a patch prior for edge refinement process. In particular, we coerce edges to become sharp and increase local contrast for edge patches. The blur kernel is then updated using the strong gradients from the restored latent image. After kernel estimation, the method of [20] is used for final non-blind deconvolution.

the right prior for x in blind deconvolution. Intuitively, the image prior used for blind deconvolution should not be too expressive, *i.e.*, the prior should not be allowed to *express* a wide range of visual phenomena, such as motion blur and defocus blur, which are natural and frequent in professional photographs. A prior that is overly expressive, such as the GMM model proposed by Zoran and Weiss [20], will inevitably accommodate blur and hinder the the convergence of the solution pair. With this in mind, our patch-based prior is specifically tailored towards modeling particular image primitives [6]– atomic elements that form the structural part of the image, namely, edges, corners, T-junctions, *etc*. To choose proper patch priors, we examine both statistical priors learned from a natural image dataset and a simple patch prior from synthetic structures. Experimental results show that, surprisingly, the simple synthetic patch prior can generate the same quality or even better results than the learned statistical prior.

2. Algorithm Overview

Our algorithm builds upon the coarse-to-fine, iterative optimization framework commonly used in recent methods [16, 4, 19, 11], as shown in Fig. 1. In particular, at each level s, we initialize k_s by upsampling k_{s-1} from the previous level, followed by a latent recovery step to solve for x_s (x-step). However, our x-step only attempts to partially restore x_s, *i.e.* edge primitives, which we deem as the only reliable image regions for both x-step and k-step. Given a newly updated x_s, k_s is updated (k-step) by comparing restored gradients and the blurred image. This procedure is carried out iteratively until convergence and the resulting k_s is propagated to the next level to initialize k_{s+1}. Unlike [4], we do not rely on heuristic image filtering steps to "guess" the ground truth gradients in the x-step. Instead, we introduce a nonparametric patch prior to coerce image primitives in x to be sharp and noise-free. This also avoids the problem of falsely penalizing large gradients, which may happen if a sparsity prior is applied over image gradients [13].

We will first introduce our patch prior formulation in Sec. 3, then describe how to incorporate the patch prior for kernel estimation in Sec. 4.

3. Nonparametric Patch Priors

Most iterative deblurring methods are carefully engineered to *drive* the latent image towards a sharper solution during the optimization process, and avoid the degenerate case of a δ PSF and a blurry x. To achieve this, we introduce two sets of independent auxiliary variables, namely $\{Z^i\}$ and $\{\sigma^i\}$, for each pixel location i considered as an edge primitive. Z^i is a particular example patch assigned to location i, and σ^i is the target local contrast we wish the latent image patch to have. Together they provide strong constraints over desired structural shapes and sharpness for primitive patches in the latent image. Note that we model patches in the normalized space: subtracting its mean and dividing by standard deviation.

We would like our patch prior to be sufficiently expressive, *i.e.*, any edge patch P in natural images can be approximated by $P = \sigma Z + \mu + \epsilon$, where σ is the patch contrast, μ is the patch intensity, ϵ is a small error term. However, this prior cannot be overly expressive, *i.e.*, it should not be allowed to express high frequency textures or gradual changes in image gradients, otherwise it will start to accommodate blur and noise in the latent image, hence losing its power to restore sharpness. Since our prior is only applied to a *subset* of pixels in the latent image, it should be distinguished from existing *generic* natural image priors such as the simple sparsity prior [13] and the complex GMM-based patch prior from Zoran and Weiss [20], which are applied over the whole image.

We will first examine how we learn a set of representative natural edge patches $\{Z_{nat}\}$. We then introduce a set of synthetic patches $\{Z_{synth}\}$ as an alternative solution.

3.1. Learning a Natural Edge Patch Prior

To model image primitives, a natural approach is to learn a prior from a training set of patches extracted along image contours. We first downsample all 500 images (grayscale) from the BSDS500 dataset [1] by half in each dimension to reduce noise and compression artifacts, then compute a mask (see Sec. 4 for more details) based on gradient magnitude. The mask is then intersected (AND operator) with human annotated ground truth contours provided by the

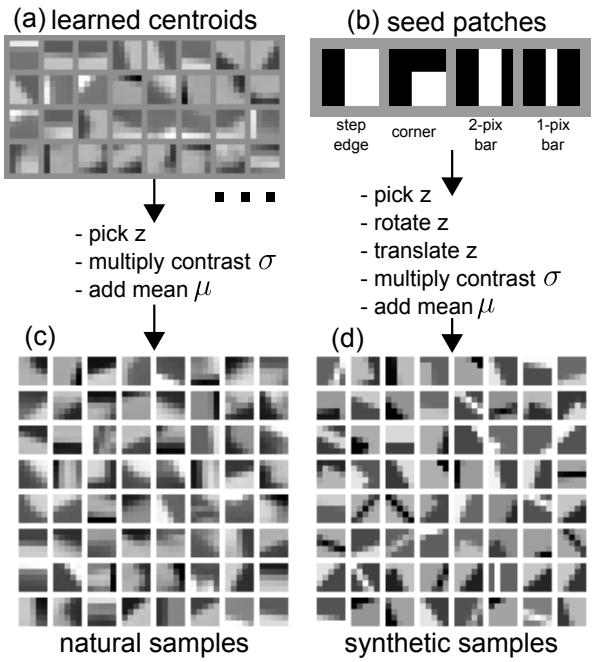

Figure 2. The generative process from our natural prior (left) and synthetic patch prior (right). (a) 32 of the 2560 learned centroids, with decreasing cluster size in scan-line order. (b) four basic structure seed patches we use as bases for our synthetic prior. (c) 64 random patch samples generated from $\{Z_{nat}\}$. (d) 64 random patch samples generated from $\{Z_{synth}\}$.

dataset to produce a final mask. We extract all 5×5 patches centered within the mask, resulting in approximately 220k patches from 500 images. We normalize each patch by removing its DC and dividing by its standard deviation (local contrast). Finally, we apply a standard k-means algorithm to learn the representative primitive patches: $\{Z_{nat}\}$ is the set of centroids. For our experiments, we use $k = 2560$. In Fig. 2(a), we show 32 such centroids, which are regularly sampled from the 2560 centroids after sorting them by cluster size. Note that we expect to have enough training patches such that the cluster centroids encode the appropriate variations in orientation and translation, hence we do not apply such transformations to $\{Z_{nat}\}$ during the optimization process.

We also learn the distribution of local contrast encoded by σ for such primitive patches. The empirical distribution of σ is shown in Fig. 3, which will be used in our framework to restore diminished/blurred image gradients in Sec. 4.

We make the following observations about our learned patch priors:

1. Patch complexity is correlated with cluster size. In particular, simple edge structures such as horizontal/vertical step edges make up the largest clusters, and the patch samples within these clusters exhibit little variation. On the other hand, the smallest clusters capture complex textures and noisy structures, and the patch samples within these clusters can be significantly different from each other. This is consistent with recent findings from Levin *et al*. [12].

2. The overall complexity of image primitives is surprisingly limited. Using $k = 2560$, there is already a good amount of redundancy in the clusters: some centroids appear almost identical to each other. Furthermore, most of the clusters capture simple step edges with varying profiles, orientations and translations. Some smaller clusters represent corners and other rare structures.

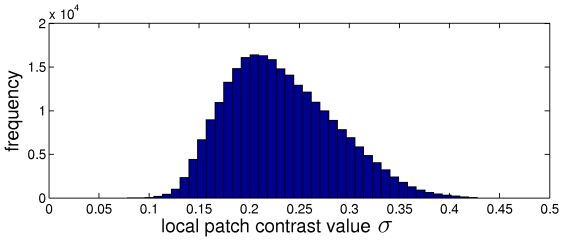

Figure 3. The empirical distribution of local constrasts from the 220k patches collected from BSDS500 [1]. The distribution is asymmetric and heavy-tailed, indicating a fair amount contrast in image primitives should be large.

3.2. A Synthetic Edge Patch Prior

The learned natural edge patches are relatively simple, but contain slight blur and noise as a result of averaging real image patches with slight misalignment. For this reason, we design a synthetic patch prior that is hopefully comparable in terms of expressiveness, while being cleaner and sharper.

To generate a set of synthetic patches $\{Z_{synth}\}$, we use four *seed patches* as shown in Fig. 2(b). These seed patches are one step edge, one corner and two bars of different widths. We consider two kinds of transformations: rotations (every 3 degrees from 0 to 360), and translations (up to $\pm 1, 2$ pixels in each direction). The set of example patches $\{Z_{synth}\}$ is generated by applying all possible combinations of these transformations to each seed patch. All patches are then normalized by subtracting its mean and dividing by its standard deviation. Fig. 2(d) provides a visualization of 64 random samples from this synthetic patch prior.

In this work we demonstrate that this synthetic prior $\{Z_{synth}\}$ works roughly as well as $\{Z_{nat}\}$, indicating that the complexity of 5×5 edge patches is limited, and that most properties of image primitives are intuitive: sharp, noise-free and geometrically simple. Nonetheless, they are still a powerful prior for deblurring.

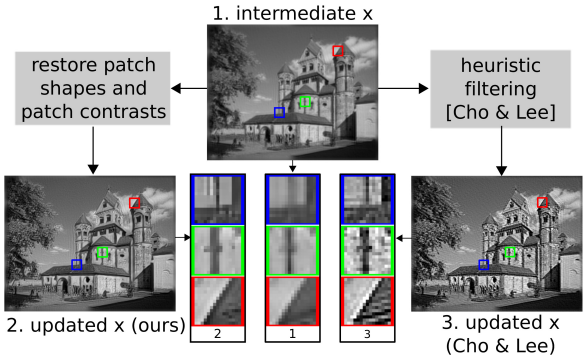

Figure 4. Restoring edges using heuristic image filtering [4] is sensitive to noise and can lead to ringing artifacts, whereas our patch-based optimization can avoid such issues by modeling larger neighborhoods.

3.3. Behavior Comparison

Cho and Lee [4] restore sharpness via a so called prediction step involving heuristic image filtering. Specifically, bilateral filtering is used to reduce noise and shock filtering is used to sharpen edges, both are applied after solving for x under a Gaussian prior over image gradients. However, such filtering procedures often fail in the presence of dense image structures and noise along edges. In contrast, we replace these steps by a more principled formulation that is robust to noise (more so than bilateral filtering) and produces sharp edges (more so than shock filtering).

In Fig. 4 we provide an illustration of how our approach can efficiently remove blur via restoring both the patch shape (Z) and patch contrast (σ). It shows the before/after comparison for an intermediate latent image x under one iteration of optimization using Cho and Lee's method, and ours. The comparison suggests that our method generates higher quality intermediate latent image x, which naturally leads to a higher quality kernel estimation in the k-step.

4. Kernel Estimation using Patch Priors

Like the deblurring framework from Cho and Lee [4], our kernel estimation approach iterates between x- and k-steps, in a coarse-to-fine manner. We will discuss these two steps separately. At the coarsest level, we initialize k by a small 3×3 Gaussian.

4.1. x-step

Given the current estimate of the blur kernel k, the goal of x-step is to produce a latent image x (or some trusted subset) that is sharp and free of artifacts (ringing, halos, etc). To estimate x we minimize the following energy function:

$$\begin{aligned} f_{\mathbf{x}}(\mathbf{x}) &= \sum_{\mathbf{D}_*} \omega_* \|\mathbf{K}\mathbf{D}_*\mathbf{x} - \mathbf{D}_*\mathbf{y}\|^2 \\ &+ \alpha\|\mathbf{D}_h\mathbf{x}\|^2 + \alpha\|\mathbf{D}_v\mathbf{x}\|^2 \\ &+ \frac{\beta}{|M|}\sum_{i\in M}\rho\left(\mathbf{P}_i\mathbf{x} - \mathbf{q}^i\right) \\ &+ \gamma\sum_{i\in M}\left(\sigma^i - F_{ref}^{-1}(F_{\sigma,x}(\sigma^i))\right)^2, \end{aligned} \quad (2)$$

where $\mathbf{K}$, $\mathbf{y}$ and $\mathbf{x}$ represent the matrix form of the blur kernel k, the blurred input image y, and the latent image x, respectively. $\mathbf{D}_*$ is the matrix form of the partial derivative operator in different directions and ω_* represents the corresponding scalar weight. As done in [16, 4], we also use zero-th, first, and second order derivatives for $\mathbf{D}_*$. We borrow the weights used in [16] for ω_*. $\mathbf{D}_h$ and $\mathbf{D}_v$ are the first order differentiation operators along the horizontal and vertical axes. $\mathbf{P}_i$ is a binary matrix extraction operator, extracting the patch at location i in the latent image x. In this work, we fix the patch size at 5×5. $\mathbf{q}_i$ is defined as $\mathbf{q}^i = \sigma^i\mathbf{Z}^i + \boldsymbol{\mu}^i$ where $\mathbf{Z}^i$ is a vector representing Z^i, the example patch assigned to location i.

We maintain a binary mask M to indicate pixel locations classified as edge primitives, and only apply the patch prior to such locations to encourage sharpness. The other regions in x are weakly regularized by a Gaussian prior over image derivatives. The use of such "edge" masks is adopted in [8] as well. $|M|$ is the number of non-zero elements in M. $\rho()$ is a robust penalty function. In particular, we use the Lorentzian loss function, defined as $\rho(r) = \log\left(1 + \frac{r^2}{2\epsilon^2}\right)$. Finally, $F_{\sigma,x}$ is the empirical cumulative distribution of $\{\sigma^i\}$ in the current latent image x, F_{ref} is the reference cumulative distribution of local contrasts, based on the learned distribution in Fig. 3.

Intuitively, the first term is the data term, enforcing the blur model. The second and third term yield a weak Gaussian prior to regularize image smoothness. Note that while regions outside the mask M do not participate in the k-step, we only use this term to weakly regularize the energy function so that optimization becomes stable. The last two terms encode our patch prior involving Z^i and σ^i, providing two strong constraints: (1) edge primitive patches in x should be similar to some example patch (after normalization), (2) the distribution of σ's in x should be similar to a reference distribution.

Directly optimizing Eqn. (2) is hard. We present an iterative approximation procedure to update the variables M, $\{Z^i\}$, $\{\sigma^i\}$ and x below. For the first iteration, we set $M = \emptyset$, meaning that we use only the Gaussian prior to initialize an intermediate latent image x, then the following procedures are applied until convergence:

1. Update M: We first obtain a binary mask by keeping the top 2% of pixel locations with the largest filter responses from a filter bank consisting of derivatives of elongated Gaussians in eight orientations. We morphologically thin this mask and then remove small isolated components. This step chooses the right locations to apply our edge patch prior.

2. Update σ^i: Fixing M, Z^i and x, we use an iterative reweighted least squares (IRLS) method to optimize Eqn. (2) with respect to σ. We present the full derivation in the supplementary material[1], but the main steps for IRLS are as follows:

   a. Let $\mathbf{r}^i = \mathbf{P}_i\mathbf{x} - \mathbf{q}^i$, compute weights w_i given current σ^i by:

   $$w_i = \left(2\epsilon^2 + \mathbf{r}_i^T\mathbf{r}_i\right)^{-1}, \tag{3}$$

   b. Update σ^i by solving a weighted least square problem:

   $$\sigma^i \leftarrow \frac{\frac{w_i\beta}{|M|}\mathbf{Z}^{i^T}\left(\mathbf{P}_i\mathbf{x} - \boldsymbol{\mu}^i\right) - \gamma F_{ref}^{-1}(F_{\sigma,x}(\sigma^i))}{\frac{w_i\beta}{|M|}\mathbf{Z}^{i^T}\mathbf{Z}^i - \gamma}. \tag{4}$$

3. Update Z^i: Holding other variables constant, we find example patch Z^i that is most similar to $\frac{\mathbf{P}_i\mathbf{x}-\boldsymbol{\mu}^i}{\sigma^i}$, for each location i.

4. Update x: Holding other variables constant, x is updated by solving the following for x:

$$\mathcal{F}^{-1}\left(\mathbf{A}\odot\mathcal{F}(x)\right) + \frac{\beta}{|M|}\sum_{i\in M}\frac{2}{2\epsilon^2+\mathbf{r}_i^T\mathbf{r}_i}\mathbf{P}_i^T\mathbf{P}_i\mathbf{x}$$
$$= \mathcal{F}^{-1}(\mathbf{B}) + \frac{\beta}{|M|}\sum_{i\in M}\frac{2}{2\epsilon^2+\mathbf{r}_i^T\mathbf{r}_i}\mathbf{P}_i^T(\sigma^i\mathbf{Z}^i + \boldsymbol{\mu}^i),$$

where

$$\mathbf{A} = \left(\sum_{\delta_*}\omega_*\overline{\mathcal{F}(\delta_*)}\odot\mathcal{F}(\delta_*)\right)\odot\overline{\mathcal{F}(k)}\odot\mathcal{F}(k) + \alpha\sum_{\delta_x,\delta_y}\overline{\mathcal{F}(\delta_*)}\odot\mathcal{F}(\delta_*),$$

and

$$\mathbf{B} = \left(\sum_{\delta_*}\omega_*\overline{\mathcal{F}(\delta_*)}\odot\mathcal{F}(\delta_*)\right)\odot\overline{\mathcal{F}(k)}\odot\mathcal{F}(y).$$

We use $\mathcal{F}$ to represent the Fourier transform and $\overline{\mathcal{F}}$ for its complex conjugate. $\odot$ is the element-wise multiply operator. Note that this equation is no longer linear in x because $\mathbf{r}_i$ involves $\mathbf{x}$ as well. Hence, we use IRLS to iteratively optimize $\mathbf{x}$, where the diagonal weighting matrix has entries $w_{ii} = \frac{2}{2\epsilon^2+\mathbf{r}_i^T\mathbf{r}_i}\forall i \in M$.

[1]Please refer to our project page: http://cs.brown.edu/~lbsun/deblur2013iccp.html

4.2. k-step

In this step, we hold x constant and optimize with respect to k only. We adopt the method of [4], with the following objective function:

$$f_k(k) = \sum_{\delta_*}\omega_*\|k*\delta_*x - \delta_*y\|^2 + \beta\|k\|^2, \tag{5}$$

where ω_* are as introduced in Sec. 4.1, and δ_* represent partial derivatives corresponding to $\mathbf{D}_*$. To speed up computation, FFT is used as derived in [4]. One important difference is that we set the gradients δ_*x outside of M to zero since we only allow edges to participate in the kernel estimation process.

5. Experimental Results

We first test our algorithm on the widely used 32-image test set introduced in [13], and further establish a synthetically blurred test set of 640 images of our own. Finally we show comparisons on blurred photographs with real unknown camera shakes.

To ensure fair comparison for evaluating estimated kernels from competing methods, we standardize the final non-blind deconvolution step by using sparse deconvolution[2] for Sec. 5.1, and the state-of-the-art method of Zoran and Weiss [20][3] for Sec. 5.2. Note that the use of [20] as the final step is a strict improvement compared to the default non-blind deconvolution step from all of the methods considered.

We make use of three measurements for quantitative analysis: mean PSNR, mean SSIM, and the geometric mean of error ratios, which is introduced in [13].

5.1. Existing Test Set from Levin *et al*.

The 32 test images in [13] were produced from 4 images and 8 different kernels. The blurred images and ground truth kernels were captured simultaneously by carefully controlling the camera shake and locking the Z-axis rotation handle of the tripod. We test our algorithm using both the natural and synthetic priors on this test set and compare with results provided by [14].

Fig. 5 shows a test image from this dataset deblurred by various methods. Note that kernels estimated by previous

[2]We use the MATLAB code provided by [14] at: http://www.wisdom.weizmann.ac.il/~levina/papers/LevinEtalCVPR2011Code.zip

[3]We use the MATLAB code from: http://www.cs.huji.ac.il/~daniez/epllcode.zip

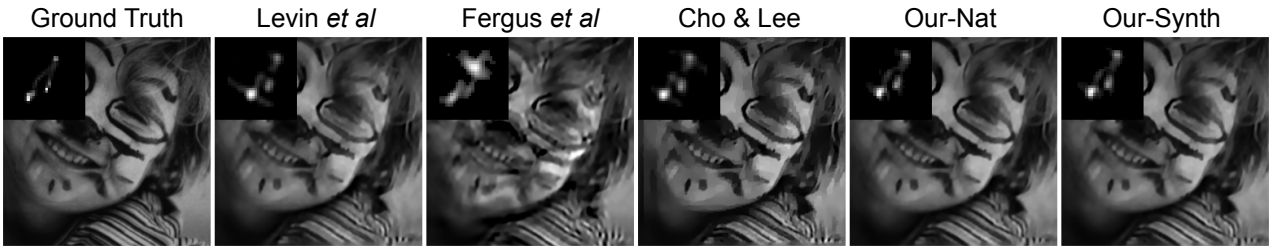

Figure 5. Comparison of results on one test image from [13]. Our kernels are less noisy and better resemble the ground truth (leftmost column). Sparse deconvolution with identical parameters are applied to all compared methods except Cho and Lee [4].

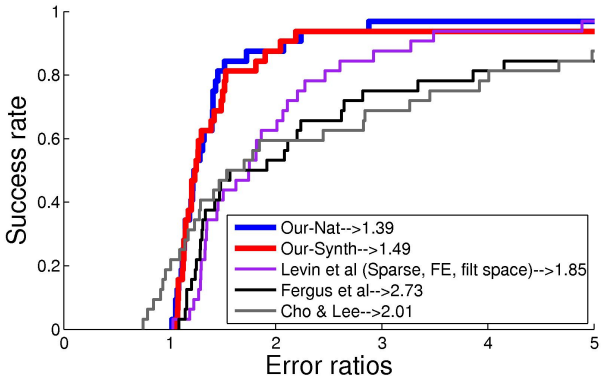

Figure 6. Performance comparison using the error ratio measure as in Levin *et al*. [13, 14]. The geometric mean of error ratios is shown in the legend for each algorithm. A ratio larger than 3 is deemed visually unacceptable, whereas a ratio less than 1 means the estimated kernel can do *better* than the ground truth kernel under the given sparse deconvolution method. It is worth mentioning that several estimated kernels from [4] appear better than the ground truth kernels. This could be due to the fact that Cho and Lee used a different deconvolution method to produce the final latent image x.

	PSNR	SSIM	Error Ratio
Known k	33.8197	0.9286	1.0000
Levin *et al*. [14]	31.1372	0.8960	1.8546
Fergus *et al*. [7]	29.4629	0.8451	2.7270
Cho & Lee [4]	30.7927	0.8837	2.0077
Our-Nat	**32.3842**	**0.9108**	**1.3917**
Our-Synth	32.1042	0.9033	1.4844

Table 1. Quantitative comparison for each method: mean PSNR, mean SSIM, and geometric mean for error ratio, computed over the 32 test images from [13, 14].

methods may contain trailing noise in certain directions. In contrast, our method produces a kernel that is most similar to the ground truth in shape, and the recovered latent images contain the least amount of artifacts.

In Fig. 6, we report the cumulative error ratio performance, which suggests that our method outperforms all competing methods on this test set. Finally, we aggregate

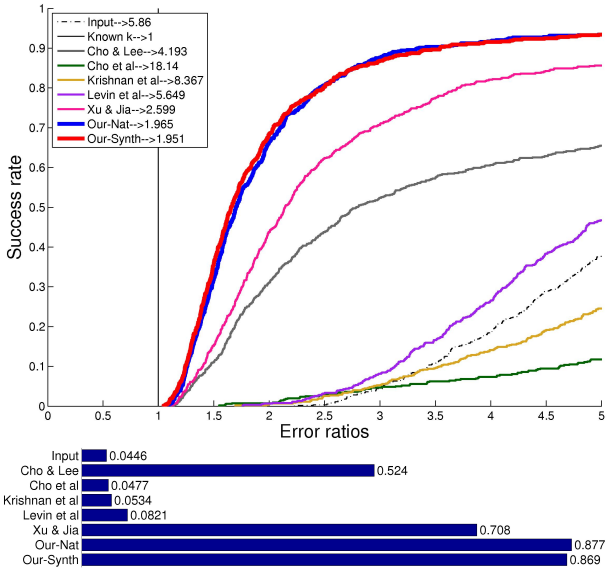

Figure 7. Performance on our synthetic test set of 640 images. Top: comparison of success rate vs error ratio for all competing methods. Our methods (thicker lines) significantly outperform all others on this test set. Bottom: a bar plot for the success rates at an error ratio of 3, which is deemed as the threshold for visually plausible deblurred results by Levin *et al*. [14].

performance over the 32 images and report the mean PSNR, mean SSIM and geometric mean error ratio for each method in Table 1.

It is interesting to note the level of saturation in performance: our method is able to achieve an error ratio of 2 for approximately 90% of the images, and the method of Cho and Lee [4] can produce several kernels that are *better* than the ground truth. The reasons are two-fold. First, the images are small and easy to deblur, because they contain plenty simple step edges, *e.g*., one of the images is an illustration and contains only clean and uninterrupted step edges. Second, the final non-blind deconvolution method–sparse deconvolution–can only achieve limited restoration performance in terms of PSNR. As a result, it's relatively easy to obtain low error ratios on these test images.

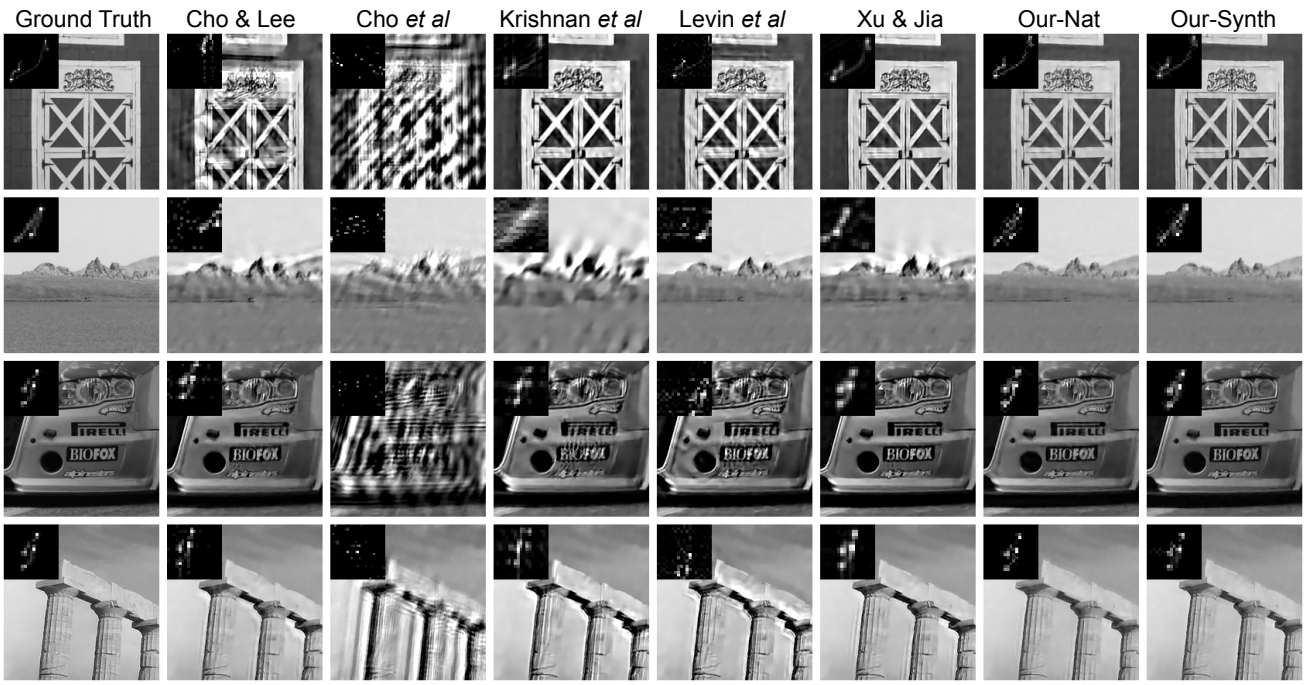

Figure 8. Qualitative comparison on four image regions from our synthetic data set. Note that previous methods tend to introduce oriented trailing noises in the estimated kernels, which could lead to low-frequency ringing artifacts in the deconvolved latent image x. The method of Zoran and Weiss [20] is used as the final non-blind step for all algorithms.

5.2. A New Synthetic Test Set of 640 Images

The 32 test images in [13] are only 255×255 in size, and limited in terms of diversity. To further test the limits of leading algorithms, and characterize the performance gap among them, we develop a synthetic test set of 640 high-resolution natural images of diverse scenes. We start with the 80 high quality natural images used in [17], and synthetically blur each of them with the 8 blur kernels from [13]. Finally, we add 1% additive Gaussian noise to the blurred images to model sensor noise. The same noise level is used in [10, 13, 20]. We present a comprehensive evaluation via both qualitative and quantitative comparisons for the methods from [4, 5, 14, 11, 19], as well as our results using the natural and synthetic priors.

Fig. 7 presents the cumulative distribution of error ratios. Our method outperforms all other methods, with [4, 19] being the most competitive. However, even with some parameter tuning, we are unable to obtain good results with the online MATLAB code packages from [14, 11, 5] on this test dataset. Fig. 8 shows a qualitative comparison of cropped results from different algorithms. Table 2 shows a quantitative evaluation of each method.

5.3. Deblurring Real Photographs

In Fig. 9 and Fig. 10 we show two comparisons on real photos with unknown camera shakes. Again for fair comparison we use different algorithms– [11, 4, 19] and ours–to estimate the kernel, followed by Zoran and Weiss [20] for the final non-blind deconvolution. However, such images often exhibit spatially varying blur, so a kernel might only explain (hence sharpen) some regions of the image. Overall our method is robust and generates results that are comparable to, if not better, than the state-of-the-art methods.

	PSNR	SSIM	Error Ratio
Input	24.7822	0.6429	5.8598
Known k	32.4610	0.8820	1.0000
Cho & Lee [4]	26.2353	0.8138	4.1934
Cho *et al*. [5]	20.1700	0.5453	18.1437
Krishnan *et al*. [11]	23.2158	0.7554	8.3673
Levin *et al*. [14]	24.9410	0.7952	5.6493
Xu & Jia [19]	28.3135	0.8492	2.5987
Our-Nat	29.5279	0.8533	1.9647
Our-Synth	**29.5585**	**0.8546**	**1.9510**

Table 2. Quantitative comparison on our synthetic test set of 640 images. Our method significantly outperforms existing methods.

6. Conclusion

We explore a new approach for kernel estimation from a single image via modeling image edge primitives using patch priors. Two type of priors are proposed, including a statistical prior learned from natural images, and a simple

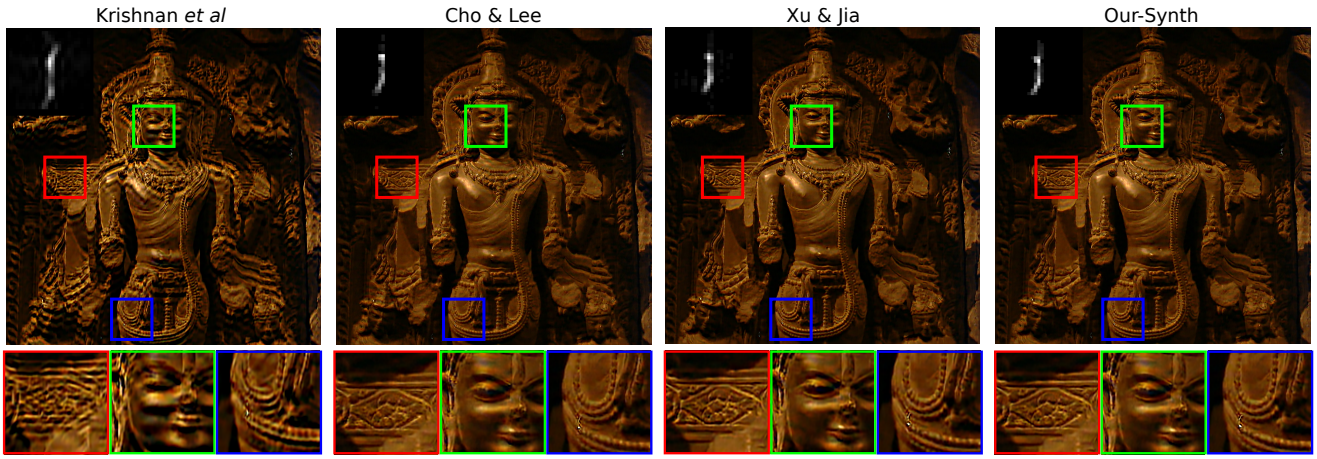

Figure 9. An example photos with unknown camera shake. Our method produces competitive results.

Figure 10. An example photo with unknown camera shake taken from Xu and Jia [19]. Our method produces sharper edges around the texts and less ringing in densely textured regions.

synthetic prior. We further show how to incorporate the priors into a deblurring objective to *significantly* improve the state-of-the-art performance. As future work, we would like to extend our image formation model to handle more severe noise and other outliers to make it more robust on low quality inputs. We would also like to improve the computational efficiency to make the system more practical.

7. Acknowledgement

James Hays is supported by NSF CAREER Award 1149853.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *TPAMI*, 2011.

[2] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *CVPR*, 2005.

[3] J. Chen, L. Yuan, C.-K. Tang, and L. Quan. Robust dual motion deblurring. In *CVPR*.

[4] S. Cho and S. Lee. Fast motion deblurring. *ACM Transactions on Graphics*, 28(5), 2009.

[5] T. S. Cho, S. Paris, B. K. P. Horn, and W. T. Freeman. Blur kernel estimation using the radon transform. In *CVPR*, 2011.

[6] M. David. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. 1982.

[7] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single photograph. *ACM Transactions on Graphics*, 25(3), 2006.

[8] N. Joshi, R. Szeliski, and D. J. Kriegman. PSF estimation using sharp edge prediction. In *CVPR*, 2008.

[9] R. Köhler, M. Hirsch, B. Mohler, B. Schölkopf, and S. Harmeling. Recording and playback of camera shake: benchmarking blind deconvolution with a real-world database. In *ECCV*, 2012.

[10] D. Krishnan and R. Fergus. Fast image deconvolution using hyper-laplacian priors. In *NIPS*, 2009.

[11] D. Krishnan, T. Tay, and R. Fergus. Blind deconvolution using a normalized sparsity measure. In *CVPR*, 2011.

[12] A. Levin, B. Nadler, F. Durand, and W. T. Freeman. Patch complexity, finite pixel correlations and optimal denoising. In *ECCV*, 2012.

[13] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *CVPR*, 2009.

[14] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Efficient marginal likelihood optimization in blind deconvolution. In *CVPR*, 2011.

[15] S. Roth and M. J. Black. Fields of experts. *IJCV*, 2009.

[16] Q. Shan, J. Jia, and A. Agarwala. High-quality motion deblurring from a single image. *ACM Transactions on Graphics*, 27(3), 2008.

[17] L. Sun and J. Hays. Super-resolution from internet-scale scene matching. In *ICCP*, 2012.

[18] Y. Weiss and W. T. Freeman. What makes a good model of natural images? In *CVPR*, 2007.

[19] L. Xu and J. Jia. Two-phase kernel estimation for robust motion deblurring. In *ECCV*, 2010.

[20] D. Zoran and Y. Weiss. From learning models of natural image patches to whole image restoration. In *ICCV*, 2011.