Deblurring Low-light Images with Light Streaks

Zhe Hu[1] Sunghyun Cho[2*] Jue Wang[2] Ming-Hsuan Yang[1]
[1] University of California, Merced [2] Adobe Research

Abstract

Images taken in low-light conditions with handheld cameras are often blurry due to the required long exposure time. Although significant progress has been made recently on image deblurring, state-of-the-art approaches often fail on low-light images, as these images do not contain a sufficient number of salient features that deblurring methods rely on. On the other hand, light streaks are common phenomena in low-light images that contain rich blur information, but have not been extensively explored in previous approaches. In this work, we propose a new method that utilizes light streaks to help deblur low-light images. We introduce a non-linear blur model that explicitly models light streaks and their underlying light sources, and poses them as constraints for estimating the blur kernel in an optimization framework. Our method also automatically detects useful light streaks in the input image. Experimental results show that our approach obtains good results on challenging real-world examples that no other methods could achieve before.

Figure 1. Deblurring low-light images. (a) A real example. (b) A cropped region from (a). (c-e) Cropped deblurring results of Cho and Lee [2], Xu and Jia [19], and our approach, respectively. Please refer to the supplementary material for full resolution images and complete results.

1. Introduction

Taking good pictures in low-light conditions such as at night or in a dark room is perhaps the most challenging task for non-professional photographers. Since a longer exposure time is often required in these cases to produce a well-lit image, the captured image using a handheld camera often ends up to be blurry due to inevitable camera shake. It is thus highly desirable to apply image deblurring techniques to recover a sharp image from a blurry low-light image.

Although single image deblurring techniques have been largely advanced in recent years [4, 20, 15, 2, 13, 11, 1, 19], the state-of-the-art methods often have difficulties to handle low-light images. This is because most recent approaches rely on salient image features such as edges [2, 9, 7, 19] for blur kernel estimation, however in low-light images the amount of salient image features that can be extracted is often limited, as shown in examples in Fig. 1. Furthermore, low-light images often have gone through heavy in-camera non-linear tone mapping, which breaks the linear blur model that most approaches assume [16].

In this paper, we examine a special phenomenon that often appears in low-light images for helping image deblurring: *light streaks*. Light streaks are caused by blurred light sources such as light bulbs, flash lights, reflected lights, etc., which are very common in both natural (e.g., stars in the sky) and man-made scenes (e.g., street lights). Given that these light sources are small, high-intensity objects, the light streaks they create roughly depict the shapes of the underlying blur kernels. Intuitively, light streaks contain rich blur information that could potentially help us achieve better deblurring results.

However, without proper treatment, the presence of light streaks can actually jeopardize the performance of existing deblurring approaches and prevent them from estimating a correct blur kernel. The reasons are twofold. Firstly, many blur kernel estimation methods extract and use salient edges

*The second author is now with Samsung Electronics.

for blur kernel estimation [2, 9, 18], but light streaks often contain sharp, strong edges that may be mis-treated as structural edges by these algorithms. Secondly, light streaks often contain saturated pixels. As shown in [3], without proper handling of saturated pixels, deconvolution of light streak pixels may result in severe ringing artifacts, causing problems for both blur kernel estimation and producing a high quality final output.

In this work, we propose a new deblurring framework that properly uses light streaks as an additional cue for blur kernel estimation. We extend the widely-used linear blur model by explicitly modeling point light sources and light streaks, resulting in a non-linear model that more accurately describes the formation of low-light images that contain light streaks. We then formulate a kernel estimation energy function that takes into account light streaks as well as other image structures. Our method also automatically detects "good" light streaks that are useful for kernel estimation. After the blur kernel is estimated, the final output image is obtained by a regularized Richardson-Lucy deconvolution with outlier handling to suppress ringing artifacts.

There has been only limited work that considers light streaks for image deblurring. Regarding blur kernel estimation, Harmeling et al. [6] and Whyte et al. [17] discard saturated pixels and estimate the blur kernel using remaining pixels, but they do not use light streaks as a cue. To suppress artifacts caused by saturated pixels in non-blind deconvolution, Cho et al. [3] and Whyte et al. [17] explicitly model saturated pixels in their optimization processes, which alternatingly detect saturated pixels and estimate the latent image using non-saturated ones. The most related work to ours is an interactive deblurring method proposed by Hua and Low [8], in which the user needs to manually select a light streak region, and the system applies image processing operations in it for extracting the kernel. Since no other image structures are used, the blur kernel generated from a small cropped region may not be optimal for the entire image. In contrast, our method not only automatically detects useful light streaks, but also incorporates them into a more principled optimization process for estimating more accurate blur kernels.

2. Light Streak Detection

We now describe how to detect image patches which contain "good" light streaks for kernel estimation, based on pre-defined properties. It is a three-step approach: we first detect a set of candidate image patches that potentially contain light streaks; we then select the light streak that is most similar to the underlying blur kernel using a power-spectrum-based metric; finally, we use the selected best light streak to find additional good light streak patches for kernel estimation.

The motivation for extracting multiple light streaks is that a single light streak may be fully or partially saturated, thus it may contain only limited information about the blur kernel. By using multiple light streaks, we can cumulatively extract more blur kernel information from them. Furthermore, using multiple light streaks from different parts of the image helps the system achieve better stability against image noise and local distortions.

2.1. Candidate patch detection

Our method first identifies a set of image patches that may potentially contain light streaks. Given the physical nature of light streaks, we define the following properties as the appearance priors for good light streak patches: (1) pixels covered by a light streak should have relatively high intensities and those on the background have relatively low intensities in a local neighborhood; (2) the high intensity pixels in a light streak patch should have a very sparse distribution; (3) the light streak should be located near the center of a patch; and (4) there should be no other image structures in a light streak patch.

Since the goal of this step is merely removing irrelevant image patches from being considered in the following steps for reducing computational cost, we apply a set of heuristic filters to quickly achieve it. Firstly, we apply two thresholds on maximum image intensity and gradient magnitude to filter out dark and flat patches. The thresholds are set adaptively based on global statistics, i.e., top 10% pixels are above the thresholds. According to Property 2, we discard those patches that contains too many high intensity pixels (more than 15%). According to Properties 3 and 4, we divide each patch into two regions: the center region whose size is half of the original patch, and the border region. We then compute how many pixels have either high intensity or high gradient magnitude (above the thresholds) in the border region, and normalize it by the number computed from the center region. If the ratio is higher than 30%, we discard the patch. In this way we can quickly rule out most irrelevant patches and generate only a reasonable number of candidate patches for more careful light streak detection.

2.2. Finding the best light streak patch

From the set of candidate light streak patches, we then carefully find the one that best resembles the underlying blur kernel. Intuitively, the best light streak patch should contain a well-lit light trajectory that has roughly the same shape as the blur kernel, on a relatively clean background. This requires the underlying light source to be well in-focus and to have a small size, and to be well separated from other image structures.

Unfortunately the blur kernel is unknown so we cannot directly use it for selecting the best light streak. Recent work [5] shows that we can get a good approximation of the power spectrum of the blur kernel directly from the in-

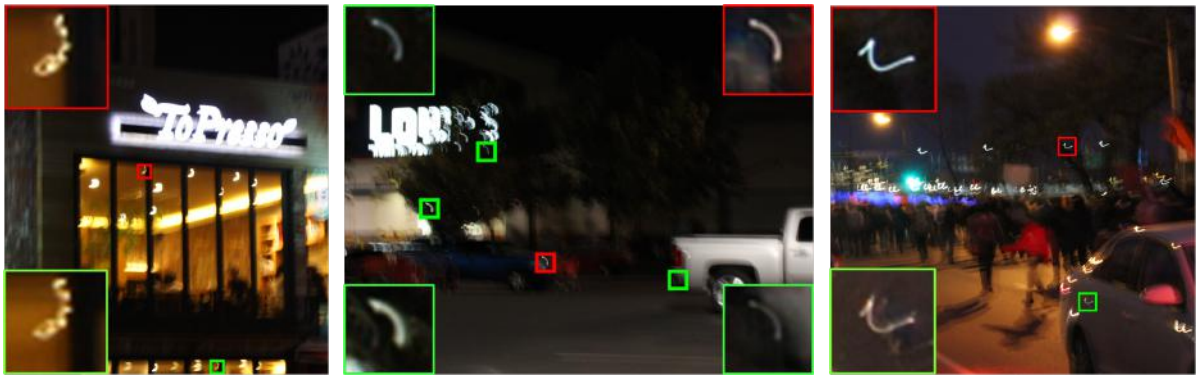


Figure 2. Examples of light streak detection. The red box indicates the best light streak patch and the green boxes show additional light streak patches that are automatically identified.

put image. We thus define a power-spectrum-based metric for selecting the best light streak.

Specifically, the power-law of natural images shows:

$$|\widehat{I}(\omega)|^2 \propto \|\omega\|^{-\beta}, \tag{1}$$

where $\widehat{I}$ is the Fourier transform of an image I, ω is the coordinate in the frequency domain and $\beta \approx 2$. It is well known that a Laplacian filter is a good approximation to $\|\omega\|^{-\beta}$, so that:

$$|\widehat{I}(\omega)|^2 |\widehat{L}(\omega)| \approx C, \tag{2}$$

where L is a Laplacian filter, and C is a constant. For a blurred image $B = K * I + N$, where K is a blur kernel, N is noise and $*$ is a convolution operator, we have:

$$|\widehat{B}(\omega)|^2 |\widehat{L}(\omega)| \approx |\widehat{I}(\omega)|^2 |\widehat{K}(\omega)|^2 |\widehat{L}(\omega)| \approx C|\widehat{K}(\omega)|^2. \tag{3}$$

In the spatial domain, we have $B \otimes B * L \approx C(K \otimes K)$, where $\otimes$ is a correlation operator. We thus define our metric as:

$$d(P, B) = \min_C \|B \otimes B * L - C(P \otimes P)\|^2, \tag{4}$$

where P is a candidate light streak patch. The optimal C can be found by solving a least square problem. Among all the candidate patches, we select the one with the smallest distance as the best light streak patch, $P_0 = \text{argmin}_P\, d(P, B)$. Note that this method also naturally favors unsaturated light streaks, as saturated ones would result in larger metric distances. In Fig. 2, we show some examples of the best light streak patch selected using this method.

2.3. Finding additional light streak patches

Next, we use the selected best light streak patch to find additional light streak patches from the initial candidate set. This is done by computing the Euclidean distance between the candidate patch P and the best patch P_0. A histogram equalization operation has been applied to all patches before computing the distances to account for the intensity difference between different light streaks. By setting a threshold, we define a set of detected light streak patches $\mathbb{P} = \{P_i\}_{i=1}^{N_P}$. We use a threshold $0.13 S_P \max(P_0)$ where S_P is the patch size in our implementation. Fig. 2 shows some examples of detected light streak patches.

3. Blur Kernel Estimation using Light Streaks

We now describe how to estimate the blur kernel using the detected light streaks as well as other image structures. The traditional linear motion blur model is formulated as $B = K * I + N$. This simple formulation however is insufficient to model the behavior of light streaks. To separately handle image structures and light streak patches, we divide the pixels in the observed image B into three complementary sets B^p, B^r and B^s, which are defined as $B^p = \bigcup P_i$, $B^r = \{x | B(x) \text{ is not saturated} \wedge x \notin P_i\ \forall i\}$ and $B^s = \{x | B(x) \text{ is saturated} \wedge x \notin P_i\ \forall i\}$. We assign each $B^\star$ a binary mask $M^\star$ so that $B^\star = M^\star \cdot B$. Here $\cdot$ denotes per-pixel multiplication. The three sets can be easily determined after the light steak detection step, before kernel estimation, and they remain fixed in later steps.

We then introduce a more accurate, nonlinear blur model for the input image as:

$$\begin{cases} B^p &= \sum P_i \\ B^r &= M^r \cdot (K * I) + N \\ B^s &= M^s \cdot c(K * I + N) \end{cases}, \tag{5}$$

where c is a clipping function defined as $c(v) = v$ if v is in the dynamic range of the camera sensor, and $c(v) = 0$ or 1 otherwise[1].

[1] In this paper, we use the dynamic range normalized into [0,1].

We use $\hat{P}_i$ to denote the unclipped light streaks such that $P_i = c(\hat{P}_i)$. We also introduce auxiliary variables D_i, for each detected light streak patch P_i, which describe the appearance of the original point light sources that produced the light streaks. We further assume each point light source has a disk shape, but may have a different size and at a different intensity value. In our method, these variables are estimated as well. Specifically, each light streak is modeled as:

$$\hat{P}_i = K * D_i + N. \tag{6}$$

Thus, the first line in Eq. (5) becomes

$$B^p = \sum c(\hat{P}_i) = \sum c(K * D_i + N). \tag{7}$$

Given the above model, we seek the best K, D_i and I that can best describe the observed image and detected light streaks. This is done by adopting the widely-used alternating optimization approach: given the initial values of the three variables, we fix two of them at each time and optimize the one that is left.

3.1. Updating K

In this step, we fix D_i and I, and update K by optimizing the following energy function:

$$\begin{aligned} f_K(K) \quad = \quad & \sum_{x \in M^r} |\partial_h B(x) - (K * P_h)(x)|^2 \\ & + \sum_{x \in M^r} |\partial_v B(x) - (K * P_v)(x)|^2 + \lambda \|K\|_1 \\ & + \mu \sum_{P_i \in \mathbb{P}} \sum_{x \in P_i} |(D_i * K)(x) - \hat{P}_i(x)|^2, \end{aligned} \tag{8}$$

where x is the pixel index. The first two terms on the right side are data terms based on the prediction scheme proposed by Cho and Lee [2]. ∂_h and ∂_v are partial differential operators along the horizontal and vertical axes, respectively, and P_h and P_v are predicted gradient maps along the two axes, respectively. Please refer to [2] for more details on how to compute P_h and P_v using image filtering techniques. Note that the first two terms are applied only on pixels in M^r, so that they are affected by neither saturated pixels nor light streaks. This is in sharp contrast to previous edge-based approaches [2, 19] where edge extraction is inevitably affected by light streaks and saturated regions. The third term is a prior on K. The last term is derived from Eq. (6).

To obtain the underlying light streak $\hat{P}_i$, we replace the saturated pixels in P_i with the interpolated pixels using spline interpolation (explained in more detail in Sec. 3.4). The computation of $\hat{P}_i$ is pre-defined before the optimization process. Then the energy in Eq. (8) is minimized using an iterative reweighted least square method (IRLS). Here we set μ as $S_I/(S_P N_P)$ at the beginning, where S_I is the image size. We reduce μ with a factor 0.75 over iterations to rely more on the data error term. We set λ as $\sigma^2 S_P^2/50^2$ with σ denoting the noise deviation of the Gaussian prior.

3.2. Updating D_i

In this step, for each selected light streak patch, we estimate its original unblurred light source. As mentioned earlier, we assume that the original point light D_i has a disk shape, and its size and intensity can vary. Thus, we model D_i as a function of two parameters t_i and r_i, which denote the intensity value and the radius of the disk, respectively. Note that t_i is not restricted to the dynamic range of the image. We then derive an energy function for this step as:

$$\begin{aligned} f_{D_i}(t_i, r_i) \quad = \quad & \|D_i(t_i, r_i) * K - \hat{P}_i\|^2 \\ & + \|D_i(t_i, r_i) - I_i\|^2, \end{aligned} \tag{9}$$

where I_i is the patch in the latent image I covering the same pixels as P_i. Since we have a strong prior knowledge about the light sources, e.g., they are usually very small and have high intensities, we thus sample a discrete set of possible t_i and r_i values, and find out the optimal one that minimizes $f_{D_i}(t_i, r_i)$. In practice, we find that this exhaustive search works well in both synthetic and real examples.

3.3. Updating I

In this step, we update the latent image I using the updated blur kernel K and light sources D_i by optimizing the following energy function:

$$\begin{aligned} f_I(I) \quad = \quad & \sum_i \mu \|D_i - I_i\|^2 + \sum_x |B(x) - c(K * I)(x)|^2 \\ & + \gamma \sum_x \left(|\partial_h I(x)|^\alpha + |\partial_v I(x)|^\alpha \right), \end{aligned} \tag{10}$$

where the third term is the sparse prior term proposed in [12] and $\alpha = 0.8$. γ is a weight for the regularization terms and we use $\gamma = 0.005$ in our experiments. We solve Eq. (10) using an IRLS method.

3.4. Initialization and implementation details

To compute $\hat{P}_i$ in Eq. (8), we calculate the 1D interpolation along horizontal and vertical axes using the Matlab function **interp1** and compute the average to replace the saturated pixels. For updating D_i, we sample a discrete set $\{100/255, 120/255, 140/255, \ldots, 1000/255\}$ for t_i and $\{1, 2, 3\}$ for r_i. For the update of I, we refer the readers to [3] for parameter settings.

Our system performs kernel estimation in the original resolution without the coarse-to-fine strategy, and it usually takes 5-10 iterations to converge. In the first iteration, we compute K by taking out the first and second terms in Eq. (8) and only considering the best detected light streak patch as:

$$\underset{K}{\operatorname{argmin}} \lambda \|K\|_1 + \mu \sum_{x \in P_1} |(D_1 * K)(x) - \hat{P}_1(x)|^2, \tag{11}$$

where we initialize D_i as the point light source with $r_i = 1$ and $t_i = \max \hat{P}_i$. Given the initial K and D_i, we then can compute the initial I, and iteratively update all three.

4. Deconvolution with Ringing Suppression

After the blur kernel is estimated, we use a non-blind deconvolution method to restore the latent image. As images taken in low-light conditions often contain lots of saturated pixels, they should be handled properly to avoid generating severe ringing artifacts.

There are some recent approaches for handling saturated pixels in non-blind deconvolution. Cho et al. [3] propose a blur model that explicitly models outliers including saturated pixels, and use an expectation-maximization (EM) method for generating the final image. Whyte et al. [17] propose a modified Richardson-Lucy (RL) method based on a blur model with a saturation function. While both methods use similar blur models, each has its own pros and cons. For example, Cho et al.'s method can handle other types of outliers, while Whyte et al.'s method provides an additional ringing prevention scheme. In our work, we derive a new deconvolution algorithm that combines the advantages of these two methods.

Our approach adopts the basic framework of RL deconvolution, because in practice we found that RL deconvolution is often more effective at suppressing ringing artifacts. In RL-deconvolution, the latent image I is estimated by maximizing a likelihood $p(B|K, I)$, which is defined using Poisson distributions. The update equation of RL deconvolution is derived by differentiating the log-likelihood with respect to I:

$$I^{t+1} = I^t \cdot K \otimes \frac{B}{I^t * K}, \tag{12}$$

where $\cdot$ and $\otimes$ mean per-pixel multiplication and correlation operations, respectively. Division is also pixel-wise.

To better handle outliers and saturated pixels, following Cho et al.'s work [3], we formulate non-blind deconvolution as a MAP problem:

$$p(I|B, K) \propto \sum_{M \in \mathbb{M}} p(B|M, K, I)p(M|K, I)p(I), \tag{13}$$

where M is a mask for specifying inliers and outliers, i.e., $M(x) = 1$ if $B(x)$ is an inlier, and $M(x) = 0$ if $B(x)$ is an outlier. $\mathbb{M}$ is a set of all possible M. We assume Poisson distributions for inliers and uniform distributions for outliers. Then, the likelihood term $p(B|M, K, I)$ is defined as $P(B(x)|M, K, I) = \mathcal{P}(B(x)|K * I(x))$ if $M(x) = 1$ and $P(B(x)|M, K, I) = C$ otherwise, where $\mathcal{P}$ is a Poisson distribution, and C is a constant defined as the inverse of the width of the dynamic range. We define $P(M|K, I)$ and $P(I)$ in the same way to Cho et al.'s approach, as detailed in [3].

Given above formulations, we derive an EM-based regularized RL deconvolution method. The E-step computes per-pixel weights W^t at t-th iteration as:

$$W^t = \frac{\mathcal{P}(B|K * I^t)P_{\text{in}}}{\mathcal{P}(B|K * I^t)P_{\text{in}} + C(1 - P_{\text{in}})}, \tag{14}$$

where $P_{\text{in}} \in [0, 1]$ is the probability that B_x is an inlier. The M-step updates the latent image I as:

$$I^{t+1} = \frac{I^t}{1 + \lambda\rho(I^t)} \cdot K \otimes \left(\frac{B \cdot W^t}{I^t * K} + 1 - W^t \right), \tag{15}$$

where $\rho(I)$ is the derivative of a sparse prior:

$$\rho(I) = \text{sign}(\partial_h I)\alpha|\partial_h I^t|^{\alpha-1} + \text{sign}(\partial_v I)\alpha|\partial_v I^t|^{\alpha-1}. \tag{16}$$

We set $\alpha = 0.8$. Then deconvolution is done by alternatingly solving Eq. (14) and Eq. (15).

In our implementation, for better computational efficiency we approximate Eq. (14) using Gaussian distributions. Then, Eq. (14) becomes:

$$W^t = \frac{\mathcal{N}(B|K * I^t, K * I^t)P_{\text{in}}}{\mathcal{N}(B|K * I^t, K * I^t)P_{\text{in}} + C(1 - P_{\text{in}})}. \tag{17}$$

We typically run 40 iterations to obtain the deconvolution result. To further suppress ringing artifacts, we also adopt the ringing prevention scheme in [17], which decomposes an image into unsaturated and saturated regions and performs deconvolution separately. Fig. 3 shows an example of our non-blind deconvolution.

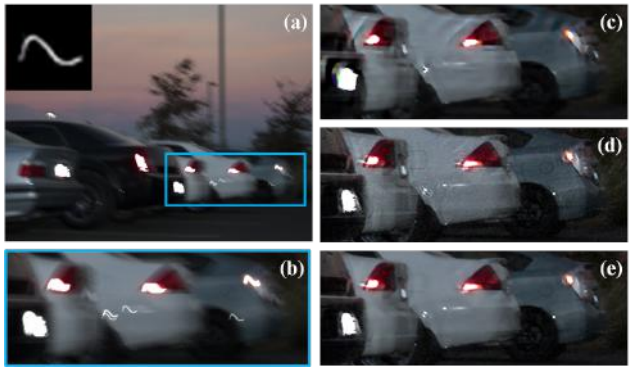


Figure 3. Non-blind deconvolution of a synthetic example. (a) Input image with the ground truth blur kernel. (b) A cropped region from (a). (c)-(e) Cropped deconvolution results of Cho et al. [3], Whyte et al. [17] and our approach, respectively. Please refer to the supplementary material for complete results.

5. Experimental Results

We implement our method in MATLAB and conduct experiments on a PC with an 1.73 GHz Core i7 CPU and 8

Table 1. Quantitative comparisons using kernel similarity (KS).

	[2]	[11]	[14]	[19]	Ours
KS	0.5323	0.5449	0.5298	0.5312	**0.7069**

GB RAM. For an image of size 700×1000, the light streak detection step takes about 2 seconds, and the kernel estimation step takes around 5 minutes with our unoptimized implementation. Due to limited space, we only show a few examples in this section, and more results can be found in the supplementary material.

To evaluate the effectiveness of the proposed light streak detection method on real-world images, we collect 40 natural low-light images that contain light streaks, and apply our method to them. We then visually examine the extracted light streak patches in each image to see if the selected best light streak patches really contain light streaks. We find that in 35 out of 40 images (87.5%), our method successfully extract correct light streaks. Fig. 2 shows examples of our light streak detection.

5.1. Comparisons with Hua and Low's method

We first compare our method with Hua and Low's approach [8], which estimates the blur kernel from a manually-selected light streak patch. Since this method only uses a single light streak patch, for a fair comparison we also limit our system to select the same patch. In this experiment, to test the robustness of the two approaches, for each input image we tried two different light streaks: the best patch suggested by our method, and a manually selected patch that is visually obvious to the user, resulting in two deblurring results for each method. Fig. 4 shows the results on one example.

The results suggest that Hua and Low's method are sensitive to the input patch: it works reasonably well with the automatically-detected, non-saturated light streak, but fails badly with the other one. This is because it only relies on the light-streak patch for extracting the blur kernel, thus it is sensitive to saturation which causes information loss. On the contrary, our method is more robust and works well even with the saturated patch, as we also rely on other image structures for kernel estimation. Furthermore, with the non-saturated patch, our method generates higher quality results than Hua and Low's method due to the proposed optimization scheme. The results also demonstrate that our automatic selection method could select good light-streak patches for deblurring. More comparisons with Hua and Low's method on real examples are shown in Fig. 5.

5.2. Comparisons on real images

Here we qualitatively compare our method with the state-of-the-art general image deblurring methods [2, 19] that performed well on the recent benchmark test [10], on some real-world examples. As shown in Fig. 5, previous methods performed poorly on these low-light images since they do not contain sufficient salient edges for kernel estimation, and their estimated kernels tend to be close to a delta function. In contrast, our method obtained more accurate kernel estimation and produced higher quality outputs.

5.3. Comparisons on synthetic images

We also prepare a synthetic dataset to quantitatively compare our method against others. To build the dataset, we first capture 11 low-light images in RAW format covering a variety of scenes, using a Canon Rebel Xsi camera with an EF-S 18-55 mm lens. For each image we stretch the pixel intensities that beyond a certain value and then apply 14 different blur kernels, followed by adding Gaussian noise with 1% variation and clipping to dynamic range [0,1]. This gives us 154 test images in total.

Fig. 7 shows some representative examples of this dataset, as well as the deblurred results using different algorithms. For a fair comparison we use the non-blind deconvolution method described in Sec. 4 for all methods. The results suggest that our results have significantly higher visual quality than those generated by other methods, due to more accurate kernel estimation.

For quantitative evaluation, we use the error-ratio histogram originally proposed in [13], and the results are shown in Fig. 6. We also compute the average kernel similarity [7] of each estimated kernel, and the results are shown in Table 1. These results demonstrate that the proposed method outperforms previous general image deblurring approaches on deblurring low-light images, due to the explicit light streak modeling.

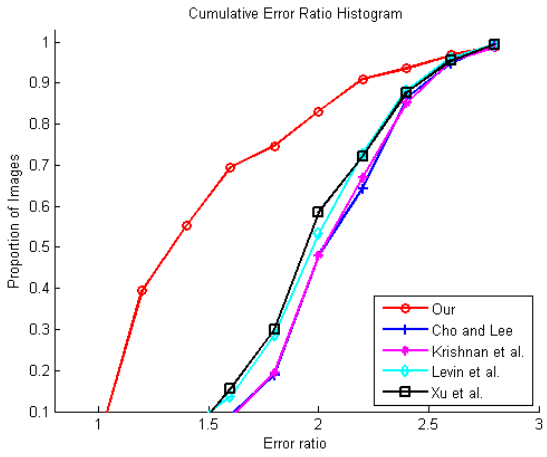


Figure 6. Cumulative error ratio histogram on the synthetic dataset.

5.4. Failure cases

The proposed method also fails in some cases. One scenario is when the underlying light sources of light streaks

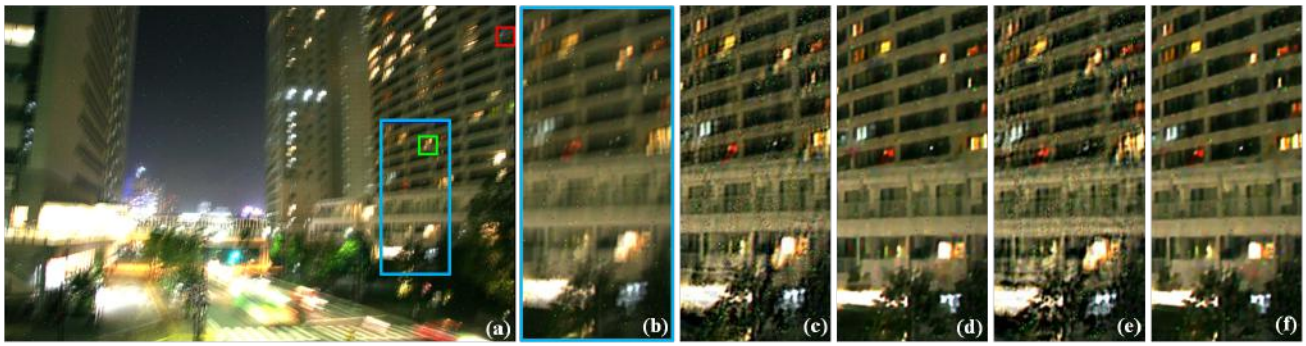


Figure 4. A comparison with Hua and Low's approach [8]. The red box indicates our selected light streak patch and the green box is a manually selected patch. (a) Input image; (b) cropped regions from (a); (c) & (e) cropped results by Hua and Low [8], using the red and green light streak patches, respectively; (d) & (f) cropped results by our method. Please refer to the supplementary material for complete results.

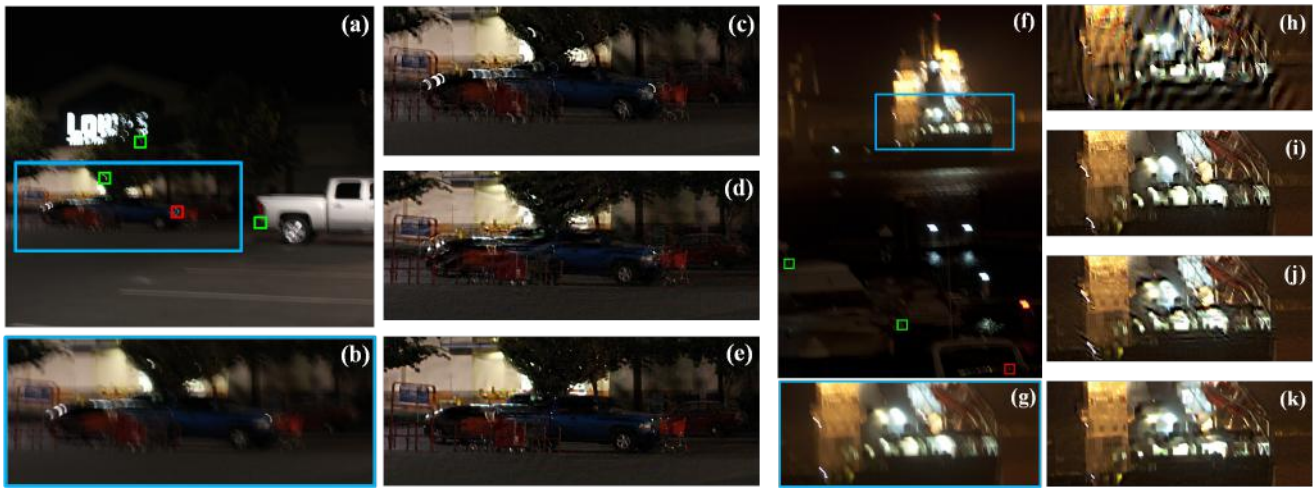


Figure 5. Comparisons with state-of-the-art methods on real examples. (a)(f) Input image; (b)(g) cropped regions from (a) and (f); (h) cropped results by Cho and Lee [2]; (c)(i) cropped results by Krishnan et al. [11]; (d)(j) cropped results by Hua and Low [8]; (e)(k) cropped results of our approach. Please refer to the supplementary material for complete results.

are large and no longer point lights, as shown in Fig. 8(a). This type of light streaks may be selected by our algorithm and could lead to an erroneous kernel estimation. This problem can be partially alleviated with user guidance by manually selecting a better patch, if it is available in the image.

Another difficulty is non-blind deconvolution on a large saturated region as shown in Fig. 8(b). For this synthetic example, even if we supply the ground truth blur kernel, the proposed method and other non-blind deconvolution methods can not generate satisfactory results due to severe information loss by intensity clipping, as shown in Fig. 8(c).

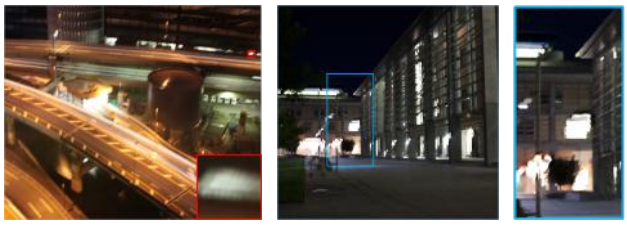
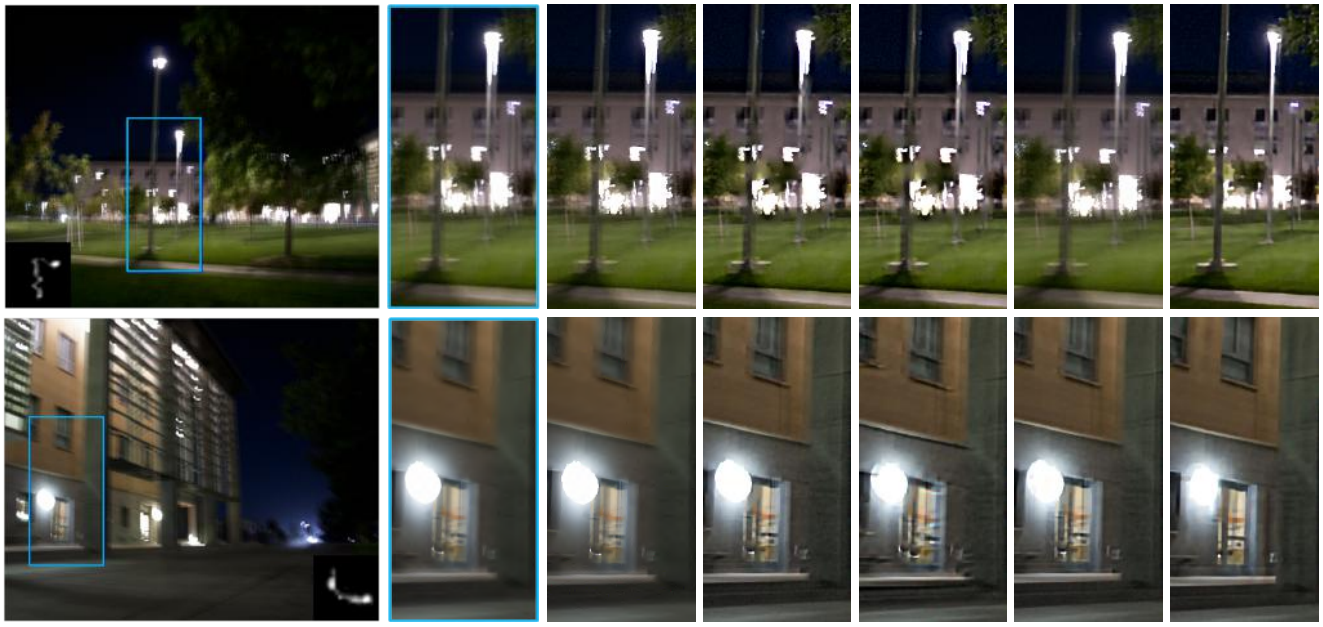


Figure 8. Examples of failure cases. (a) a blurry image with failed light streak detection; (b) a deblurred image using the ground truth kernel; (c) a cropped region of (b).

6. Conclusion

In this paper we propose a new deblurring method that explicitly models light streaks for low-light image deblurring. Our method detects the light streaks appearing in the blurry image and incorporates them into an optimization framework, which jointly considers light streaks and other image structures for kernel estimation. We also carefully



(a) (b) (c) (d) (e) (f) (g)

Figure 7. Comparisons on synthetic examples. (a) Input image with ground truth kernel; (b) cropped region from (a); (c)-(g) cropped results by Cho and Lee [2], Levin et al. [14], Krishnan et al. [11], Xu et al. [19] and our approach, respectively. Please refer to the supplementary material for complete results.

suppress the ringing artifacts caused by light streaks in the non-blind deconvolution step. Experimental results show that our system can obtain decent results on the challenging images that previous methods can not handle.

Acknowledgements

This work was done when the first author was an intern at Adobe Research. This research is partially supported by NSF CAREER Grant #1149783, NSF IIS Grant #1152576 and gift from Adobe.

References

[1] J.-F. Cai, H. Ji, C. Liu, and Z. Shen. Framelet-based blind motion deblurring from a single image. *TIP*, 21(2):562–572, 2012. 1

[2] S. Cho and S. Lee. Fast motion deblurring. In *SIGGRAPH Asia*, 2009. 1, 2, 4, 6, 7, 8

[3] S. Cho, J. Wang, and S. Lee. Handling outliers in non-blind image deconvolution. In *ICCV*, 2011. 2, 4, 5

[4] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single photograph. In *SIGGRAPH*, pages 787–794, 2006. 1

[5] A. Goldstein and R. Fattal. Blur-kernel estimation from spectral irregularities. In *ECCV*, 2012. 2

[6] S. Harmeling, S. Sra, M. Hirsch, and B. Schölkof. Multiframe blind deconvolution, super-resolution, and saturation correction via incremental EM. In *ICIP*, 2010. 2

[7] Z. Hu and M.-H. Yang. Good regions to deblur. In *ECCV*, 2012. 1, 6

[8] B.-S. Hua and K.-L. Low. Interactive motion deblurring using light streaks. In *ICIP*, 2011. 2, 6, 7

[9] N. Joshi, R. Szeliski, and D. J. Kriegman. PSF estimation using sharp edge prediction. In *CVPR*, 2008. 1, 2

[10] 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*, pages 27–40, 2012. 6

[11] D. Krishnan, T. Tay, and R. Fergus. Blind deconvolution using a normalized sparsity measure. In *CVPR*, pages 233–240, 2011. 1, 6, 7, 8

[12] A. Levin, R. Fergus, F. Durand, and W. T. Freeman. Image and depth from a conventional camera with a coded aperture. In *SIGGRAPH*, 2007. 4

[13] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *CVPR*, pages 1964–1971, 2009. 1, 6

[14] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Efficient marginal likelihood optimization in blind deconvolution. In *CVPR*, pages 2657–2664, 2011. 6, 8

[15] Q. Shan, J. Jia, and A. Agarwala. High-quality motion deblurring from a single image. In *SIGGRAPH*, 2008. 1

[16] Y.-W. Tai, X. Chen, S. Kim, S. J. Kim, F. Li, J. Yang, J. Yu, Y. Matsushita, and M. S. Brown. Nonlinear camera response functions and image deblurring: Theoretical analysis and practice. *TPAMI*, 35(10):2498–2512, 2013. 1

[17] O. Whyte, J. Sivic, and A. Zisserman. Deblurring shaken and partially saturated images. In *ICCVWorkshops*, pages 745–752, 2011. 2, 5

[18] L. Xu and J. Jia. Two-phase kernel estimation for robust motion deblurring. In *ECCV*, 2010. 2

[19] L. Xu, S. Zheng, and J. Jia. Unnatural L0 sparse representation for natural image deblurring. In *CVPR*, 2013. 1, 4, 6, 8

[20] L. Yuan, J. Sun, L. Quan, and H. Shum. Image deblurring with blurred/noisy image pairs. In *SIGGRAPH*, 2007. 1