Unnatural L_0 Sparse Representation for Natural Image Deblurring

Li Xu Shicheng Zheng Jiaya Jia
The Chinese University of Hong Kong
http://www.cse.cuhk.edu.hk/leojia/projects/l0deblur/

Abstract

We show in this paper that the success of previous maximum a posterior (MAP) based blur removal methods partly stems from their respective intermediate steps, which implicitly or explicitly create an unnatural representation containing salient image structures. We propose a generalized and mathematically sound L_0 sparse expression, together with a new effective method, for motion deblurring. Our system does not require extra filtering during optimization and demonstrates fast energy decreasing, making a small number of iterations enough for convergence. It also provides a unified framework for both uniform and non-uniform motion deblurring. We extensively validate our method and show comparison with other approaches with respect to convergence speed, running time, and result quality.

1. Introduction

Single-image motion deblurring, a.k.a. blind deconvolution, was extensively studied in these a few years and has achieved great success with a few milestone solutions. Because naive maximum a posterior (MAP) inference could fail on natural images, state-of-the-art methods either maximize marginalized distributions [5, 17, 18, 6] or propose novel techniques in MAP to effectively avoid trivial delta kernel estimates [11, 20, 3, 25, 4, 10].

The set of effective techniques that reinvigorate MAP [20, 3, 25] can produce high-quality results in seconds and were broadly adopted in a number of applications. They also form basic steps for spatially-variant deblurring. The particularly useful techniques include adaption of the energy function during optimization [20], explicit sharp edge pursuit [11, 13, 19, 3, 25, 4, 10], edge selection [25], and employment of normalized sparsity measure [16]. These methods achieve efficient inference with their distinct formulation and optimization steps, discussed below.

1.1. Analysis

Prior MAP-based approaches can be roughly categorized into two groups, i.e., methods with *explicit* edge prediction

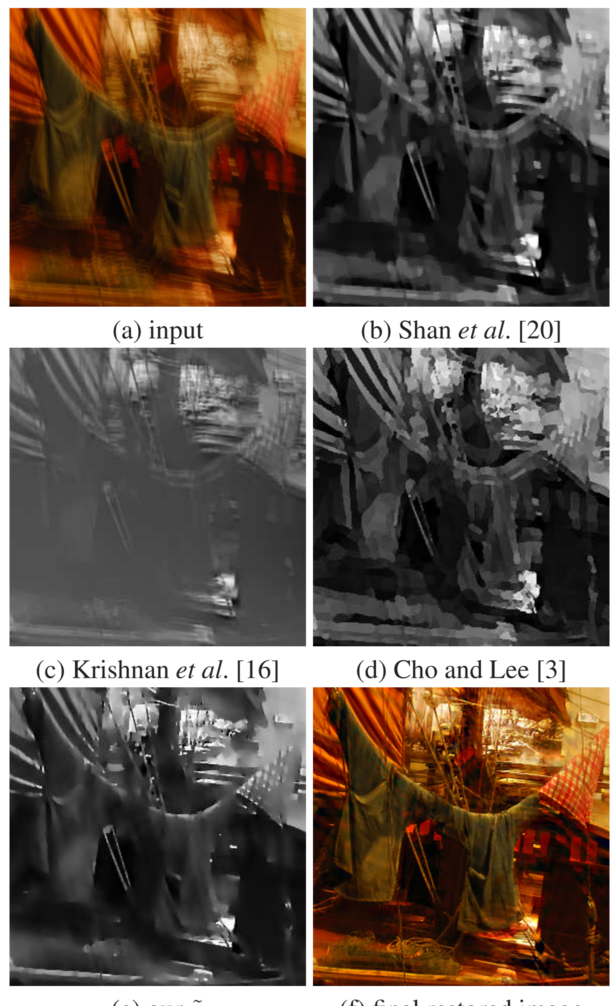


Figure 1. Intermediate unnatural image representation exists in many state-of-the-art approaches.

[11, 13, 19, 3, 25, 9, 23], which are referred to as semi-blind schemes, and those [20, 16] *implicitly* incorporating special regularization to remove detrimental structures in early stages and gradually enrich image details in iterations.

We found representative approaches in these two respective groups share the commonness in the middle of the procedure to generate one or multiple special maps only containing salient image edges. These maps are vital to make motion deblurring accomplishable in different MAP-variant frameworks.

Implicit Regularization Shan *et al*. [20] adopted a sparse image prior. This method, in the first a few iterations, uses a large regularization weight to suppress insignificant structures and preserve strong ones, creating crisp-edge image results, as exemplified in Fig. 1(b). This scheme is useful to remove harmful subtle image structures, making kernel estimation generally follow correct directions in iterations.

Krishnan *et al*. [16] used an L_1/L_2 regularization scheme. The main feature is to adapt L_1-norm regularization by treating the L_2-norm of image gradients as a weight in iterations. One intermediate result from this method is shown in Fig. 1(c). The main difference between this form and that of [20] is on the way to adapt regularization strength in iterations. Note both of them suppress details in the early stage during optimization.

Explicit Filter and Selection In [19, 3], shock filter is introduced to create a sharpened reference map for kernel estimation. Cho and Lee [3] performed bilateral filter and edge thresholding in each iteration to remove small- and medium-amplitude structures (illustrated in Fig. 1(d)), also avoiding the trivial solution.

Xu and Jia [25] proposed a texture-removal strategy, explained and extended in [28], to guide edge selection and detect large-scale structures. The resulting edge map in each step is also a small-edge-subdued version from the natural input. These two schemes have been extensively validated in motion deblurring.

Unnatural Representation The above techniques enable several successful MAP frameworks in motion deblurring. All of them have their intermediate image results or edge maps different from a natural one, as shown in Fig. 1, due to only containing high-contrast and step-like structures while suppressing others. We generally call them *unnatural representation*, which is the key to robust kernel estimation in motion deblurring.

1.2. Our Contribution

Based on the step-edge properties in unnatural representation, we in this paper propose a new sparse L_0 approximation scheme to generalize these frameworks. Compared to local shock filter, our edges are not explicitly created by filtering in extra steps. Instead, we incorporate a new regularization term consisting of a family of loss functions to approximate the L_0 cost into the objective, which, during optimization, leads to consistent energy minimization and accordingly fast convergence.

Our L_0 scheme is mathematically established with high-sparsity-pursuit regularization. It also assures only salient change in the image is preserved and made use of, making our method orders of magnitudes faster than other alternatives with implicit sparse regularization.

Besides this new sparse image representation, we also contribute a unified framework for both uniform and non-uniform deblurring, which no longer relies on ad-hoc edge selection, spatial filtering, or edge re-weighting. This framework does not sacrifice the competency in solving the challenging deblurring problem. Given a family of loss functions, based on which a graduate non-convexity could be applied, significant edges quickly improve kernel estimates in only a few iterations. This is most beneficial for non-uniform deblurring where intensive computation is needed in each iteration.

1.3. Other Related Work

In non-uniform deblurring, considering 3D camera rotation, Shan *et al*. [21] proposed solving in-plane rotation using single image. Whyte *et al*. [24] presented a non-uniform method using variational Bayesian. Tai *et al*. [22] solved non-blind deconvolution with a general projective motion model. Joshi *et al*. [12] advocated a hardware solution that physically captures camera rotation. Gupta *et al*. [7] proposed a different 3D approximation considering translation, as well as in-plane rotation. It is shown in [14] that both the 3D models [24, 7] produce good results in approximating the original 6D model. For acceleration, locally uniform approximation was adopted by Harmeling *et al*. [8] and Hirsch *et al*. [9], which combines the patch-based model and a global 3D camera motion one. Whyte *et al*. [23] concurrently proposed a fast forward model using FFTs and addressed the problem of pixel saturation. Shock filter is applied to generate sharp edge prediction. In dealing with depth variation, a tree structure was proposed in [26] to hierarchically estimate blur kernels with scale change.

2. Framework

We denote by x the latent image and y the blurred observation. x and y are in their vector forms. The discrete blur model for camera shake can be expressed as

$$y = \sum_m k_m H_m x + \varepsilon, \tag{1}$$

where x, y, and noise ε are $N \times 1$ vectors. N is the number of pixels in the image. m indexes camera pose samples. H_m is a $N \times N$ transformation matrix, which corresponds to either camera rotation or translation for pose m. k_m denotes the time that camera pose m lasts and is a weight in this function. Eq. (1) models the blurred image as summing unblurred images from all camera poses, which approximates continuous integral on light receival for each pixel.

Because H can be camera rotation R or translation M, we discuss these two cases. Camera rotation with 3 degree of freedoms (DoF) is generally sufficient to model non-uniform deblurring [14]. Each R_m thus corresponds to a camera rotation pose, sampled in 3D. We replace each H_m in Eq. (1) by rotation R_m to reduce the solution space. Due to the bi-linearity of the blur model $\sum_m k_m R_m x$, there exist two different forms as

$$\sum_m k_m R_m x = B_R x = A_R k, \tag{2}$$

where $B_R = \sum_m k_m R_m$ and vector $k = (k_0 \;\; k_1 \;\; \cdots)^T$, containing all k_m. $\mathrm{col}_m(A_R) = R_m x$ where $\mathrm{col}_m(\cdot)$ fetches the m-th column of matrix A.

Blur with in-plane translation M is referred to as uniform blur. Each M_m is thus a sample in 2D and its total number is called kernel size. In camera translation, we similarly substitute M_m for H_m in Eq. (1), which yields

$$\sum_m k_m M_m x = B_M x = A_M k, \tag{3}$$

where B_M and A_M are block Toeplitz with Toeplitz blocks (BTTB) matrices, since camera translation is linear translation invariant (LTI).

New Sparsity Function Our framework contains a sparse $\phi_0(\cdot)$ loss function, which can effectively approximate L_0 sparsity during iterative optimization. Given an input image z, it regularizes the high frequency part by manipulating gradient vectors $\partial_* z$, where $* \in \{h, v\}$ denoting two directions, for each pixel i. The function is defined as

$$\phi_0(\partial_* z) = \sum_i \phi(\partial_* z_i), \tag{4}$$

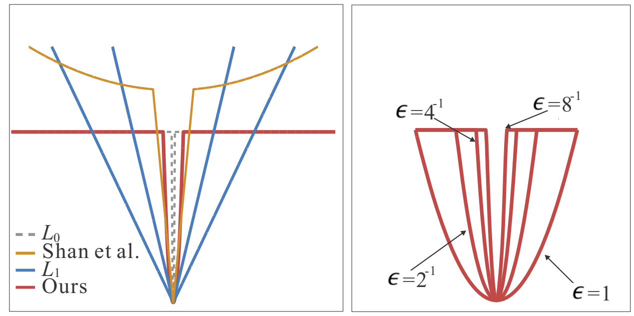
where

$$\phi(\partial_* z_i) = \begin{cases} \frac{1}{\epsilon^2}|\partial_* z_i|^2, & \text{if } |\partial_* z_i| \le \epsilon \\ 1, & \text{otherwise} \end{cases} \tag{5}$$

$\phi(\cdot)$ is actually a piecewise function that concatenates a quadratic penalty and a constant. When $|\partial_* z_i| < \epsilon$, $\phi(\cdot)$ is continuous, a necessary condition to form a loss function. The red curve in Fig. 2(a) illustrates the form of $\phi(\cdot)$. It is very close to the most sparse L_0 function. A few other sparsity-pursuit functions used in deblurring [20, 16] are also plotted in this figure. With higher sparsity, $\phi(\cdot)$ raises a few desirable properties, discussed later.

Final Objective $\phi_0(\cdot)$ is incorporated in our method to regularize optimization, which seeks an intermediate sparse representation $\tilde{x}$ containing only necessary edges. Our objective to estimate the blur kernel from the input image is

$$\min_{(\tilde{x},k)} \left\{ \|\sum_m k_m H_m \tilde{x} - y\|^2 + \lambda \sum_{*\in\{h,v\}} \phi_0(\partial_* \tilde{x}) + \gamma\|k\|^2 \right\}, \tag{6}$$



(a) (b)

Figure 2. Plots of different penalty functions.

H_m could be either R_m or M_m depending on solving the uniform or non-uniform deblurring problem, as shown in Eqs. (2) and (3). λ and γ are two regularization weights.

The function has three terms. The data fidelity term $\|\sum_m k_m H_m \tilde{x} - y\|^2$ enforces the blur model constraint. $\|k\|^2$ helps reduce kernel noise. It also enables fast kernel estimation using FFTs with the quadratic form. $\phi_0(\partial_* \tilde{x})$ is the new regularization term, which is instrumental in our method, to guide kernel estimation.

Property Analysis Gradients with different amplitudes are penalized in $\phi(\cdot)$ when ϵ is small. Combined with the fidelity term $\|\sum_m k_m H_m \tilde{x} - y\|^2$, $\phi(\cdot)$ has an effect to remove fine structures and keep useful salient ones in $\tilde{x}$ in order to minimize the total cost. These benefits in part stem from the inhomogeneity property (a.k.a. scale invariance) of the near-L_0 measure – that is, $\phi_0(a^t \partial_* z) \approx \phi_0(\partial_* z)$ given any positive-element vector a.

The *unnatural* L_0 *representation* computed from our method is image $\tilde{x}$ produced in iterations to solve Eq. (6). One example has been shown in Fig. 1(e). Compared to employing shock filter [19, 3] as an extra step that cannot fit into the overall function for consistent energy minimization, $\phi(\cdot)$ and $\tilde{x}$ are elegantly incorporated in one objective, optimizing which monotonically decreases energy. Meanwhile, $\tilde{x}$ is not produced by local filtering, which thus guarantees to contain only *necessary* strong edges, regardless of blur kernels.

3. Optimization

Eq. (6) is solved by alternatively computing

$$\tilde{x}^{t+1} = \operatorname*{argmin}_{\tilde{x}} \left\{ \|B^t \tilde{x} - y\|^2 + \lambda \sum_{*\in\{h,v\}} \phi_0(\partial_* \tilde{x}) \right\}, \tag{7}$$

$$k^{t+1} = \operatorname*{argmin}_{k} \left\{ \|A^{t+1} k - y\|^2 + \gamma\|k\|^2 \right\}, \tag{8}$$

in each iteration $t+1$, where the information of k^t and $\tilde{x}^t$ is embedded in the blur matrices B^t and A^t respectively. By convention, blur kernels are estimated in a coarse-to-fine manner in an image pyramid. Estimate from one image level is taken as an initialization of the next one. We elaborate in what follows the optimization process in iteration $t+1$ in the finest level. Computation in other coarser levels and in different iterations is similar.

3.1. Solve for $\tilde{x}^{t+1}$ with k^t

Eq. (7) is non-convex w.r.t. $\tilde{x}$ due to the incorporation of $\phi_0(\cdot)$. It can be optimized using the half-quadratic L_0 minimization solver introduced in [27]. We employ a similar scheme that minimizes a family of loss functions. This scheme starts from an easy convex expression and heads towards the ideal solution in iterations.

Taking ϵ as a parameter, $\phi(\partial_* z_i)$ defined in Eq. (5) is equivalent to

$$\phi(\partial_* z_i;\epsilon) = \min_{l_{*i}} \left\{ |l_{*i}|^0 + \frac{1}{\epsilon^2}(\partial_* z_i - l_{*i})^2 \right\}, \tag{9}$$

where $* \in \{h, v\}$. Each $l_{*i} \in \mathbb{R}$ and each $|l_{*i}|^0$ is a number to the zero power: $|l_{*i}|^0 = 1$ if $l_{*i} \neq 0$ and $|l_{*i}|^0 = 0$ otherwise. Proof that Eq. (5) is equivalent to Eq. (9) is presented in our supplementary file in the project website. Eq. (9) reveals the fact that $\phi(\partial_* z_i)$ is actually a minimum of $\left\{|l_{*i}|^0 + \frac{1}{\epsilon^2}(\partial_* z_i - l_{*i})^2\right\}$, independent of l_{*i}.

With Eq. (9), a family of loss functions are obtained by setting ϵ differently. Four examples are shown in Fig. 2(b) where ϵ decreasing from 1 to $1/8$ makes the resulting function approach the L_0 one. The objective for computing $\tilde{x}$ given a specific ϵ is therefore rewritten as

$$\min_{\tilde{x},l} \left\{ \frac{1}{\lambda}\|B\tilde{x} - y\|^2 + \sum_{*\in\{h,v\}} \sum_i \left\{ |l_{*i}|^0 + \frac{1}{\epsilon^2}(\partial_* \tilde{x}_i - l_{*i})^2 \right\} \right\}. \tag{10}$$

We alternate between computing $\tilde{x}$ and updating l_{hi} and l_{vi} in iterations for each loss function controlled by ϵ.

Update l Solving for l_{hi} in the function $\{|l_{hi}|^0 + 1/\epsilon^2(\partial_h \tilde{x}_i - l_{hi})^2\}$ can be achieved by hard thresholding, given by

$$l_{hi} = \begin{cases} 0, & |\partial_h \tilde{x}_i| \leq \epsilon \\ \partial_h \tilde{x}_i, & \text{otherwise} \end{cases} \tag{11}$$

The proof that it holds is provided in our project website. The result of l_{vi} can be obtained similarly. With the pixel-wise closed-form solution, updating l is thus computationally easy and quick.

Update $\tilde{x}$ After fixing l, the energy w.r.t. $\tilde{x}$ is quadratic. The optimal solution is yielded by solving a linear equation. It is efficient when dealing with in-plane translational camera motion M, as the matrix-vector production with regard to the BTTB matrix can be achieved using FFTs. The solution is expressed as

$$F(\tilde{x}) = \frac{\overline{F(B_M)} \cdot F(y) + \frac{\lambda}{\epsilon^2}(\overline{F(\partial_h)} \cdot F(l_h) + \overline{F(\partial_v)} \cdot F(l_v))}{\overline{F(B_M)} \cdot F(B_M) + \frac{\lambda}{\epsilon^2} F_D^2}, \tag{12}$$

where $F(\cdot)$ is the FFT operator, which takes an image vector or a BTTB kernel matrix as input. $\overline{F(\cdot)}$ is the complex conjugate. $F^{-1}(\cdot)$ is the inverse FFT. Multiplication and division are element-wise operation on two complex vectors. l_h and l_v are vectors concatenating all l_{hi} and l_{vi} respectively. F_D^2 denotes $|F(\partial_h)|^2 + |F(\partial_v)|^2$, where $|\cdot|$ is element-wise absolute.

When considering non-uniform blur caused by camera rotation, which is spatially variant, the blur matrix B_R is no longer block Toeplitz with Toeplitz blocks (BTTB). We turn to the fast forward approximation with locally-uniform assumption [9, 23], which regularly divides the image into P patches. Every two neighboring patches have 50% overlap area. In this approximation, each patch has one blur kernel basis $A_{R\delta}$, generated by applying rotation to a special patch δ containing all black pixels except for a white point at the center. A blur kernel for each patch is then formed using $A_{R\delta}k = \sum_i k_i R_i \delta$, similar to that in Eq. (2). The basis $A_{R\delta}$ is computed beforehand and is image independent [9].

We denote by $C_p(\tilde{x})$ and $C_k(A_{R\delta}k)$ the p-th patch from the latent image and its corresponding kernel $A_{R\delta}k$ respectively. A blurred patch $C_p(y)$ is generated by convolving $C_k(A_{R\delta}k)$ and $C_p(\tilde{x})$. In frequency domain, this process is expressed as

$$y = \sum_{p=1}^{P} C_p^{-1} F^{-1}(\underbrace{F(C_{pk}(A_{R\delta}k))}_{\text{kernel}} \cdot \underbrace{F(w \cdot C_p(\tilde{x}))}_{\text{patch}}) + \varepsilon, \tag{13}$$

where $C_p^{-1}(\cdot)$ is the operator to paste the patch back to the image. w is a vector representing the Bartlett-Hann window function tapering to zeros near the patch boundary, which helps blend overlaid patches.

This model enables a closed-form approximation of $\tilde{x}$ by deconvolving each patch separately, written as

$$\tilde{x} = \frac{1}{W} \sum_p C_p^{-1} F^{-1} \frac{\overline{F(C_k(A_{R\delta}k))} \cdot F(w \cdot C_p(y)) + \frac{\lambda}{\epsilon^2} F_{Dl}}{F_k^2 + \frac{\lambda}{\epsilon^2} F_D^2}, \tag{14}$$

where $F_k^2 = |F(C_k(A_{R\delta}k))|^2$ and $F_{Dl} = (\overline{F(\partial_h)} \cdot F(C_p(l_h)) + \overline{F(\partial_v)} \cdot F(C_p(l_v)))$. $1/W$ is a weight to suppress visual artifacts caused by the window functions [9].

For non-uniform deblurring, we alternate between Eqs. (11) and (14) to update l and $\tilde{x}$ respectively. On the contrary, when the blur is uniform, Eqs. (11) and (12) are used instead. In implementation, we use a family of 4 loss functions with $\epsilon \in \{1, 2^{-1}, 4^{-1}, 8^{-1}\}$. It starts from $\epsilon = 1$, as illustrated in Fig. 2. This process corresponds to lines 6-9 in Algorithm 1 and is conceptually explainable by Graduate Non-Convexity (GNC) [2].

One important consideration is to save computation especially in early estimation stages. We achieve it by setting iteration numbers for different loss functions inversely proportional to ϵ, as indicated in line 6 in Algorithm 1. Large-ϵ

Algorithm 1: L_0 Deblurring in One Image Level

input : blurred image y
output: blur kernel k, deblurred image $\tilde{x}$

1 initialize k^1 from the coarser-scale kernel estimate
2 **for** $t = 1 : 5$ **do**
3   *// update image*
4     $\epsilon \leftarrow 1$
5   **for** $i = 1 : 4$ **do**
6   **for** $j = 1 : \epsilon^{-1}$ **do**
7   solve for l using Eq. (11)
8   solve for $\tilde{x}^{t+1}$ using Eqs. (12) or (14)
9   **end**
10         $\epsilon \leftarrow \epsilon/2$ *// graduate non-convexity*
11   **end**
12   *// update kernel*
13   solve for k^{t+1} using Eqs. (15) or (17)
14 **end**

loss functions need only a small number of iterations because they are more convex-like, easy to optimize. Also, their results are taken as an initialization for further refinement in smaller-ϵ loss functions; coarse estimates suffice.

3.2. Solve for k^{t+1} with $\tilde{x}^{t+1}$

The energy function w.r.t. k in Eq. (8) is quadratic. With the duality of the blur kernel and latent image in convolution, the A_M matrix for translational camera motion is BTTB, making blur kernel estimation also find a closed-form solution in frequency domain [25]. It is expressed as

$$k^{t+1} = F^{-1}\left(\frac{\overline{F(A_M^{t+1})}F(y)}{|F(A_M^{t+1})|^2 + \gamma}\right), \tag{15}$$

where γ is the regularization weight given in Eq. (6).

For the rotational model, A_R cannot be diagonalized using FFTs. We thus iteratively update k following the multiplication rule. Specifically, taking derivatives of Eq. (8) and setting them to zeros yield

$$\frac{A_R^T y}{(A_R^T A_R + \gamma)k} = 1. \tag{16}$$

$A_R^T A_R k$ and $A_R^T y$ can be efficiently computed using the forward approximation. To further reduce iterations, we introduce a parameter α controlling the "step size" [1], making updating expressed as

$$k^{(n+1)} = k^{(n)} \cdot \left(\frac{A_R^T y}{(A_R^T A_R + \gamma)k^{(n)}}\right)^{\alpha}, \tag{17}$$

where α is set to 1.5 to let the algorithm converge quickly, resulting in the final estimate k.

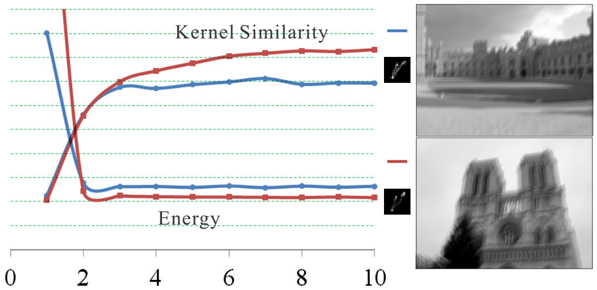


Figure 3. Convergence illustration. The red and blue curves are for the two examples respectively. "Energy" and "Kernel Similarity" plots denote resulting energy and kernel similarity to the ground truth in iterations. The results get stable in less than 5 iterations.

Algorithm 1 shows main steps. Kernel estimation in line 13 takes the major computation. Our method only needs to perform it 5 times (according to number t in Algorithm 1) in one image level, compared to tens or even hundreds iterations involved in other approaches.

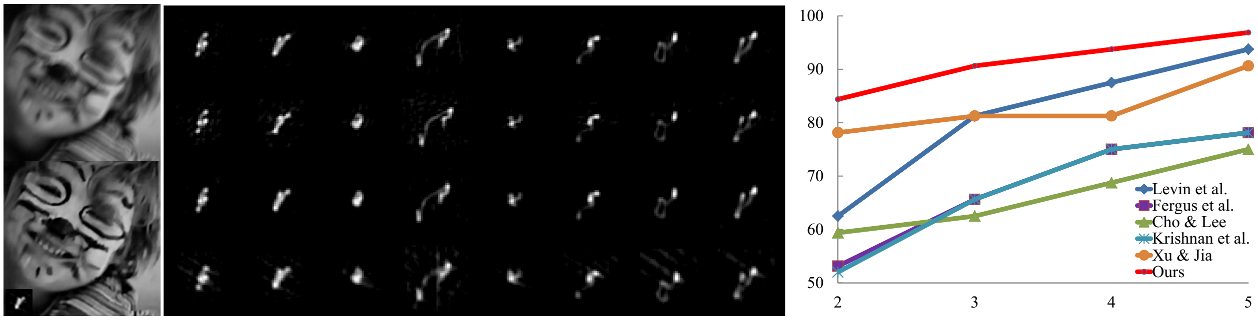
3.3. Final Image Restoration

The computed map $\tilde{x}$ is not the final latent natural image estimate due to lack of details. In the final step, we restore the natural image by non-blind deconvolution given the final kernel estimate. A Hyper-Laplacian prior with $L_{0.5}$ norm regularization [15] is used. Image restoration for both the uniform and non-uniform blur is accelerated by FFTs.

4. Discussion

Difference to Shock Filter Compared to edge prediction using shock filter and edge thresholding [3, 25], our approach employs Eqs. (11) and (12) to provide more appropriate edge reference maps within a well-established optimization framework. Eq. (11) achieves theoretically sound gradient thresholding without extra ad-hoc operations. In the sequel, our method does not have the edge location problem inherent in shock filter when blur kernels are highly non-Gaussian or the saddle points used in shock filter do not correspond to latent image edges. Our optimization framework (Eq. (12)) can naturally produce a sparse representation faithful to the input, vastly benefitting motion deblurring.

Fast Convergence We have observed fast convergence in our method. We plot in Fig. 3 the energies w.r.t the number of iterations for two examples. In practice, 5-pass kernel estimation in one image level is enough, compared to hundreds of iterations by variational Bayesian inference [5], and tens of iterations in the methods of [20, 16]. Our estimation quality is also high. We measure the similarity between the estimated kernels and the ground truth using the maximum correlation, counting in kernel shift. The upper



(a) (b) (c)

Figure 4. Quantitative evaluation on the dataset [17].

	Gradient	Gradient/Intensity	Intensity
PSNRs	29.50	32.35	28.52

Table 1. Image space vs. gradient space.

Methods	255x255	800x800	1024x1024
Cho and Lee [3] (C++)	0.77	5.80	11.53
Xu and Jia [25] (C++)	0.80	5.73	13.60
Krishnan *et al.* [16] (Matlab)	25.55	215.08	273.98
Levin *et al.* [18] (Matlab)	76.69	1084.12	1737.37
Ours (Matlab)	1.05	5.78	12.27

Table 2. Running time (in seconds) of different methods in three image resolutions.

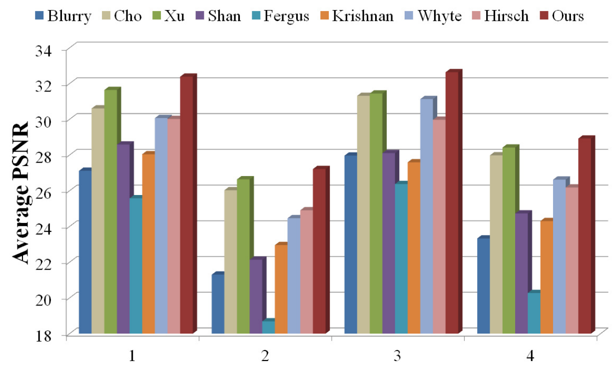


Figure 5. Quantitative comparison on the dataset [14]. The numbers below the horizontal axis index image sets.

two plots in Fig. 3 manifest that the rapid convergence does not sacrifice the quality of kernel estimates.

Image Space versus Filter Space The data term constraint can be defined in either image space (enforcing intensity level similarity) or gradient space (for gradient domain confidence). We conduct experiments on the dataset from [18] and list the average PSNRs for all 32 images in Table 1. The finding is that using the image space constraint for updating $\tilde{x}$ and gradient domain energy to update kernel k (middle of Table 1) is better than other alternatives.

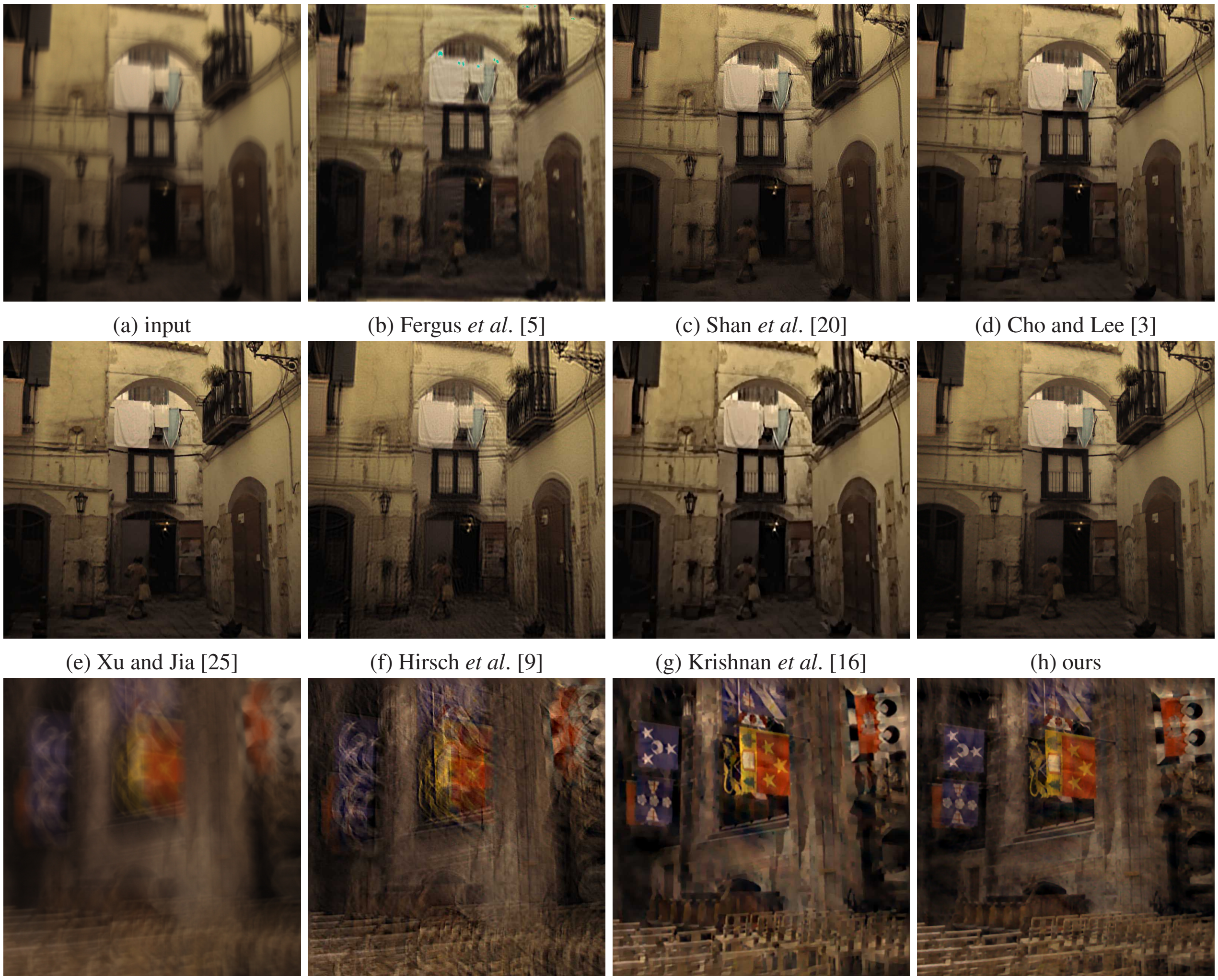
Running Time All steps can be accelerated by FFTs. We compare the running time of several representative deblurring methods, of which implementations are available. Running time reported in Table 2 is obtained on the same PC with an Intel i7 CPU and 8GB memory. Our Matlab implementation is quite efficient, which can be further sped up in optimized C.

5. Experimental Results

We experiment with data on two publicly available blur-image sets [17, 14]. The set of [17] contains 32 images of size 255×255 blurred with 8 different kernels. Error ratio between our deconvolved images and the ground truth is obtained. We use the provided script and non-blind deconvolution function to generate the results, for fairness. We set λ and γ to $2E-3$ and 40 respectively for all examples. We compare our error ratios with those of Fergus et al. [5], Cho and Lee [3], Xu and Jia [25], Levin *et al.* [18], and Krishnan *et al.* [16], and show them in Fig. 4(c). An input image and our result are shown in Fig. 4(a). All the 32 kernel estimates from the proposed method are shown in (b). As indicated in [17], error ratios over 2 will make the result visually implausible. Our method takes the lead with 93.75% of the results under error ratio 2.

Quantitative Evaluation on the dataset of [14] We also test our algorithm on another dataset, where images and ground truth kernels are provided [14]. This dataset consists of 4 images, each is blurred with 12 kernels, including several large ones. Two examples are shown in Fig. 6 with result comparison. Quantitative evaluation is conducted by comparing each deblurring result with 199 unblurred images captured along the camera motion trajectory and recording the largest PSNR.

For each image example, we quantitatively compare average PSNRs among different methods in Fig. 5. Note that all top ranking methods [25, 3, 23] use shock filter except ours. The proposed method ranks #1 now.



(a) input (b) Fergus *et al*. [5] (c) Shan *et al*. [20] (d) Cho and Lee [3]

(e) Xu and Jia [25] (f) Hirsch *et al*. [9] (g) Krishnan *et al*. [16] (h) ours

(i) input (j) Shan *et al*. [20] (k) Cho and Lee [3] (l) ours

Figure 6. Visual comparison of state-of-the-art methods. Please view these results in their original resolutions in our project website.

Non-uniform Deblurring Our framework is fully applicable to non-uniform deblurring with the model depicted in the paper. Fig. 7 shows two examples. Our results are visually comparable to others. The result shown in Fig. 7(i) is generated using 1128 seconds. For comparison, previously most efficient approach [9] takes 1567 seconds for the same image. More results are included in the project website. An executable is also publicly available.

6. Concluding Remarks

We have presented a new framework for both uniform and non-uniform motion deblurring, leveraging an unnatural L_0 sparse representation to greatly benefit kernel estimation and large-scale optimization. We proposed a unified model, which seeks gradient sparsity close to L_0 to remove pernicious small-amplitude structures. The method not only provides a principled understanding of effective motion deblurring strategies, but also notably augments performance based on the new optimization process.

Acknowledgements

The work described in this paper was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region (Project No. 413110).

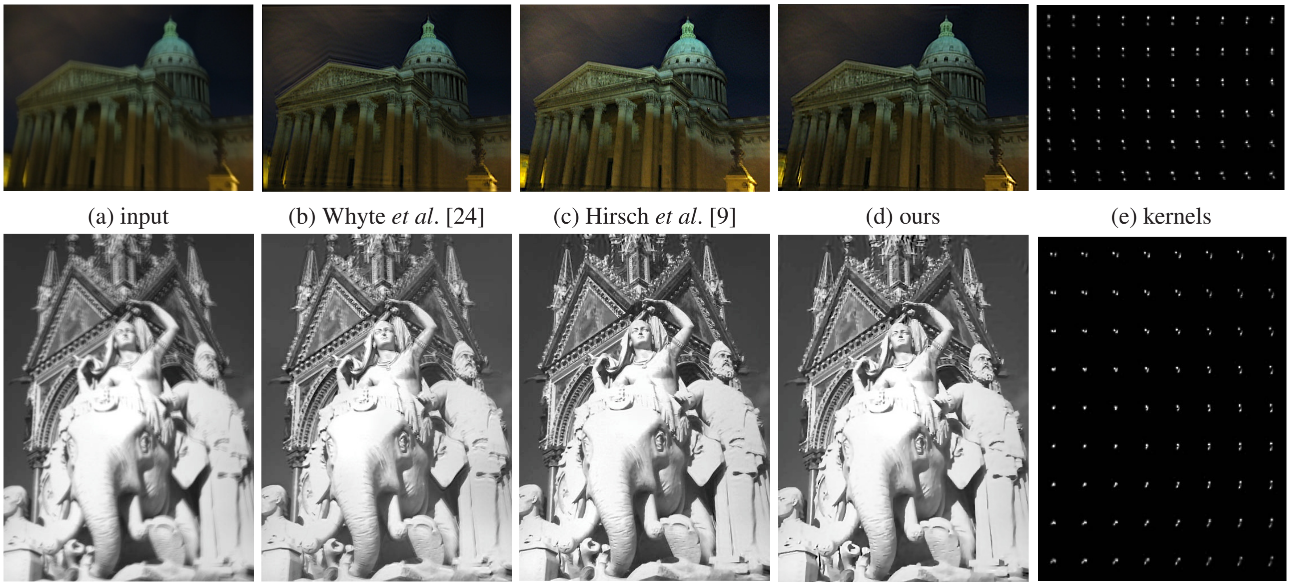
References

[1] D. S. Biggs and M. Andrews. Acceleration of iterative image restoration algorithms. *Applied Optics*, 36(8):1766–1775, 1997.

[2] A. Blake and A. Zisserman. *Visual reconstruction*. MIT Press, 1987.

[3] S. Cho and S. Lee. Fast motion deblurring. *ACM Trans. Graph.*, 28(5), 2009.

[4] T. S. Cho, S. Paris, B. K. P. Horn, and W. T. Freeman. Blur kernel estimation using the radon transform. In *CVPR*, pages 241–248, 2011.



(a) input (b) Whyte *et al*. [24] (c) Hirsch *et al*. [9] (d) ours (e) kernels

(f) input (g) Harmeling *et al*. [8] (h) Hirsch *et al*. [9] (i) ours (j) kernels

Figure 7. Non-uniform deblurring results. Kernels are resized for visualization.

[5] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single photograph. *ACM Trans. Graph.*, 25(3):787–794, 2006.

[6] A. Goldstein and R. Fattal. Blur-kernel estimation from spectral irregularities. In *ECCV*, pages 622–635, 2012.

[7] A. Gupta, N. Joshi, C. L. Zitnick, M. F. Cohen, and B. Curless. Single image deblurring using motion density functions. In *ECCV*, pages 171–184, 2010.

[8] S. Harmeling, M. Hirsch, and B. Schölkopf. Space-variant single-image blind deconvolution for removing camera shake. In *NIPS*, pages 829–837, 2010.

[9] M. Hirsch, C. J. Schuler, S. Harmeling, and B. Schölkopf. Fast removal of non-uniform camera shake. In *ICCV*, pages 463–470, 2011.

[10] Z. Hu and M.-H. Yang. Good regions to deblur. In *ECCV*, pages 59–72, 2012.

[11] J. Jia. Single image motion deblurring using transparency. In *CVPR*, 2007.

[12] N. Joshi, S. B. Kang, C. L. Zitnick, and R. Szeliski. Image deblurring using inertial measurement sensors. *ACM Trans. Graph.*, 29(4), 2010.

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

[14] R. Koehler, M. Hirsch, S. Harmeling, B. Mohler, and B. Schölkopf. Recording and playback of camera shake: benchmarking blind deconvolution with a real-world database. In *ECCV*, pages 27–40, 2012.

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

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

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

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

[19] J. H. Money and S. H. Kang. Total variation minimizing blind deconvolution with shock filter reference. *Image and Vision Computing*, 26(2):302–314, 2008.

[20] Q. Shan, J. Jia, and A. Agarwala. High-quality motion deblurring from a single image. *ACM Trans. Graph.*, 27(3), 2008.

[21] Q. Shan, W. Xiong, and J. Jia. Rotational motion deblurring of a rigid object from a single image. In *ICCV*, 2007.

[22] Y.-W. Tai, P. Tan, and M. S. Brown. Richardson-lucy deblurring for scenes under a projective motion path. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(8):1603–1618, 2011.

[23] O. Whyte, J. Sivic, and A. Zisserman. Deblurring shaken and partially saturated images. In *ICCV Workshops*, pages 745–752, 2011.

[24] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce. Non-uniform deblurring for shaken images. In *CVPR*, pages 491–498, 2010.

[25] L. Xu and J. Jia. Two-phase kernel estimation for robust motion deblurring. In *ECCV*, pages 157–170, 2010.

[26] L. Xu and J. Jia. Depth-aware motion deblurring. In *ICCP*, 2012.

[27] L. Xu, C. Lu, Y. Xu, and J. Jia. Image smoothing via l_0 gradient minimization. *ACM Trans. Graph.*, 30(6), 2011.

[28] L. Xu, Q. Yan, Y. Xia, and J. Jia. Structure extraction from texture via relative total variation. *ACM Transactions on Graphics (TOG)*, 31(6), 2012.