Letter

Blur-Kernel Bound Estimation From Pyramid Statistics

Shaoguo Liu, Haibo Wang, Jue Wang, and Chunhong Pan

***Abstract*— This letter presents an approach for automatically estimating the spatial bound of the blur kernel in a motion-blurred image based on the statistics of multilevel image gradients. We observe that blur has a significant impact on the histogram of oriented gradients (HOGs) at higher levels of an image pyramid, but has much less of an impact at coarser levels. Based on this fact, we estimate the spatial bound of the unknown blur kernel using a learning-based approach. We first learn a generic pyramid HOG model from natural sharp images, then given an HOG pyramid of a blurry image, we predict the corresponding model of its latent sharp image. Finally, we learn another model to predict the spatial kernel bound from the difference between the observed and the predicted HOG pyramids. Experimental results show that the proposed method can estimate accurate blur kernel sizes, enabling existing blind deconvolution methods to achieve best possible results.**

***Index Terms*— Blur-kernel bound estimation, image deblur, motion deblur, motion prior, pyramid statistics.**

I. Introduction

Images taken by handheld cameras are often blurry due to camera shake, which can be mathematically modeled by

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

where $\mathbf{k}$ is the blur kernel, $\mathbf{L}$ is the latent sharp image, $\mathbf{n}$ is the white noise, and $\otimes$ is the convolution operator. The purpose of image deblurring, or blind deconvolution, is to recover $\mathbf{L}$ and $\mathbf{k}$ from $\mathbf{B}$, which is a severely ill-posed problem. Furthermore, $\mathbf{k}$ could be either static or spatially varying across the image. Following the majority of previous work, we focus on the case of static $\mathbf{k}$.

Image deblurring has been an active research topic in recent years. Roughly speaking, existing approaches can be classified into three categories: 1) maximum *a posteriori* (MAP) estimation-based; 2) variational Bayesian (VB)-based; and 3) edge prediction-based approaches. MAP-based methods [1]–[3] try to maximize the joint posterior probability with respect to $\mathbf{k}$ and $\mathbf{L}$, while VB-based methods [4]–[6] consider a distribution of probable $\mathbf{L}$s in order to find the best $\mathbf{k}$. Edge-based approaches [7]–[9] explicitly extract structural edges and use their sharpened version obtained by either heuristic filtering [7] or global optimization [9] for kernel estimation. Given the ill-posed nature of the problem, all these approaches rely on using additional image priors for regularizing $\mathbf{L}$, such as the Gaussian prior on its intensities [10] and derivatives [11], the heavy-tailed Gaussian prior on its gradient histogram [4], and the $l_{1,2}$ regularization on its gradients [2]. The blur kernel $\mathbf{k}$ is often assumed to be sparse and continuous [1], [3], [12].

(a) (b) (c) (d)

Fig. 1. Motion deblurring with different kernel sizes. (a) Blurry image in [9]. (b) Deblurring using [7] with a 49×49 kernel. (c) Deblurring using [7] with a 69×69 kernel. (d) Deblurring using [7] with a 139×139 kernel. The different deblurring results show that choosing an appropriate kernel size is important for deblurring.

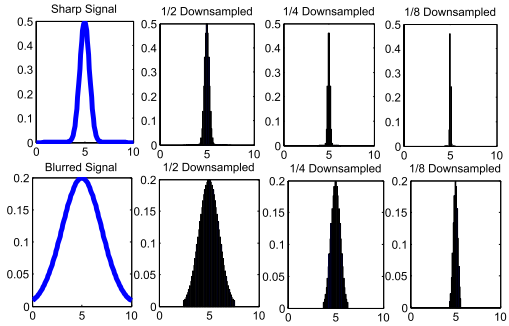

Fig. 2. Downsampling decreases blurriness. From left to right, as the signal is downsampled more, the blurred signal becomes sharper and more similar to the original signal.

The spatial bound of $\mathbf{k}$, also called the blur kernel size, is an important parameter in all image deblurring approaches, and is often manually selected by a tedious trial-and-error process. Furthermore, we found that existing methods are sensitive to the setting of this parameter, an improper blur kernel size often leads to significant kernel estimation errors [2], [7], which is shown in Fig. 1. In this letter, we present an approach to automatically infer a proper kernel size from an input image, which not only saves the user's effort for tedious parameter tweaking, but also allows the existing deblurring approaches to be automatically and reliably applied on photo albums for the first time.

Our approach is based on analyzing the statistics of image gradients, as motion blur will significantly remove high-frequency image content. Ideally, if $\mathbf{B}$ and $\mathbf{L}$ are both known, we can compute the difference between gradient statistics to infer the blur kernel size. However, given we do not know $\mathbf{L}$, we use a learning-based approach to try to recover the gradient statistics of $\mathbf{L}$ from that of $\mathbf{B}$, using image pyramids. This is based on the following two key observations.

1) The impact of motion blur to the image gradient will reduce in downsampled versions of the image, thus for low levels of

Manuscript received September 6, 2014; revised December 29, 2014; accepted March 26, 2015. Date of publication April 2, 2015; date of current version May 3, 2016. This work was supported by the National Natural Science Foundation of China under Grant 91338202, Grant 61370039, and Grant 61203279. This paper was recommended by Associate Editor X. Li.

S. Liu and C. Pan are with National Laboratory of Pattern Recognition, Chinese Academy of Sciences, Beijing 100190, China (e-mail: sgliu2013@gmail.com; chunhongp@gmail.com).

H. Wang is with Shandong University, Jinan 250100, China (e-mail: hbwang1427@gmail.com).

J. Wang is with Adobe Research, San Jose, CA 95110 USA (e-mail: arphid@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2015.2418585

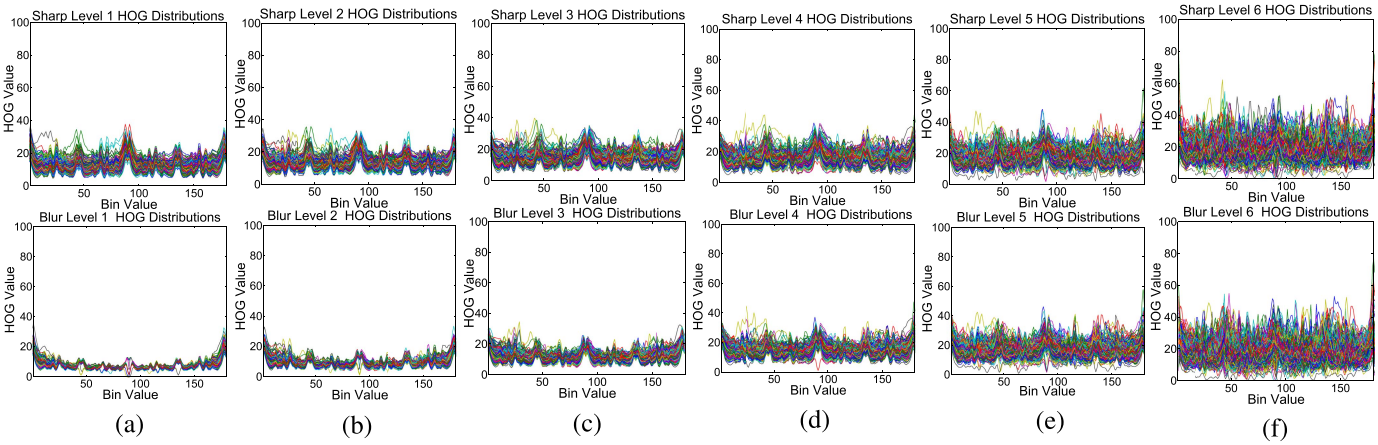

Fig. 3. Illustrating the pyramid HOG distributions. Top row: HOGs of 1600 natural images. Bottom row: HOGs of artificially blurred versions of the 1600 images. Note that a random kernel of size 15×1 is used for the illustration purpose. One can easily find out that from left to right as the pyramid level goes down, the similarity of the two distributions in each column increases. (a) Level 1. (b) Level 2. (c) Level 3. (d) Level 4. (e) Level 5. (f) Level 6.

Fig. 4. Examples images in our data set.

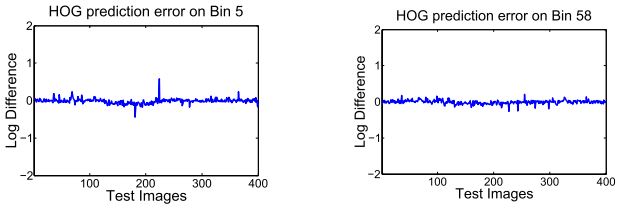

Fig. 5. Illustrating the prediction error between the inferred high-level HOG features using the learned SVM models and the ground-truth HOG in two randomly selected histogram bins. Thousand two hundred images are used to train the SVMs while the remaining 400 ones are used for the testing. (a) Result 1. (b) Result 2.

the pyramid, **L** and **B** have roughly the same gradient statistics (demonstrated in Figs. 2 and 3).

2) It is possible to recover higher level gradient statistics from lower level statistics using machine learning. This is because it has been shown that natural images can be reconstructed using a small set of common image patches [13], thus different images may share some common properties on how the gradient statistics change from low to high pyramid levels, which can be learned from a training data set that contains a large number of sharp natural images.

II. Algorithm

A. Workflow

Our method contains two main steps. In training step, we compute histogram of oriented gradient (HOG) [14] features in the pyramids of 1600 sharp images of a variety of natural scenes (as shown in Fig. 4). For each bin of the histogram, we learn the following two sets.

1) A set of support vector machines (SVMs) [15] that can predict high-level HOG features from low-level ones in the pyramid.
2) Another set of SVMs that map the difference between the observed and the predicted high-level HOG features to a 1-D kernel size. In testing step, for given **B**, we first infer the high-level HOG features of **L** using the SVMs learned in 1), and then estimate a series of 1-D kernel sizes from the SVMs learned in 2). Finally, the 2-D kernel size is determined by fitting the series of 1-D kernel sizes with an ellipse.

B. Learning Pyramid HOG Models

Pyramid HOG refers to the collection of HOGs built at each level of an image pyramid. Each histogram bin represents a unique gradient direction weighted by the gradient magnitude in that direction. A 180-bin HOG is used here, covering the interval of [1°, 180°]. Each HOG is first l_2-normalized, then smoothed with a 1-D-Gaussian filter of size 3×1.

Let h_{ij} denote the jth HOG bin at the ith pyramid level, where $i \in \{1, 2, \ldots n\}$ and $j \in \{1°, 2°, \ldots 180°\}$. For better understanding, the pyramid HOGs of the 1600 training images with $n = 6$ are shown in Fig. 3. Note that the blurry images are synthesized with random motion kernels.

Since we only care about how the HOG features change across different levels rather than their absolute values, we take the following normalization step in the log space:

$$r_{ij} = \log(h_{nj}) - \log(h_{ij}), \quad i = 1, 2, \ldots, n-1. \tag{2}$$

Here, we experimentally set n=6 according to the results in Fig. 3, for each HOG bin we have five such log differences: $r_{1j}, r_{2j}, r_{3j}, r_{4j}, r_{5j}$. As one can see from Fig. 5, the low-level features r_{4j}, r_{5j} of the sharp and the blurry images are similar to each other. For a blurry image **B**, we thus seek to estimate the high-level features r_{1j}, r_{2j}, r_{3j} in **L** from the low-level features r_{4j}, r_{5j} computed from **B**. To this end, we learn the following four SVM models in each histogram bin:

$$\begin{aligned} &\mathbf{M}_{0j}(r_{1j}|r_{2j}, r_{3j}, r_{4j}, r_{5j}), \quad \mathbf{M}_{1j}(r_{1j}|r_{4j}, r_{5j}) \\ &\mathbf{M}_{2j}(r_{2j}|r_{1j}, r_{4j}, r_{5j}), \quad \mathbf{M}_{3j}(r_{3j}|r_{1j}, r_{4j}, r_{5j}). \end{aligned} \tag{3}$$

Since we use 180-bin HOG, there are in total 180×4 such learned models. The rationale behind the definition of the models is as follows. $\mathbf{M}_{1j}$ infers r_{1j} from r_{4j} and r_{5j}, which can give us a rough r_{1j} value. Then, $\mathbf{M}_{2j}$ and $\mathbf{M}_{3j}$ uses r_{4j}, r_{5j} and the estimated r_{1j} to infer r_{2j} and r_{3j}, respectively. Since r_{2j} and r_{3j} belong to the intermediate levels, the inference of using r_{4j}, r_{5j}, and r_{1j} jointly avoids any bias toward higher or lower levels. Finally, r_{2j}, r_{3j}, r_{4j}, and r_{5j} are jointly used to infer a refined

Algorithm 1 High-Level Sharp Feature Inference

Input: low-level features r_{1j}^0, r_{4j}, r_{5j}, maximum iteration number $t_{\max}$, stopping threshold τ
Output: $r_{1j}^{\text{final}}, r_{2j}^{\text{final}}, r_{3j}^{\text{final}}$
Step 1: infer r_{2j}^t using model $\mathbf{M}_{2j}$
Step 2: infer r_{3j}^t using model $\mathbf{M}_{3j}$
Step 3:
infer r_{1j}^{t+1} using model $\mathbf{M}_{0j}$.
if $\|r_{1j}^{t+1} - r_{1j}^t\|_2^2 \leqslant \tau$ or $t \geqslant t_{\max}$
$r_{1j}^{\text{final}} = r_{1j}^{t+1}, r_{2j}^{\text{final}} = r_{2j}^t, r_{3j}^{\text{final}} = r_{3j}^t$.
exit the iterative process.
else
$r_{1j}^k = r_{1j}^{k+1}$, then go to **Step 1**.

r_{1j} in $\mathbf{M}_{0j}$. In practice, the high-level features r_{1j}, r_{2j}, r_{3j} are inferred in the following order. First, we derive r_{1j}^0 using model $\mathbf{M}_{1j}$ with the low-level features r_{4j}^0 and r_{5j}^0. Then, we iteratively perform the following three steps.

Step 1: Infer r_{2j}^t using model $\mathbf{M}_{2j}$.

Step 2: Infer r_{3j}^t using model $\mathbf{M}_{3j}$.

Step 3: Refine the inference of r_{1j}^{t+1} using $\mathbf{M}_{0j}$.

The iterative procedure stops when $\|r_{1j}^{t+1} - r_{1j}^t\|_2^2$ is below a preset threshold τ or the maximum number of iterations has been reached. The whole procedure is also summarized in Algorithm 1. τ is set to be 0.001 and the maximum iteration number is 10. Fig. 5 shows the accuracy of the inferred high-level HOG features.

C. 1-D Kernel Size Inference

In this step, we infer a 1-D kernel size in each HOG bin. We define a new feature, which is the difference between the estimate high-level features r_i^s and the corresponding features r_i^b measured from $\mathbf{B}$

$$\Delta r_i = r_i^s - r_i^b, \quad i = 1, 2, 3. \tag{4}$$

For training purposes, we synthetically blur the 1600 sharp images with 13 Gaussian kernels by varying kernel size as $\sigma = 1, 3, \ldots, 25$ as the input blurry images $\mathbf{B}$. There are in total $1600 \times 13 = 20\,800$ such blurry images. We choose to use Gaussian kernels due to its symmetry in all directions. For each HOG bin, we train a linear SVM model to predict σ from the feature $\{\Delta r_i\}$

$$\mathbf{M}_\sigma^j = P(\sigma_j | \Delta r_{1j}, \Delta r_{2j}, \Delta r_{3j}). \tag{5}$$

The SVM parameters are optimized via cross validation. Inferring a 1-D kernel size in a histogram bin is done by

$$\sigma_j = \arg\max P\left(\sigma | \mathbf{M}_\sigma^j, \Delta r_{1j}, \Delta r_{2j}, \Delta r_{3j}\right), \quad j = 1, \ldots, 180. \tag{6}$$

D. 2-D Kernel Shape Estimation

In this step, we estimate a 2-D elliptical kernel shape based on the series of 1-D kernel sizes inferred for the 180 histogram bins. Let $P_j = (\sigma_j, j)$ denote the 1-D kernel size in jth bin. By converting (σ_j, j) to the Euclidean space, we obtain

$$x_j = \sigma_j \cos(j), \quad y_j = \sigma_j \sin(j). \tag{7}$$

We then use the 180 Euclidean coordinates (x_j, y_j)s to approximate an ellipse (shown in Fig. 6). Therefore, via ellipse fitting we can determine the main direction of the blur, and also the intensity

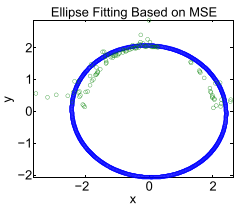

Fig. 6. Illustrating the principal kernel estimation via ellipse fitting. Green points are the estimated directional kernel sizes while blue is the fitted ellipse.

of the blur along the main direction and its orthogonal direction. The parametric expression of an ellipse is

$$ax^2 + 2cxy + by^2 = 1. \tag{8}$$

Stacking the ellipse equations of all (x_j, y_j)s yields

$$\begin{bmatrix} x_1^2 & 2x_1y_1 & y_1^2 \\ x_2^2 & 2x_2y_2 & y_2^2 \\ \vdots & \vdots & \vdots \\ x_{180}^2 & 2x_{180}y_{180} & y_{180}^2 \end{bmatrix} \boldsymbol{\phi} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \tag{9}$$

where $\boldsymbol{\phi} = [a \ c \ b]^T$ are the ellipse parameters. This forms a typical overdetermined problem. We use least squares in conjunction with random sample consensus [16] to obtain a noise-free solution. The ellipse axes are then calculated from ϕ according to

$$\sigma_x = 2\sqrt{\frac{2}{|a+b+\sqrt{(a-b)^2+4c^2}|}}$$
$$\sigma_y = 2\sqrt{\frac{2}{|a+b-\sqrt{(a-b)^2+4c^2}|}}. \tag{10}$$

The axes (σ_x, σ_y) form a principal Gaussian blur kernel. Since the interval $[-3\sigma, 3\sigma]$ retains 99.73% of the Gaussian energy, for robustness, the final kernel size is determined by

$$k_x = 6\sigma_x + 1, \quad k_y = 6\sigma_y + 1. \tag{11}$$

Initialized with this kernel size, the actual blur kernel can be estimated using existing deblurring algorithms [2], [7].

III. Experiments

A. Experimental Settings

MATLAB implementation of our method takes 6 s to estimate the motion kernel size of a 1024×768 blurry image on a PC with an Intel Q9550 3-GHz CPU. We evaluate the presented approach on both synthesized motion blurs and real blurry photographs with the following comparative strategies: 1) *Baseline 1*—a large kernel size 99×99 and 2) *Baseline 2*—a moderate kernel size 59×59. The synthetic blurs are generated by blurring 80 sharp images [17] with eight randomly generated motion kernels [5]. After kernel size estimation, two state-of-the-art deblurring algorithms [7][1] and [2][2] are used to estimate the actual motion kernel. For the synthetic blurry images, since their sharp correspondents are known, we quantitatively measure estimated kernel size in terms of the peak signal-to-noise ratio (PSNR) [18] and structural similarity (SSIM) [19] criteria. For the real blurry photographs, only qualitative measurements are performed. Fivefold cross validation is used to determine the linear SVM parameters from the range of $\sigma = 2$, $\gamma \in (2^{-9}, 2^9)$, $C \in (2^{-7}, 2^7)$, and $\epsilon \in (0, 5)$.

[1]We use the binary executable code provided by the authors.
[2]http://cs.nyu.edu//∼dilip/research/blind-deconvolution/

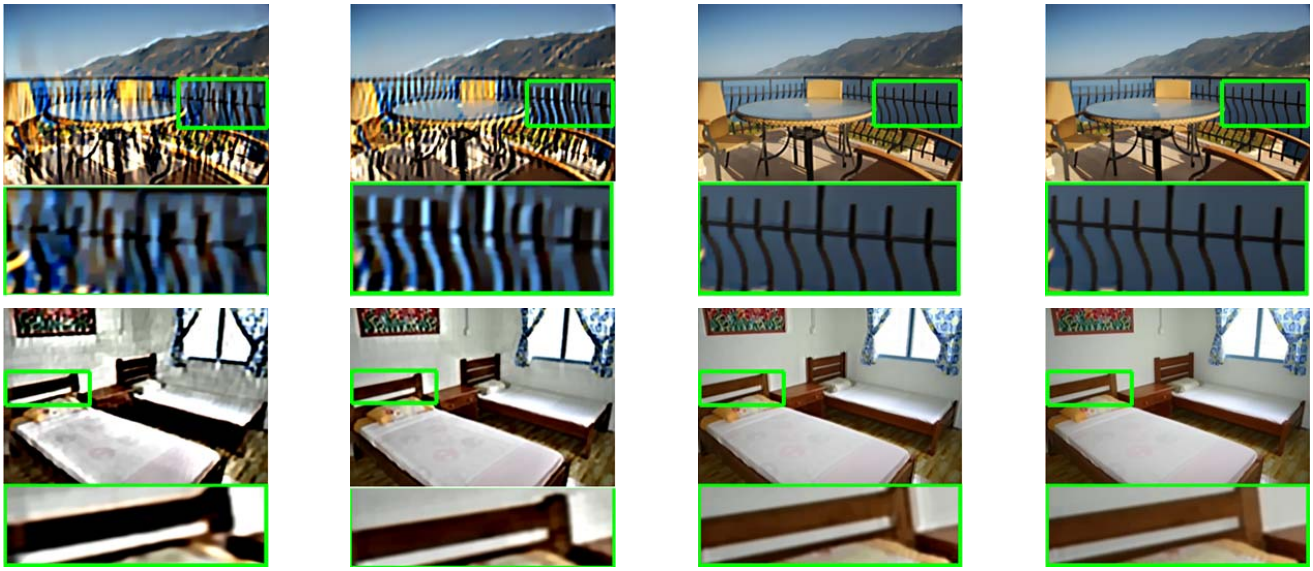

Fig. 7. **Top row**: deblurring a blurry balcony image using Cho and Lee's algorithm [7]. **Bottom row**: deblurring a blurry hotel image using Krishnan's algorithm [2]. (b) Using *Baseline 1*. (c) Using *Baseline 2*. (c) Using our *Estimated* kernel size. (d) Corresponding sharp image.

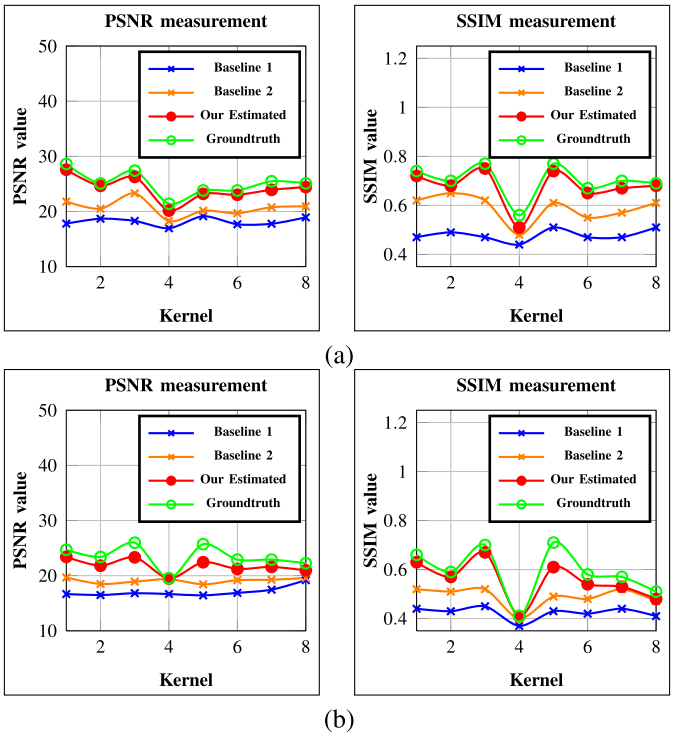

Fig. 8. Comparing the PSNR and SSIM values when different motion kernel sizes are used. x-axis is the blur kernel index. (a) Deblurring with Cho and Lee's algorithm [7]. (b) Deblurring with Krishnan's algorithm [2].

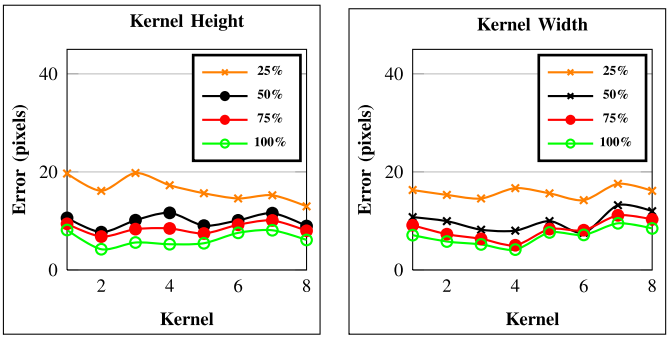

Fig. 9. Kernel size estimation error when different percentages of the 1600 sharp training are used.

B. Results on Synthetic Data

Fig. 7 shows the deblurring results of a blurry balcony image and a blurry hotel image using the algorithms in [2] and [7]. Different kernel sizes are sequentially applied to initialize the two deblurring algorithms. For the balcony image, both baseline sizes lead to a severe ringing effect, especially in the highlighted fence region. In contrast, the restored image using the kernel size estimated by the presented approach is visibly not different to the corresponding sharp ground truth. For the hotel image, similar results can be observed in Fig. 7. Using the estimated kernel size succeeds in recovering the sharp details of the ground-truth image.

Fig. 8 shows the corresponding mean PSNR and SSIM values of using the different kernel sizes. In conjunction with Cho and Lee's deblurring algorithm [7], the PSNR and SSIM values of using our estimated kernel size are closely similar to those of using the true kernel size. On the other hand, using the large baseline size (99×99) leads to worst PSNR and SSIM values. Using a moderate baseline size (59×59) yields better results, which are still apparently worse than using the kernel size estimated with the presented approach.

Fig. 9 shows errors of the estimated kernel sizes when 25%, 50%, 75%, and 100% of the 1600 sharp training images are used, respectively. The error is calculated by comparing the estimated kernel size with the ground-truth size. Low estimation errors are obtained using more than 50% of the data set.

C. Results on Real Blurry Photographs

Fig. 10 shows the deblurring results of two real blurry photographs. On the deblurred results of the poster image (top row), using the

Fig. 10. **Top row**: deblurring a blurry poster using Cho and Lee's algorithm [7]. **Bottom row**: deblurring a blurry book cover using Krishnan's algorithm [2]. From left are the deblurring results with (a) *Baseline 1*, (b) *Baseline 2*, and (c) *Estimated*.

baseline sizes causes severe ringing effect on the deblurred character region. In contrast, using the kernel size estimated by the presented approach successfully removes the blurriness of the characters, without introducing ringing artifacts. Similar results can be observed on the book image (bottom row). Using the baseline kernel sizes causes ringing artifact on the boundaries of the restored characters. The characters restored using the estimated kernel size look perceptually more natural.

IV. Conclusion

We have proposed an approach for estimating motion-blur kernel size based on the statistics of image pyramids. This approach is based on the phenomenon that at lower levels of an image pyramid, the HOG features of a blurry image are similar to those of the corresponding sharp image. By learning a set of directional regression models from a large data set, the kernel size of a blurry image can be estimated without knowing the sharp latent image. Experimental results show that the proposed method can estimate an accurate kernel size given a blurry input image, which is essential for the existing image deblurring algorithms to achieve good performance.

References

[1] Q. Shan, J. Jia, and A. Agarwala, "High-quality motion deblurring from a single image," *ACM Trans. Graph.*, vol. 27, no. 3, 2006, Art. ID 73.
[2] D. Krishnan, T. Tay, and R. Fergus, "Blind deconvolution using a normalized sparsity measure," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 233–240.
[3] L. Xu, S. Zheng, and J. Jia, "Unnatural L_0 sparse representation for natural image deblurring," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 1107–1114.
[4] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman, "Removing camera shake from a single photograph," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 787–794, 2006.
[5] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, "Understanding and evaluating blind deconvolution algorithms," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 1964–1971.
[6] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, "Efficient marginal likelihood optimization in blind deconvolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 2657–2664.
[7] S. Cho and S. Lee, "Fast motion deblurring," *ACM Trans. Graph.*, vol. 28, no. 5, 2009, Art. ID 145.
[8] N. Joshi, R. Szeliski, and D. Kriegman, "PSF estimation using sharp edge prediction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.
[9] L. Xu and J. Jia, "Two-phase kernel estimation for robust motion deblurring," in *Proc. 11th Eur. Conf. Comput. Vis.*, 2010, pp. 157–170.
[10] J. Biemond, A. M. Tekalp, and R. L. Lagendijk, "Maximum likelihood image and blur identification: A unifying approach," *Opt. Eng.*, vol. 29, no. 5, pp. 422–435, 1990.
[11] Y.-L. You and M. Kaveh, "A regularization approach to joint blur identification and image restoration," *IEEE Trans. Image Process.*, vol. 5, no. 3, pp. 416–428, Mar. 1996.
[12] J. Chen, L. Yuan, C.-K. Tang, and L. Quan, "Robust dual motion deblurring," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.
[13] D. Zoran and Y. Weiss, "From learning models of natural image patches to whole image restoration," in *Proc. IEEE Int. Conf. Comput. Vis.*, Nov. 2011, pp. 479–486.
[14] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, pp. 886–893.
[15] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, 2011, Art. ID 27.
[16] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.
[17] L. Sun and J. Hays, "Super-resolution from Internet-scale scene matching," in *Proc. IEEE Int. Conf. Comput. Photogr.*, Apr. 2012, pp. 1–12.
[18] *Peak Signal-to-Noise Ratio*. [Online]. Available: https://en.wikipedia.org/wiki/Peak_signal-to-noise_ratio.
[19] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.