Fast Motion-Deblurring of IR Images

Nisha Varghese, Mahesh Mohan M. R., and A. N. Rajagopalan, *Senior Member, IEEE*

***Abstract*—Camera gimbal systems pervade various applications such as navigation, target tracking, security and surveillance. The need for higher steering rate (rotation angle per second) of gimbal often results in motion blur in the captured video frames. Motion deblurring in real-time is difficult with existing blind restoration methods which incur large execution times while attempting to retrieve latent images from blurry inputs using high-dimensional optimization. On the other hand, deep learning methods for motion deblurring, though fast, do not generalize satisfactorily with domain shifts. In this work, we address the problem of real-time motion deblurring in infrared (IR) images captured by a real gimbal-based system. We propose two blur-kernel estimation methods and reveal how *a priori* knowledge of the blur-kernel can be used in conjunction with non-blind deblurring methods to achieve real-time performance. We experimentally show that, in comparison to the state-of-the-art techniques in deblurring, our method is better-suited for practical gimbal-based imaging systems.**

***Index Terms*—Blur-kernel, gimbal based imaging, IR deblurring dataset, real-time motion deblurring, video surveillance.**

I. Introduction

CAMERA gimbal systems [1] used in navigation, target tracking, security and surveillance, work by controlling the trajectory of a camera-mounted on a gimbal in order to obtain an extended field-of-view (FOV). During exposure, due to relative motion between scene and the gimbal, the captured images are typically degraded by motion blur. This can, in turn, lead to low probability of threat detection and higher false alarm rate during tracking and surveillance [2]–[4]. Since most of the applications require blur-free images, motion deblurring is a very important problem [5].

A number of traditional methods exist for blind motion deblurring in which *both* the blur-kernel and the latent image are estimated from the blurry input. The problem of deblurring is an ill-posed one, and sharp image estimation typically necessitates the use of natural image priors [5], [6], [17], [19]. The methods of [7] and [8] accommodate space-variant blur-kernels but their high computational cost renders them ineffective for real-time applications. The method of [9] is fast but is restricted to mild blur. Gimbal motion is predominantly a 1D rotational yaw motion and the blur-kernel can be assumed to be space-invariant [10]. For space-invariant deblurring, we can model the blurred image $\mathbf{B}$ as a clean image $\mathbf{L}$ convolved with a blur-kernel $\mathbf{k}$ and corrupted by noise $\mathbf{n}$ [5], [15]–[21]:

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

where $*$ denotes convolution. There are several works on space-invariant blind deblurring [15]–[20] for recovering both the blur-kernel and latent image. State-of-the-art deblurring methods are designed for an unconstrained system such as random hand-held motion. Consequently, the number of unknowns and hence the processing time for these traditional methods is unacceptably high as they do not leverage the underlying motion model in gimbal-based systems where camera motion is usually constrained or is tractable.

The task of deblurring an image with a known blur-kernel is known as *non-blind* deblurring. Numerous non-blind deconvolution approaches exist, varying greatly in their speed and sophistication [21]–[27]. Since the problem is only to estimate clean image $\mathbf{L}$ given the blurred image $\mathbf{B}$ and the blur-kernel $\mathbf{k}$, the processing time for non-blind deconvolution is comparatively far less than any blind deblurring method.

Another class of methods use deep learning (DL) networks. As these methods typically require only a single pass over the network for deblurring, their processing time is comparatively quite less [2], [30]–[36]. However, DL networks need a realistic training dataset with a large number of blurred and corresponding deblurred image-pairs or many unpaired sharp and blurred real-world images [37]. This requirement is often difficult to meet practically, especially in the case of IR images. Further, one pertinent problem is regarding generalization, i.e., a network trained using a particular dataset struggles when confronted with unseen examples from other datasets even if the domain shifts are small.

In this work, we deal with IR image deblurring and propose two effective blur-kernel estimation methods for real-time motion deblurring of images captured using gimbal-based systems. In our approach, blur-kernel estimation is followed by *non-blind* deblurring for achieving real-time performance. While we have considered a gimbal set-up which extends the FOV via yaw-based gimbal motion, our work can be extended to other gimbal trajectories also. Our main contributions are as follows:

- We provide a mathematical framework to estimate the blur-kernel in gimbal-based IR imaging systems, and leverage it to devise an effective non-blind deblurring solution that is almost real-time.
- We experimentally verify that our method outperforms competing methods by a significant margin via extensive evaluations with *real* gimbal data.
- Our model can be used to create large-scale motion blur datasets for gimbal-based IR systems that can be harnessed by deep learning methods.

Manuscript received October 14, 2021; revised December 20, 2021; accepted December 28, 2021. Date of publication January 5, 2022; date of current version January 28, 2022. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Yue Wu. *(Corresponding author: Nisha Varghese.)*

The authors are with the Department of Electrical Engineering, Indian Institute of Technology Madras, Chennai 600036, India (e-mail: ee19d750@smail.iitm.ac.in; ee14d023@ee.iitm.ac.in; raju@ee.iitm.ac.in).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LSP.2022.3140685, provided by the authors.

Digital Object Identifier 10.1109/LSP.2022.3140685

II. The Proposed Approach

Our central idea is that if we know the blur-kernel for a blurred video-frame, then we can speed-up deblurring by a large factor by using that blur-kernel within a non-blind deblurring framework. Among the many existing non-blind deblurring methods, we zero-in on three popular methods, namely, Wiener filter (WF) [28], Richardson-Lucy (RL) deconvolution [22], [23] and deblurring using hyper-Laplacian priors [24] due to their attractive execution times. Non-blind deblurring methods yield good results provided the blur-kernel is accurate [21]–[27]. Towards this end, we propose two blur-kernel estimation methods for the problem on hand.

A. Blur-Kernel Estimation Using Blur-Sharp Pair

The strength of this method lies in the fact that it does not warrant knowledge of motion or camera parameters. It is well-known that temporal averaging of a number of consecutive sharp images, captured from a camera, yields a blurred image that resembles a motion blurred image. To get sharp images from the gimbal-system, the camera is panned very slowly with a lower steering rate, say s_{r_low} deg/sec, so as to obtain blur-free images of the scene. Blurred images corresponding to different steering rates can then be obtained by suitably averaging N ($N \in \{1, 2, 3, \ldots, 13\}$) number of sharp frames from s_{r_low} deg/sec. The center frame is taken as the sharp image. The blurred image $\mathbf{B}$ along with its sharp image counterpart $\mathbf{L}$ can be used to estimate the blur-kernel $\mathbf{k}$ corresponding to the motion blur content in the blurred image by solving Eq. (2) using conjugate gradient.

$$\mathbf{k} = \arg\min_{\mathbf{k}} ||\mathbf{B} - \mathbf{L} * \mathbf{k}||_2 \quad \text{s. t.} \quad \sum \mathbf{k} = 1 \tag{2}$$

This method works for any type of blur-kernel (refer to supplementary material (Section S2)). Eq. (2) takes an initial guess for $\mathbf{k}$ and refines it over iterations. A uniform kernel is a reasonable choice as an initial estimate of $\mathbf{k}$.

The blur-kernels obtained by averaging frames for different values of N correspond to different steering rates. The value of N can be found either empirically by matching the deblurred results for different steering rates or it can be derived if the exposure time t_{exp} of the camera is known. The rotation angle of a camera with steering rate s_r (in deg/sec) and exposure time t_{exp} is given by $\theta_{s_r} = t_{exp} \cdot s_r$. A steering rate of s_{r_low} deg/sec covers 1° in $1/(s_{r\_low})$ sec. If f_r is the frame rate (in frames/sec) then, the number of frames is given by

$$N = \frac{f_r \cdot \theta_{s_r}}{s_{r\_low}} \tag{3}$$

B. Analytical Modeling of Blur-Kernel

In the previous section, we showed how the blur-kernel can be estimated from a blur-sharp pair. In this section, as an alternative method, we propose to find the blur-kernel mathematically for the case of gimbal motion that pans a scene with an IR mounted camera with known intrinsic camera parameters. This analytical approach does not need any blur-sharp image pairs.

For this purpose, we first assume a maximum pixel spread s (for each steering rate), from the center of the image, due to camera motion. We next find the rotation angle of the camera corresponding to every pixel displacement upto the maximum pixel displacement s. Then we find the transformation of the center pixel value using homography equations for every rotation angle of the camera. We first derive the equation for the rotation angle corresponding to a pixel spread that occurs in the image due to yaw-rotation induced motion blur.

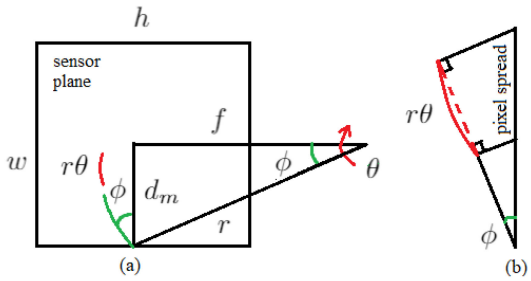


Fig. 1. Relation between rotation angle and pixel spread.

1) Rotation Angle θ Corresponding to a Pixel Spread: The situation on hand is illustrated in Fig. 1. The camera is rotated by an angle θ where w and h are the dimensions of the image, f is the focal length of the camera, d_m is the maximum displacement from the center of the image, and r is the distance from the edge pixel point to camera center.

When the camera rotates by an angle θ, an arc with length $r\theta$ will be formed in the rear of the sensor plane whose image is the pixel spread on the image plane. Assuming a linear arc length $r\theta$, we can write the arc length as the projection from the sensor plane (as seen in Fig. 1(b)). Therefore,

$$r\theta = p_s \cdot \cos\phi \tag{4}$$

where ϕ is the angle between the arc and the sensor plane, and p_s is the corresponding pixel spread. From Fig. 1(a), it can be seen that the angle between the arc and the sensor plane is equal to the angle between f and r. Therefore $\cos\phi$ can be written as $\cos\phi = f/r$ and r can be written in terms of f and d_m as $r = \sqrt{f^2 + d_m^2}$. Hence from Eq. (4), the rotation angle θ corresponding to the pixel spread can be written as

$$\theta = p_s \cdot \left(\frac{f}{f^2 + d_m^2}\right) \tag{5}$$

We next list the steps to find the blur-kernel corresponding to the pan motion of the camera given the intrinsic and motion parameters of the camera.

1) The intrinsic camera matrix $\mathbf{K} = \begin{bmatrix} f & 0 & w/2 \\ 0 & f & h/2 \\ 0 & 0 & 1 \end{bmatrix}$.
   If the focal length (f) of the camera is not known, then it can be found out using FOV α and the frame dimensions w and h as
   $$f = \frac{\sqrt{h^2 + w^2}}{2\tan(\alpha/2)} \tag{6}$$
2) Find the rotation angles of the camera θ_1 and θ_{max} corresponding to one pixel spread and maximum pixel spread s respectively, using Eq. (5).
3) Since we consider rotation about the y axis, the rotation matrix $\mathbf{R}$ corresponding to the rotation angle θ can be written as
   $$\mathbf{R} = \mathbf{R}_y = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{7}$$

Find the rotation matrices for the rotation angles from $-\theta_{max}$ to $+\theta_{max}$ with step size θ_1.

4) For pure rotation model, homography $\mathbf{H}$ [7] can be written as $\mathbf{H} = \mathbf{KRK}^{-1}$. Find all the $\mathbf{H}$ matrices corresponding to every rotation obtained in step 3.
5) For the center pixel value **x**, find its transformed point **x'** $= \mathbf{H}$**x** for all $\mathbf{H}$ and mark 1 at every **x'** if **x'** is an integer. If **x'** is not an integer, then distribute 1 to the neighbouring positions of **x'** using bilinear interpolation and normalise it so that it sums to 1.
6) If we are interested in obtaining blur-kernels corresponding to different positions (other than center) of the image, repeat step 5 for the other pixel locations.

It is straightforward to extend this to general gimbal-trajectories.

2) Selection of Maximum Pixel Spread: If the exposure time t_{exp} is known, then the maximum spread s for a steering rate s_r rad/sec can be determined a priori as follows.

The total angle of rotation ϕ' during the exposure time can be written as $\phi' = s_r \cdot t_{exp}$. Then by using Eq. (5), we can find the maximum pixel spread s from the center as

$$s = \left(\frac{\phi'}{2}\right) \cdot \left(\frac{f^2 + d_m^2}{f}\right) \tag{8}$$

The blur-kernel for each steering rate can be computed in advance using our proposed methods (Section II-A and II-B) and can be stored in a Look-Up-Table (LUT). For a given steering rate of the input video, the corresponding blur-kernel can be obtained from LUT and can be used in non-blind methods to get the deblurred output.

III. Experimental Results

In this section, we demonstrate the results of our proposed blur-kernel estimation i) from blur-sharp pair (Section II-A), and ii) analytically (Section II-B) followed by standard non-blind deblurring methods WF [28], RL [22], [23] and [24]. We also analyse their real-time performance. For comparisons, we consider a recent blind deblurring method [20] and a deep learning method ESTRNN [35]. We also consider the blur-kernel returned from [20] and use it for non-blind deblurring.

A. Datasets and Implementation Details

We present results on two datasets (Dataset1 and Dataset2) with real IR images, captured from an actual gimbal system that pans a scene at 30 frames/sec. Both the datasets contain videos of steering rates 1 deg/sec and 10 to 60 deg/sec with a step-size of 10 deg/sec. Dataset2 contains an extra video of steering rate 0.5 deg/sec. Both of our real video datasets inherently contain some noise. For higher steering rates, Dataset2 contains videos which are comparatively more noisy than Dataset1. The intrinsic parameters of the camera along with the values chosen for N and s (Section II) for each steering rate for both the datasets are given in supplementary material (Section S3). Using these parameters, the blur-kernels are estimated from the proposed methods for different steering rates, and for both the datasets independently. The corresponding kernels are stored in an LUT. For a fair comparison with the DL method, we finetune ESTRNN [35] on IR images. We denote them as ESTRNN1 and ESTRNN2 (trained on image deblurring dataset consisting of blur-sharp image pairs formed (see Section II) from Dataset1 and Dataset2, respectively). We perform deblurring with Intel Xeon e5-2630v4 processor @ 2.2 GHz.

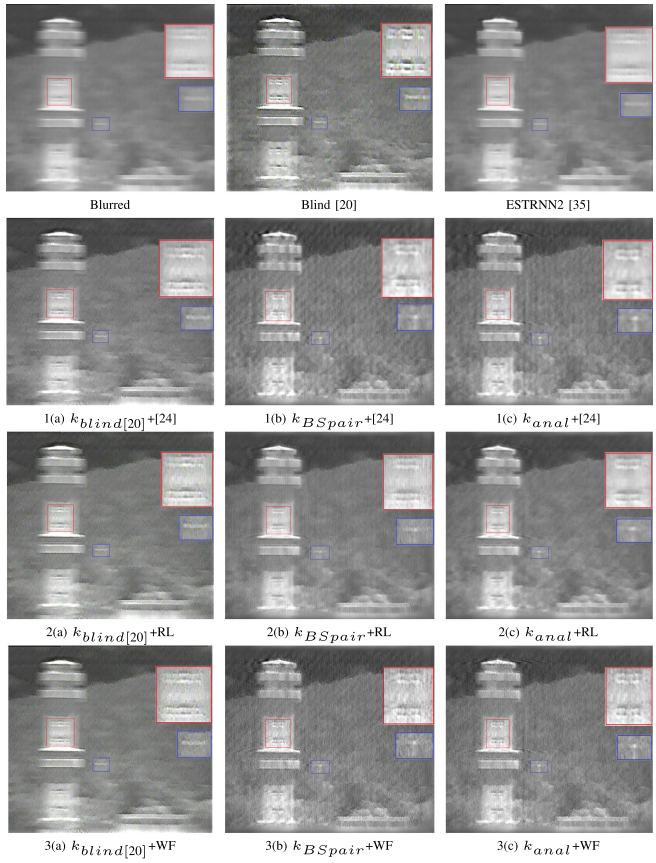


Fig. 2. The blurred image and deblurred results of blind method [20] and DL network ESTRNN2 [35] are given in the first row. Comparisons of non-blind deblurring methods (1. Krishnan *et al.* [24], 2. RL, 3. WF) for steering rate of 60 deg/sec using blur-kernels from three different methods (a: from blind method [20], b: estimated using blur-sharp pair, c: analytical blur-kernel). Note that our methods (b) and (c) deliver the best performance consistently.

B. Qualitative Analysis

We use the following nomenclature for blur-kernels.

a) $k_{blind[20]}$: Blur-kernel estimated using blind method [20].
b) k_{BSpair}: Blur-kernel as obtained from LUT.
c) k_{anal}: Analytical blur-kernel from LUT.

In Fig. 2, we give comparison results for a real blurred image from Dataset1 corresponding to steering rate of 60 deg/sec. The result of blind method [20] is not correct as it outputs 3 squares in a row (see the red zoomed inset on the tower) when actually there are only 2 squares in a row in that region. Also, the processing time is about 5 minutes/image which precludes it for real-time processing. Even though ESTRNN2 [35] takes only 30 ms/image, its deblurring performance is poor as it was trained on Dataset2. This underscores a known fact that DL networks are susceptible to domain-dependence, even between two different IR datasets. From Fig. 2, it is amply evident that our methods (b and c) yield best performance consistently across WF, RL as well as [24]. Only the proposed methods reconstruct the four small squares in the zoomed portion of the tower (see red zoomed inset) perfectly. The tip of a branch in the image is also deblurred to a dot (see the blue zoomed inset) whereas the competing methods struggle to deblur these small

TABLE I
QUANTITATIVE EVALUATIONS USING PSNR/SSIM & NIQE/PIQUE/BRISQUE AND PROCESSING TIME FOR NBD (NON-BLIND DEBLURRING) METHODS

	Method	PSNR(dB)/SSIM↑			NIQE/PIQUE/BRISQUE↓		
Dataset1	**Blind [20]**	28.5/0.78			9.3/43.8/40.1		
	ESTRNN2 [35]	28.4/0.77			9.7/44.8/46.1		
	NBD	**[24]**	**RL**	**WF**	**[24]**	**RL**	**WF**
	$k_{blind[20]}$	28.3/0.77	28.3/0.76	28.1/0.73	10.7/41.3/43.5	10.6/44.4/43.5	10.6/41.2/43.4
	Ours: k_{BSpair}	29.7/0.82	29.5/0.81	29.8/0.83	9.0/40.9/38.4	10.5/42.7/43.4	8.7/42.5/43.0
	Ours: k_{anal}	30.7/0.86	30.1/0.85	31.0/0.88	5.1/40.1/39.3	8.0/25.8/38.5	8.1/40.3/43.2
Dataset2	**Blind [20]**	30.6/0.75			6.7/39.8/38.3		
	ESTRNN1 [35]	30.2/0.69			5.0/29.7/27.3		
	NBD	**[24]**	**RL**	**WF**	**[24]**	**RL**	**WF**
	$k_{blind[20]}$	29.8/0.69	28.5/0.67	28.5/0.68	5.4/23.6/35.0	6.8/25.1/24.4	8.7/28.6/35.8
	Ours: k_{BSpair}	35.3/0.87	35.8/0.86	35.5/0.87	4.8/34.1/34.8	6.9/28.8/20.4	5.5/23.2/20.5
	Ours: k_{anal}	35.4/0.88	36.3/0.87	35.6/0.84	4.5/23.6/31.1	5.1/18.1/14.7	5.3/20.8/20.9

Processing time per frame (sec)			
NBD	**[24]**	**RL**	**WF**
Using single core of processor	0.3	0.24	0.04
Using Parallel processing (in CPU machine)			
Real-time video (30 frames)	0.090	0.075	0.014
Real-time visualization (312 frames)	0.040	0.069	0.011

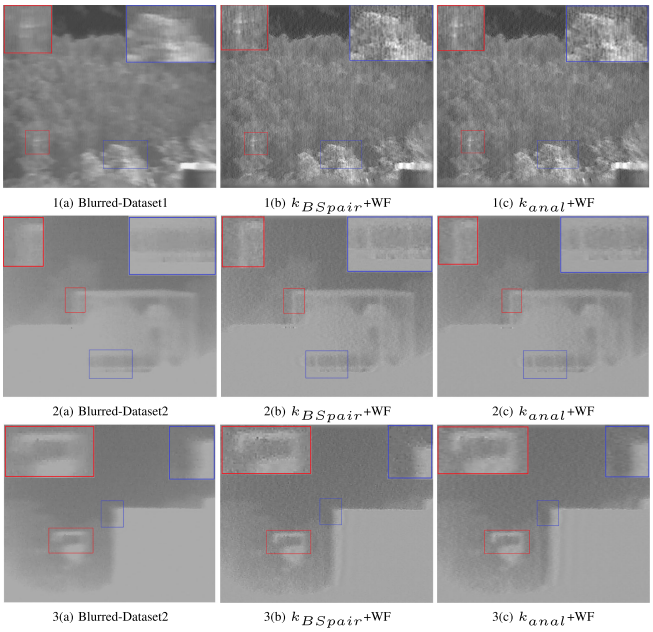


Fig. 3. Outputs of our methods ((b): k_{BSpair}+WF and (c): k_{anal}+WF) for blurred images from 1. Dataset1 (50 deg/sec), 2. Dataset2 (60 deg/sec) and 3. Dataset2 (50 deg/sec). Note that both our methods perform consistently well.

structures. Our results are better than $k_{blind[20]}$ also. Visually, motion deblurring quality is best by using k_{anal} with WF. In Fig. 3, deblurred results for one more real blurred image from Dataset1 and two real blurred images from Dataset2, using our proposed methods k_{BSpair}+WF and k_{anal}+WF are given. From the figure, it can be seen that deblurring quality of our methods is again quite good. In the first row, the branches and leaves are sharply visible; in the second row, edges, especially the thin vertical bars (given in the blue zoomed inset) emerge clearly; and in the third row, vertical edges of the structure (given in zoomed insets) have very little smear. Comparison results of our best method k_{anal}+WF with ESTRNN on more real blurred images are given in the supplementary material (Section S1).

C. Quantitative Comparisons and Time-Complexity

In order to quantitatively evaluate efficacy, from each dataset, we generated 20 blurred images (10 images corresponding to 50 deg/sec and 10 images at 60 deg/sec steering rate) by temporally averaging the lowest steering rate images (1 deg/sec for Dataset1 and 0.5 deg/sec for Dataset2). The center frame from the temporally averaged sequence was treated as the clean image, although, strictly speaking, in our captured videos, even the images corresponding to the lowest steering rate are not noise-free. These images are then deblurred using blind [20], ESTRNN [35] as well as non-blind methods (using $k_{blind[20]}$, k_{BSpair}, and k_{anal}). The average PSNR and SSIM values (higher is better) for these methods are given in Table I. For both the datasets, our methods yield significantly higher values.

For deblurring real images, since ground truth is not available, we use no-reference image quality scores such as naturalness image quality evaluator (NIQE) [38], perception based image quality evaluator (PIQUE) [39] and blind/referenceless image spatial quality evaluator (BRISQUE) [40] (lower is better). The average scores for each dataset computed over 20 real deblurred images (10 images each from steering rates of 50 deg/sec and 60 deg/sec) are also given in Table I. It can be observed that our proposed method (especially k_{anal}) comfortably outperforms [20], ESTRNN [35] and $k_{blind[20]}$.

With a known blur-kernel, and by using a single core of the processor, per image, the processing time of non-blind deblurring methods [24], RL and WF are given in the last column of Table I. Even with a single core, WF delivers real-time performance (0.04 sec/frame). For further speed up, with 24 parallel workers, the processing time for a real-time blurred video input (assuming 30 frames/sec) and for real-time visualization of deblurred results of an already available blurred video (assuming 312 frames) are also given in Table I. WF achieves excellent real-time performance for both the cases. For the case of 312 frames, both WF and [24] achieve real-time performance.

In summary, blur-kernel estimated using our proposed methods followed by Wiener deconvolution is optimal both in terms of deblurring quality and real-time processing requirement.

Additional Remarks: Our work can be leveraged to create large-scale IR datasets with realistic gimbal motion blur (an example is given in Section S4 in supplementary material). Such datasets, which are a rarity, can be a valuable asset for contemporary deep learning methods.

IV. CONCLUSION

In this work, we dealt with the problem of real-time motion deblurring in gimbal-based systems. Our studies on real IR datasets captured from an actual gimbal system revealed that existing methods fall significantly short of delivering real-time deblurring performance. We modeled the motion blur-kernel using a low steering rate dataset as well as analytically with the knowledge of camera parameters. We used the estimated kernels in conjunction with non-blind deblurring methods to deliver excellent performance both in terms of deblurring quality and real-time processing.

REFERENCES

[1] K. Jedrasiak, D. Bereska, and A. Nawrat, "The prototype of gyro-stabilized UAV gimbal for day-night surveillance," in *Advanced Technologies for Intelligent Systems of National Border Security*, vol 440, A. Nawrat, K. Simek, and A. Świerniak Eds. Berlin, Germany: Springer, 2013, pp. 107–115.

[2] O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas, "DeblurGAN: Blind motion deblurring using conditional adversarial networks," in *Proc. Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 8183–8192.

[3] I. Vasiljevic, A. Chakrabarti, and G. Shakhnarovich, "Examining the impact of blur on recognition by convolutional networks," 2016, *arXiv:1611.05760*. [Online]. Available: https://arxiv.org/abs/1611.05760

[4] S. Dodge and L. Karam, "Understanding how image quality affects deep neural networks," in *Proc. IEEE 8th Int. Conf. Qual. Multimedia Experience*, 2016, pp. 1–6.

[5] A. N. Rajagopalan and R. Chellappa, *Motion Deblurring: Algorithms and Systems*. Cambridge, MA, USA: Cambridge Univ. Press, 2014.

[6] J. Pan, D. Sun, P. Hanspeter, and Y. Ming-Hsuan, "Blind image deblurring using dark channel prior," in *Proc. Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 1628–1636.

[7] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce, "Non-uniform deblurring for shaken images," *Int. J. Comput. Vision*, vol. 98, no. 2, pp. 168–186, Jun. 2012.

[8] 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 *Proc. Eur. Conf. Comput. Vision*, 2012, pp. 27–40.

[9] M. Delbracio, I. Garcia-Dorado, S. Choi, D. Kelly, and P. Milanfar, "Polyblur: Removing mild blur by polynomial reblurring," *IEEE Trans. Comput. Image*, vol. 7, pp. 837–848, 2021, doi: 10.1109/TCI.2021.3100998.

[10] A. Levin, Y. Weiss, and F. Durand, "Understanding and evaluating blind deconvolution algorithms," in *Proc. Conf. Comput. Vision Pattern Recognit.*, 2009, pp. 1964–1971.

[11] T. Michaeli and M. Irani, "Blind deblurring using internal patch recurrence," in *Proc. Eur. Conf. Comput. Vision*, 2014, pp. 783–798.

[12] L. Xiao, J. Wang, W. Heidrich, and M. Hirsch, "Learning high-order filters for efficient blind deconvolution of document photographs," in *Proc. Eur. Conf. Comput. Vision*, 2016, pp. 734–749.

[13] L. Xu and J. Jia, "Two-phase kernel estimation for robust motion deblurring," in *Proc. Eur. Conf. Comput. Vision*, 2010, pp. 157–170.

[14] L. Xu, S. Zheng, and J. Jia, "Unnatural L0 sparse representation for natural image deblurring," in *Proc. Conf. Comput. Vision Pattern Recognit.*, 2013, pp. 1107–1114.

[15] R. Fergus, S. Barun, 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.

[16] J. Dong, J. Pan, Z. Su, Yang, and Ming-Hsuan, "Blind image deblurring with outlier handling," in *Proc. Int. Conf. Comput. Vision*, 2017, pp. 2478–2486.

[17] Y. Yan, W. Ren, Y. Guo, R. Wang, and X. Cao, "Image deblurring via extreme channels prior," in *Proc. Conf. Comput. Vision Pattern Recognit.*, 2017, pp 4003–4011.

[18] Y. Liu, W. Dong, D. Gong, L. Zhang, and Q. Shi, "Deblurring natural image using super-Gaussian fields," in *Proc. Eur. Conf. Comput. Vision*, 2018, pp. 467–484.

[19] F. Wen, R. Ying, Y. Liu, P. Liu, and T. -K. Truong, "A simple local minimal intensity prior and an improved algorithm for blind image deblurring," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 8, pp. 2923–2937, Aug. 2020.

[20] M. Jin, S. Roth, and P. Favaro, "Normalized blind deconvolution," in *Proc. Eur. Conf. Comput. Vision*, 2018, pp. 694–711.

[21] S. Vasu, R. M., Venkatesh, and A. N. Rajagopalan, "Non-blind deblurring: Handling kernel uncertainty with CNNs," in *Proc. Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 3272–3281.

[22] W. Richardson, "Bayesian-based iterative method of image restoration," *J. Opt. Soc. Amer. A*, vol. 62, pp. 55–59, 1972.

[23] L. Lucy, "An iterative technique for the rectification of observed distributions," *J. Astron.*, vol. 79, pp. 745–754, 1974.

[24] D. Krishnan and R. Fergus, "Fast image deconvolution using hyper-Laplacian priors," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 1033–1041.

[25] D. Zoran and Y. Weiss, "From learning models of natural image patches to whole image restoration," in *Proc. Int. Conf. Comput. Vision*, 2011, pp. 479–486.

[26] S. Cho, J. Wang, and S. Lee, "Handling outliers in non-blind image deconvolution," in *Proc. Int. Conf. Comput. Vision*, 2011, pp. 495–502.

[27] H. Ji and K. Wang, "Robust image deblurring with an inaccurate blur kernel," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 1624–1634, Apr. 2012.

[28] R. C. Gonzalez, R. E. Woods, and B. R. Masters, *Digital Image Processing*, 2nd ed. Englewood Cliffs, NJ, USA: Prentice Hall, 2002, pp. 374–379.

[29] A. A. Sarawade and N. N. Charniya, "Infrared thermography and its applications: A review," in *Proc. 3rd Int. Conf. Commun. Electron. Syst.*, 2018, pp. 280–285.

[30] S. Ramakrishnan, S. Pachori, A. Gangopadhyay, and S. Raman, "Deep generative filter for motion deblurring," in *Proc. Int. Conf. Comput. Vision Workshops*, 2017, pp. 2993–3000.

[31] M. Suin, K. Purohit, and A. N. Rajagopalan, "Spatially-attentive patch-hierarchical network for adaptive motion deblurring," in *Proc. Conf. Comput. Vision Pattern Recognit.*, 2020, pp. 3606–3615.

[32] S. Zhang, Z. Zhu, J. Cheng, Y. Guo, and Y. Zhao, "Edge heuristic GAN for non-uniform blind deblurring," *IEEE Signal Process. Lett.*, vol. 26, no. 10, pp. 1546–1550, Oct. 2019.

[33] O. Kupyn, T. Martyniuk, J. Wu, and Z. Wang, "DeblurGAN-v2: Deblurring (orders-of-magnitude) faster and better," in *Proc. Int. Conf. Comput. Vision*, 2019, pp. 8878–8887.

[34] S. Nah, T. H. Kim, and K. M. Lee, "Deep multi-scale convolutional neural network for dynamic scene deblurring," in *Proc. Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 3883–3891.

[35] Z. Zhong, Y. Gao, Y. Zheng, and B. Zheng, "Efficient spatio-temporal recurrent neural network for video deblurring," in *Proc. Eur. Conf. Comput. Vision*, 2020, pp. 191–207.

[36] K. Purohit and A. N. Rajagopalan, "Region-adaptive dense network for efficient motion deblurring," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 11882–11889.

[37] K. Zhang *et al.*, "Deblurring by realistic blurring," in *Proc. Conf. Comput. Vision Pattern Recognit.*, 2020, pp. 2734–2743.

[38] A. Mittal, R. Soundararajan, and A. C. Bovik, "Making a completely blind image quality analyzer," *IEEE Signal Process. Lett.*, vol. 22, no. 3, pp. 209–212, Mar. 2013.

[39] N. Venkatanath, D. Praneeth, M. Bh, C. Channappayya, and S. S. Medasani, "Blind image quality evaluation using perception based features," in *Proc. Nat. Conf. Commun.*, 2015, pp. 1–6.

[40] A. Mittal, A. K. Moorthy, and A. C. Bovik, "No-reference image quality assessment in the spatial domain," *IEEE Trans. Image Process.*, vol. 21, no. 12, pp. 4695–4708, Dec. 2012.

Fast Motion-deblurring of IR images (Supplementary Material)

Nisha Varghese, Mahesh Mohan M. R., and A. N. Rajagopalan, *Senior Member, IEEE*

The figures and tables of this supplementary material are numbered using a prefix S, and are arranged as follows:

1. Qualitative comparisons of our best method k_{anal}+WF, with DL method ESTRNN [35], on some more images from Dataset1 and Dataset2.
2. Validation of algorithm for blur-kernel estimation using blur-sharp pair for any arbitrary kernel.
3. Camera parameters and the values of N and s (Sec. II) chosen for each steering rate for both the datasets.
4. An example of a synthesized blurred image.

S1. Qualitative Analysis

A comparison of deblurred results of our method k_{anal}+WF with ESTRNN [35] on two different images from both Dataset1 and Dataset2 are given in Fig. S1. It can be seen that our deblurring quality is distinctly better than ESTRNN (see the highlighted portions).

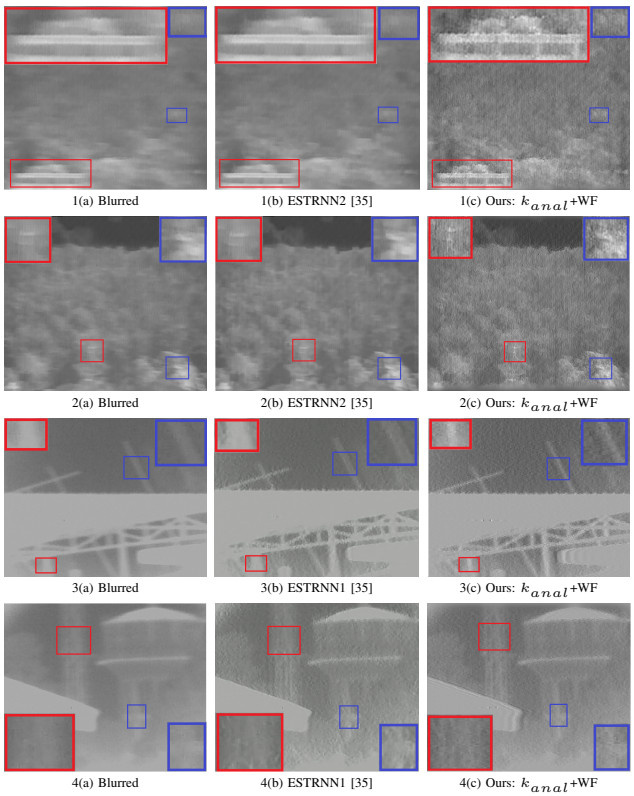
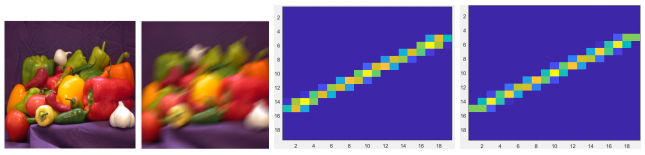


Fig. S1. Comparison of deblurred results of our method k_{anal}+WF (c) with ESTRNN [35] (b), on two images from both the datasets, Dataset1 (1-2) and Dataset2 (3-4).

S2. Blur-kernel estimation using blur-sharp pair

In this section, we revisit the blur-kernel estimation method used in Sec. II(A) and reveal that our method of blur-kernel estimation from a blur-sharp pair can be used for even arbitrary blur-kernels.



(a) Sharp (b) Blurred (c) Estimated kernel (d) Original kernel

Fig. S2. Comparison of estimated kernel (c) (from blur sharp pair (a-b)), and the original kernel (d). Note that the blur-kernel estimation method using blur-sharp pair estimates PSFs similar to the ground truth.

Figure S2 gives an example of blur-sharp pair used for PSF estimation. The blurred image in Fig. S2(b) is formed synthetically by blurring the sharp image (Fig. S2(a)) with a diagonal motion blur kernel of size 21 pixels at an angle of 30° (Fig. S2(d)). By solving Eq. 2, the estimated blur-kernel from the blur-sharp pair is given in Fig. S2(c). It can be seen that the estimated blur kernel is quite close to the original kernel.

S3. Camera parameters and values for N and s

As given in Sec. II, values of N (number of frames to be averaged) and s (maximum pixel spread) can be found out from camera and dataset parameters. The values thus obtained are given in Table S1.

TABLE S1
Camera/Dataset parameters and values obtained for N and s

Steering rate (°/sec)			10	20	30	40	50	60
Dataset1	FOV = 8°, t_{exp}= 5 ms, s_{r_low}= 1°/sec	N	2	3	5	6	8	9
	f_r= 30 frames/sec, w = 558, h = 481	s	3	5	7	10	12	14
Dataset2	FOV = 3.7°, t_{exp}= 2 ms, s_{r_low}= 0.5°/sec	N	2	3	4	5	6	8
	f_r= 30 frames/sec, w = 579, h = 461	s	2	4	6	8	10	12

S4. Blurred Image Synthesis

(a) Synthesized (sharp-blur pair) (b) Real - SR60 deg/sec

Fig. S3. An example of our synthesized blurred image (a) for the steering rate 60 deg/sec. Note that (a) looks quite similar to real blurred image (b).