SEMI-SUPERVISED LEARNING OF CAMERA MOTION FROM A BLURRED IMAGE

Nimisha T M[*], *Vijay Rengarajan*[†], *and Rajagopalan Ambasamudram*[*]

[*]Indian Institute of Technology Madras [†]Carnegie Mellon University

ABSTRACT

We address the problem of camera motion estimation from a single blurred image with the aid of deep convolutional neural networks. Unlike learning-based prior works that estimate a space-invariant blur kernel, we solve for the global camera motion which in turn represents the space-variant blur at each pixel. Leveraging the camera motion as well as the clean reference image during training, we resort to a semi-supervised training scheme that utilizes the strengths of both supervised and unsupervised learning to solve for the camera motion undergone by a space-variant blurred image. Finally, we show the effectiveness of such a motion estimation network with applications in space-variant deblurring and change detection.

Index Terms— Camera motion estimation, motion blur, deblurring, change detection, deep neural networks.

1. INTRODUCTION

Inferring camera motion from images is important in many tasks including robotic navigation, depth estimation, and splicing detection [1]. Oftentimes the camera motion is embedded in the captured image itself in the form of motion blur. With a boom in lightweight hand-held imaging devices, motion blur has become very prevalent, and it has been a major confrontation in long exposure photography.

Recent deep learning works [2, 3] consider the default use case of deblurring due to camera motion where a clean image devoid of motion blur is preferred by users. These works follow an end-to-end deblurring learning framework where the camera motion is not estimated at all. We observe that estimating camera motion that caused the blur has attracting characteristics in that it can be used for different applications such as change detection including that of deblurring. Therefore, in this paper, we intend to address the problem of single-image camera motion estimation, and we provide the applications of deblurring and change detection as its epilogues.

Classical non-learning based works do deal with camera motion estimation, but inherently they are tied to a specific application, deblurring in the case of [4, 5, 6], and change detection in the case of [7, 8]. In our work, we estimate the motion independent of these applications, and the user can choose to perform any task that leverages the camera motion post estimation.

We propose a deep convolutional neural network (CNN) that solves for the camera motion given a single motion blurred image. Given the fact that we have access to the ground truth camera motion as well as clear images during the training phase, we adopt a semi-supervised learning scheme. Our method benefits from ground-truth camera motion measurements as an unambiguous cue for supervised learning, and at the same time, it uses the latent clean image for an unsupervised prediction of camera motion. During the testing phase, the network takes in the blurred frame and outputs the camera motion undergone during the capturing time. The network learns space-variant blur in the form of inplane camera translations and inplane

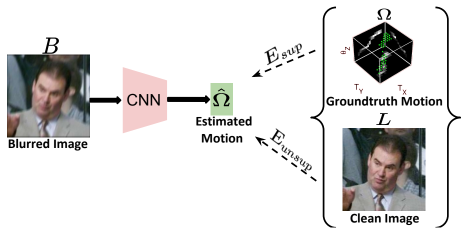


Fig. 1. Our network model utilizes the supervised motion cost as well as the unsupervised image cost for estimating global camera motion from a single blurred image.

rotations. The camera motion thus estimated can either be used to deblur the input image or be used to reblur the paired image to detect changes. A comprehensive diagram of our network and the cost functions used is provided in Fig. 1.

Related Works There is a vast literature on blind motion deblurring, scene registration, and change detection tasks. Inverting the blur to estimate the underlying camera motion and the latent image from a single blurred image is a highly ill-posed problem. Deblurring works in literature can be broadly classified as (a) conventional approaches and (b) deep learning approaches. Conventional methods formulate deblurring as a prior-based energy minimization problem and deploy iterative optimization schemes to solve the underlying blur and latent image. Earlier works [9, 10] in this category assume a space-invariant blurring model and approximate the motion using convolutions. These methods do not capture the space-varying nature of blur caused due to camera rotation that is a very important aspect of hand-held imaging [11]. More recent works [4, 5, 6] estimate the global camera motion rather than blur kernels at each pixel using formulations such as motion density function [4] and transformation spread function [12].

The other class of works on deblurring is rooted to the current trend of supervised deep learning models. Initial works [13, 14, 15] have shown great success in estimating the blur function thereby aiding in space-invariant deblurring. These models estimate the blur kernel using the network and perform non-blind deblurring outside of the network to produce the clean image. Since these methods estimate a single space-invariant kernel for the entire image, they cannot deal with space-varying blur. Following this, the work in [16] proposes a deep classification network that estimates a parametrized kernel at each patch of a space-variant blurred image. Recently, there are networks [2, 3] that perform end-to-end deblurring and skip the estimation of blur kernels completely. But none of these methods solve for a global camera motion.

Surveying an area and keeping track of the changes helps in timely monitoring and assistance. The task of finding changes between two images become highly challenging when the reference data collected under smooth conditions are clean and those collected during the survey have blurry artifacts. Existing works [7, 8] solve the problem of the blur estimation and change detection from a clean-blurred image pair in a single optimization framework, whereas [17] proposes a deep-network-based change detection scheme given a pair of clean images devoid of any motion artifacts.

Contributions We propose a convolutional neural network that estimates from a single motion blurred image, the global camera motion that causes the underlying space-variant blur. To the best of our knowledge, this is the first attempt to estimate the global camera motion from a space-variant blurred image using a semi-supervised learning scheme. We also show two applications that leverage the estimated camera motion: single image deblurring, and change detection given a reference and blurred image pair, with appropriate comparisons against existing works.

2. BLUR FORMATION MODEL

When a camera moves during the exposure time, different warped versions of the scene are accumulated by the sensor resulting in a blurred image. Hence, a space-variant blurred image can be represented as a weighted sum of warped instances of a clean image (which is the image as captured without any motion) with the weights corresponding to different camera poses along its trajectory [12, 6, 4].

$$B = \sum_{k=1}^{|S|} \omega_k \mathcal{H}_k(L) \tag{1}$$

where B is the observed blurred image, L is the corresponding latent clean frame, the scalar ω_k represents the fraction of time that the camera has spent in the k^{th} camera pose among the $|S|$ camera poses, and $\mathcal{H}_k(\cdot)$ defines the homography operator that warps the input image to the k^{th} pose. The camera pose space S is a 3D space consisting of inplane translations and rotations discretized in a manner such that a displacement of at least one pixel exists between two camera poses. Analogous to the point spread function (PSF), we have $\sum_{k=1}^{|S|} \omega_k = 1$ corresponding to energy conservation. As can be observed, the camera motion and the clean image are associated together in the formation of the blurred image. While it is tempting to get back the clean image from the blurred one, estimating camera motion would possibly serve many applications. Yet we do not choose to ignore the above association in that we employ the latent image corresponding to the blurred image in our semi-supervised training of a neural network that learns to map the blurred image to camera motion.

3. NETWORK ARCHITECTURE AND TRAINING

The goal is to design a neural network that takes a single blurred image as the input and provides the camera motion as the output. The input RGB image is of the form of a 3D tensor, and the output is the vector $\Omega = [\omega_1, \ldots, \omega_{|S|}]^T$ corresponding to the weights of the camera poses.

3.1. Network Architecture

Inspired by Alexnet [18], our network consists of convolutional layers that extract features followed by fully-connected (FC) layers that map them to the camera motion. Each convolutional layer is followed by a non-linear rectified linear unit activation (ReLU) and a batch-normalization unit (except for the first layer). The first FC layer is followed by a Tanh unit, and the output from the last FC layer is subsequently passed through a soft-max layer restricting the final output values to lie in $[0, 1]^{|S|}$ to obtain the camera motion vector $\widehat{\Omega}$, since the pose weights can be viewed as a probability density function indicating the fraction of time that the camera spent in each pose during motion. The complete network architecture with the sizes of feature maps at each layer is shown in Fig. 2. The filter size is fixed as 5×5 in all convolutional layers.

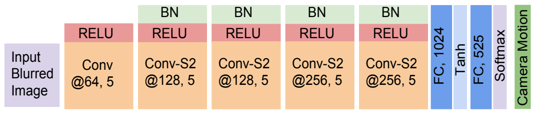


Fig. 2. **Network architecture** Conv@m,n represents convolution with $n \times n$ filter size, and m output feature maps. S2 denotes downsampling by a factor of 2.

We adopt a semi-supervised learning scheme with a supervised motion cost and an unsupervised image cost.

Supervised cost Since the camera poses are ordered at the end of the network, the network is entailed to predict the correct poses and its associated weights. Therefore, we use an L_2-norm loss (mean square error) on the estimated camera motion and the original motion corresponding to the blurred image. In addition, based on the fact that the camera sees only a sparse set of poses from the entire camera pose space, we also impose a sparsity constraint on the estimated camera motion $\widehat{\Omega}$ using an L_1 penalty. Hence, we define the following:

$$E_{mse} = \|\widehat{\Omega} - \Omega_{orig}\|_2^2, \qquad E_{spar} = \|\widehat{\Omega}\|_1. \tag{2}$$

These two loss functions are combined to form the supervised cost:

$$E_{sup} = \lambda_1 E_{mse} + \lambda_2 E_{spar}. \tag{3}$$

Unsupervised cost As noted in Section 2, we propose to exploit the association of the latent image and the camera motion during training. Also, we have access to the clean image along with the camera motion for every blurred image during training. Hence, we define an unsupervised cost based on the image error that facilitates better convergence of the network as follows:

$$E_{unsup} = \lambda_3 \|B - \sum_{k=1}^{|S|} \widehat{\omega}_k \mathcal{H}_k(L)\|_2^2. \tag{4}$$

This cost represents the reblurring error that results when the clean image is blurred using the estimated camera pose weights and compared against the original blurred image B. Writing the above cost in terms of the network output Ω, we have

$$E_{unsup} = \lambda_3 \|b - A\widehat{\Omega}\|_2^2. \tag{5}$$

Here, b is the lexicographically ordered column-vector of the blurred image B, $A_{N \times |S|}$ is a matrix having the k^{th} column as the lexicographically ordered warped clean image L warped by the homography operator $\mathcal{H}_k$, and thus $A\widehat{\Omega}$ represents weighing each column of A with the values from $\widehat{\Omega}$ producing the blurred version of the clean image.

3.2. Training Details

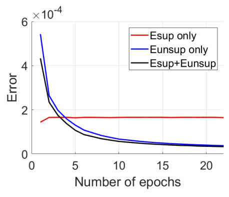
We prepared the training and validation datasets by synthetically blurring the clean images from PASCAL VOC dataset [19]. We first resized the images to 128 × 128 and then generated the blurred images using a 3D camera motion with inplane translations (t_x and t_y) ranging from $[-2 : 2]$ pixels with a step size of one pixel, and inplane rotation $r_z \in [-5 : 5]^\circ$ with a step size of 0.5° that amounts to a pose space containing $|S| = 525$ poses. Note that even this small motion range can create a large amount of space-varying blur for a 128 × 128 image. A total of 200k random connected camera motions are generated, and by picking a random clean image for each of them, we have the training dataset size of 200k. The validation set is also created in a similar manner. We used Torch for training and testing with the following options: ADAM optimizer with momentum values $\beta_1 = 0.9$ and $\beta_2 = 0.99$, learning rate of 10^{-4}, batch-size of 64, $\lambda_1 = 0.01$, $\lambda_2 = 0.1$, $\lambda_3 = 1$, and the total cost as $E_{sup} + E_{unsup}$.

Gradient during backpropagation The gradients for the L_2 and L_1 norms for the supervised cost (3) are employed using the default options in Torch, and the gradient for the unsupervised cost (5) is calculated as follows:

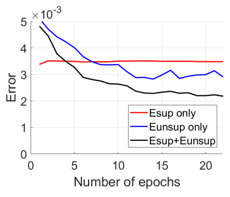
$$E_{unsup} = b^T b - b^T A\Omega - \Omega^T A^T b + \Omega^T A^T A\Omega \quad (6)$$

$$\partial E_{unsup}/\partial\Omega = -2b^T A + 2\Omega^T A^T A \quad (7)$$

For each pair of blurred and clean images, the matrix A is formed on-the-fly using GPU-based warping operations while calculating the gradient in (7) during backpropagation.



(a) Training



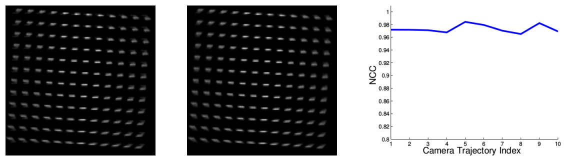
(b) Validation

Fig. 3. Effect of cost functions during training and validation.

Effect of cost functions We checked the effect of the cost functions E_{sup} and E_{unsup} in (3), and (5), respectively, on the convergence of our proposed network during training. Fig. 3 shows the training and validation error curves. Using just the supervised motion cost has convergence issues, but a combined cost converges faster. The camera motion cost alone constrains the solution space to a single value instead of the correct motion. This is due to the softmax and L_1 penalty layers which favors sparse solutions. When the additional unsupervised cost is used, the motion estimated has to not only follow the sparse constraint but should also be able to reproduce the blurred image from the clean input image. This helps to retain sparsity of the motion, and at the same time, produces a motion similar to the ground truth.

4. EXPERIMENTS

Given a blurred image, the camera motion can be obtained as a weight vector corresponding to the camera poses by forwarding the image through the trained network. We provide a quantitative metric for our estimated camera motion over that of the ground truth (GT) using Normalized Cross Correlation (NCC), and we also discuss the applications of deblurring and change detection.



(a) GT motion (b) Estimated motion (c)

Fig. 4. Blur kernels produced at different pixel locations using (a) ground-truth (GT) motion and (b) estimated motion. (c) Normalized cross-correlation (NCC) for ten trajectories from dataset [11].

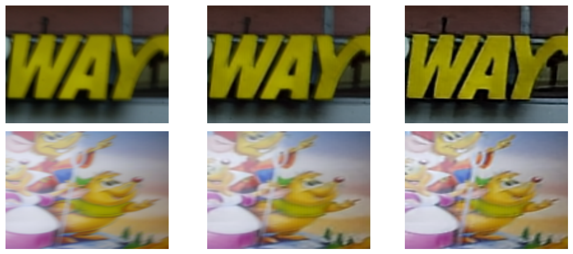
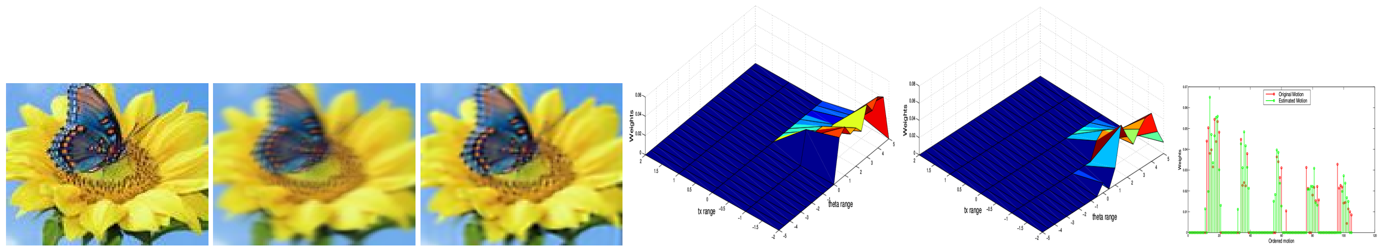

(a) Blurred inputs (b) Outputs of [2] (c) Our outputs

Fig. 5. Deblurring comparisons with the latest deblurring work [2].

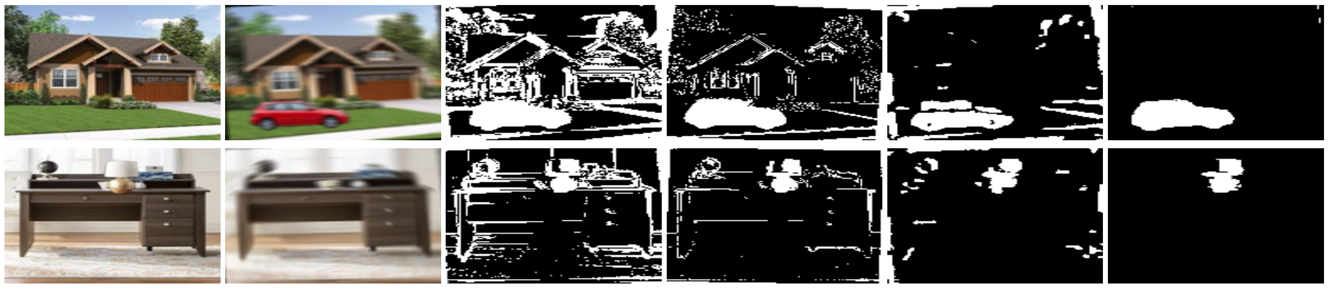
Motion estimation We tested the efficacy of our network to correctly predict the camera motion from a single blurred frame. For this, we took 10 different GT camera motions from the dataset in [11] confined to inplane rotation and translations. These motions were used to blur clean images from the test set of PASCAL VOC dataset. The blurred images were then passed through our network to predict the camera motion. To check the consistency of prediction, we used the NCC metric. The camera motion can be projected to the 2D image plane at different locations to obtain the blur kernel at these locations. We used GT and estimated motion to generate the blur kernels at different pixel locations. The estimated and GT kernels are then matched using NCC and averaged over many locations. An example of blur kernels produced with GT and estimated motion are shown in Figs. 4(a) and (b), respectively. Fig. 4(c) depicts the NCC over the ten different GT and estimated camera motions. All these values are close to one indicating that the estimated motion is able to replicate the blur kernels similar to that produced by GT motion at all pixel positions.

Deblurring Once the camera motion is estimated, a non-blind deblurring is performed to obtain the deblurred image. We use L_1 prior on the image gradient for regularization. The latent image L is obtained by $\min_l\{\|b - \widehat{\mathcal{M}}l\|_2^2 + \lambda|\nabla l|_1\}$ where $\widehat{\mathcal{M}}$ is formed from $\hat{\Omega}$. Let M_k warps the image l using a single homography, then $\widehat{\mathcal{M}}$ is the convex combination of all M_k's resulting in a blur matrix, $\widehat{\mathcal{M}}=\sum_k \widehat{w}_k M_k$. Each row of $\widehat{\mathcal{M}}$ addresses the blur at a corresponding pixel location. We use ADMM-based approach [20] to solve for the latent image. Real examples on image deblurring from the camera motion estimated by our network are provided in Fig.5. Comparison with the latest end-to-end deblurring work [2] is provided in Fig. 5(b). It can be observed that our method works on par with [2]. We also provide a synthetic example in Fig. 6. The reference clean image Fig. 6(a) is blurred with the synthesized camera motion Fig. 6(d) to get the blurred input in Fig. 6(b). This blurred image is forward passed through our network to obtain the camera motion in Fig. 6(e) which is close to the ground truth in Fig. 6(d). Our deblurred result is provided in Fig. 6(c).



(a) Clean image (b) Blurred input (c) Our output (d) Ground-truth motion (e) Estimated motion (f)

Fig. 6. **Deblurring** (a) Clean image; (b) blurred image synthesized using the camera motion in (d); (c) deblurred image using the estimated camera motion (e) using our network; (f) The 2d plot of the ordered camera poses and the corresponding weights for the ground-truth and estimated camera motions in (d) and (e).



(a) I_1 (b) I_2 (c) Direct differencing (d) I_2[2]-I_1 (e) [7] (f) Our output

Fig. 7. **Change Detection** (a) Clean reference; (b) blurred occluded image; (c) direct difference of (a) and (b); (d) result obtained by deblurring (b) using [2] followed by registration and differencing with (a); (e) result obtained from [7]; (f) motion estimated from (b) using our network applied to (a) and then differencing;

Change Detection Given an image pair with one clean (Fig. 7(a)) and another blurred (Fig. 7(b)), our aim is to detect changes between the pair. A simple differencing and thresholding will lead to many unwanted changes to be detected as in Fig. 7(c) due to the blurry artifact in the second image. We deblurred the second image using [2], aligned it with the clean image using SIFT correspondences [21], then subtracted from Fig. 7(a), and finally thresholded to obtain Fig. 7(d). The work in [7] does a joint alignment and change detection, the result of which is provided in Fig. 7(e). Finally, for our output, we used the camera motion estimated from our network and forward blurred the clean reference. The result is then subtracted from Fig. 7(b) and thresholded to obtain the result in Fig. 7(f). From the results, it can be seen that our method detects the changes correctly compared to other methods. Any artifacts or edge smoothening in the deblurring phase or any misalignments in the alignment phase result in spurious changes as in Fig. 7(d). Similarly, the combined registration and alignment of [7] highly depends on the optimization scheme, and is a time consuming process. We also provide quantitative metrics in Table 1. We took 100 clean images and added occlusions at random positions. These were then blurred using global camera motions. Comparisons as in Fig. 7 were conducted and the obtained changes were compared with the ground truth. We report the average accuracy using Percentage of Correct Classification (PCC), Jaccard Coefficient (JC) and Yule Coefficient (YC) measures [22]. We also computed the total execution time (excluding the absolute differencing and thresholding part) for our method and all the comparison methods. The result is provided in Table 2. From Tables 1 and 2, it can be observed that our method is fast, and at the same time, it detects changes comparable to the ground truth.

More details of the quantitative measures, additional results, and our network limitations are provided in the supplementary material.

Table 1. Quantitative comparison for change detection.

Methods	PCC	JC	YC
Ours	**99.31**	**0.6808**	**0.7488**
I_2 [2]-I_1	89.50	0.1198	0.1193
[7]	94.87	0.2613	0.2731

T/F: True/False, P/N: Positive/Negative, Percentage of correct classification PCC = (TP+TN)/(TP+TN+FP+FN), Jaccard coefficient JC = TP/(TP+FP+FN), and Yule coefficient YC = |TP/(TP+FP) + TN/(TN+FN) - 1|. The higher are the values, the better is the method.

Table 2. Computational time in seconds for change detection.

Methods	Ours (GPU+Torch)	[7] (CPU+MATLAB)	I_2 [2]-I_1 (GPU+Torch; MATLAB*)
Steps	Motion Estimation: 0.13 Forward Blurring: 1.68	9.85	Deblurring [2]: 3.02 Alignment*: 0.23
Total	1.81	9.85	3.25

5. CONCLUSIONS

We proposed a deep network to solve for the global camera motion from a single blurred image. The estimated global motion can deal with space-variant blurs caused by camera rotation in addition to translations. We showed applications of global camera motion estimated by our network in single-image deblurring as well as change detection under blur. Quantitative analysis and run time comparisons with other competing methods for change detection favored our proposed method.

6. REFERENCES

[1] Makkena Purnachandra Rao, AN Rajagopalan, and Guna Seetharaman, "Harnessing motion blur to unveil splicing," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 4, pp. 583–595, 2014.

[2] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee, "Deep multi-scale convolutional neural network for dynamic scene deblurring," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[3] TM Nimisha, Akash Kumar Singh, and AN Rajagopalan, "Blur-invariant deep learning for blind-deblurring," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4762–4770.

[4] Ankit Gupta, Neel Joshi, C Lawrence Zitnick, Michael Cohen, and Brian Curless, "Single image deblurring using motion density functions," *Computer Vision–ECCV 2010*, pp. 171–184, 2010.

[5] Li Xu, Shicheng Zheng, and Jiaya Jia, "Unnatural l0 sparse representation for natural image deblurring," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 1107–1114.

[6] Oliver Whyte, Josef Sivic, Andrew Zisserman, and Jean Ponce, "Non-uniform deblurring for shaken images," *International journal of computer vision*, vol. 98, no. 2, pp. 168–186, 2012.

[7] Vijay Rengarajan, Ambasamudram Narayanan Rajagopalan, Rangarajan Aravind, and Guna Seetharaman, "Image registration and change detection under rolling shutter motion blur," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 10, pp. 1959–1972, 2017.

[8] Vijay Rengarajan Angarai Pichaikuppan, Rajagopalan Ambasamudram Narayanan, and Aravind Rangarajan, "Change detection in the presence of motion blur and rolling shutter effect," in *European Conference on Computer Vision*. Springer, 2014, pp. 123–137.

[9] Sunghyun Cho and Seungyong Lee, "Fast motion deblurring," in *ACM Transactions on Graphics (TOG)*. ACM, 2009, vol. 28, p. 145.

[10] Qi Shan, Jiaya Jia, and Aseem Agarwala, "High-quality motion deblurring from a single image," in *Acm transactions on graphics (tog)*. ACM, 2008, vol. 27, p. 73.

[11] Rolf Köhler, Michael Hirsch, Betty Mohler, Bernhard Schölkopf, and Stefan Harmeling, "Recording and playback of camera shake: Benchmarking blind deconvolution with a real-world database," in *European Conference on Computer Vision*. Springer, 2012, pp. 27–40.

[12] Chandramouli Paramanand and Ambasamudram N Rajagopalan, "Non-uniform motion deblurring for bilayer scenes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1115–1122.

[13] Christian J Schuler, Michael Hirsch, Stefan Harmeling, and Bernhard Schölkopf, "Learning to deblur," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 7, pp. 1439–1451, 2016.

[14] Ayan Chakrabarti, "A neural approach to blind motion deblurring," in *Computer Vision – ECCV 2016*, 2016, pp. 221–235.

[15] Li Xu, Jimmy SJ Ren, Ce Liu, and Jiaya Jia, "Deep convolutional neural network for image deconvolution," in *Advances in Neural Information Processing Systems*, 2014, pp. 1790–1798.

[16] Jian Sun, Wenfei Cao, Zongben Xu, and Jean Ponce, "Learning a convolutional neural network for non-uniform motion blur removal," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2015, pp. 769–777.

[17] Ken Sakurada and Takayuki Okatani, "Change detection from a street image pair using cnn features and superpixel segmentation.," in *BMVC*, 2015, pp. 61–1.

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[19] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, pp. 303–338, 2010.

[20] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al., "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[21] David G Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[22] R. J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, "Image change detection algorithms: a systematic survey," *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 294–307, March 2005.