Learning deep neural networks in blind deblurring framework

Junde Wu Xiaoguang Di Jiehao Huang
Yu Zhang
Control & Simulation Center, Harbin Institution of Technology
jundewu.hit@gmail.com, dixiaoguang@hit.edu.cn

Abstract

Recently, end-to-end learning methods based on deep neural network (DNN) have been proven effective for blind deblurring. Without human-made assumptions and numerical algorithms, they are able to restore blurry images with fewer artifacts and better perceptual quality. However, without the theoretical guidance, these methods sometimes generate unreasonable results and often perform worse when the motion is complex. In this paper, for overcoming these drawbacks, we integrate deep convolution neural networks into conventional deblurring framework. Specifically, we build Stacked Estimate Residual Net (SEN) to estimate the motion flow map and Recurrent Prior Generative and Adversarial Net (RP-GAN) to learn an image prior constrained term in half-quadratic splitting algorithm. The generator and discriminators are also designed to be adaptive to the iterative optimization. Comparing with state-of-the-art end-to-end learning based methods, our method restores reasonable details and shows better generalization ability.

1. Introduction

Motion blur is a commonly appeared degradation of image qualities. The blurry images generally caused by the shakes of camera and fast object motions. Most of the deblurring methods are modelled by :

$$O = I_s * K + n \tag{1}$$

where * denotes convolution operator, O, I_s, K, n are observed blurry image, latent sharp image, blur kernel and noise respectively. Solving I_s from Eqn. (1) is highly ill-posed due to the unknown blur kernel and extra noise.

The end-to-end learning based methods [19, 17, 29, 9] solved it by training a neural network to restore the clear images from blurred observations directly. Depending on the strong fitting ability of deep neural network (DNN), they restored images with fewer artifacts and better visual effect.

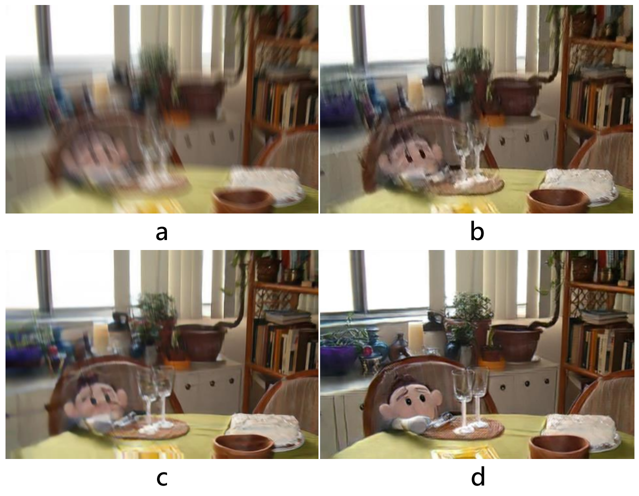

Figure 1: A deblurred example. (a) Blurry image. (b) Result of Nah *et al*. [17]. (c) Result of Tao *et al*. [30]. (d) Ours.

However, in practice, we also find some limitations of this kind of methods. Firstly, most of them perform worse when the motion is too large or the blur is highly space-variant. Besides, sometimes they restore heavy blurry content to unreasonable objects since lacking theoretical guidance.

In this work, we take advantage of both generalization capability of conventional deblurring framework and strong fitting ability of neural network to build a more effective blind deblurring method. Specifically, we design a generative and adversarial net (GAN) to generate the optimal solution of a sub-problem constrained by image prior in half-quadratic splitting algorithm. By doing that, we avoid learning an explicit prior and corresponding numerical optimization process. As the network is supervised by sharp images directly, the corrupted deconvolutional results can be constrained by nature image priors rather than assumption-based ones.

For adapting to iterative optimization, we build the generator with recurrent structure which enables it to weight different prior constraint depending on the input data. Unlike the commonly used GAN structure, we use two discriminators to constrain the recurrent generator. One is conditional Generative and Adversarial Net (cGAN) discriminator [16], which plays generative and adversarial game with the generator. The other one is artifacts penalized discriminator, which is built to impose extra punishment on artifacts. Two discriminators work together for promising that the image quality can be improved through iterations. We call the network as Recurrent Prior Generative and Adversarial Net (RP-GAN). For blind deblurring , we build a stacked residual network, called Stacked Estimate Net (SEN), to estimate the space-varying motion flow at each pixel of the blurry image. In that case, RP-GAN can be trained to be robust to the mistakenly estimated motion blur kernels. In the end, we send the result to the whole system again to further process the residue blur. The global iteration approximately restores the nonlinear blur by eliminating the linear blur content iteratively.

In conclusion, the main contributions of this paper are:

- We build SEN to estimate the nonuniform motion flow map. Its accuracy exceed all the other deep learning based methods.
- We propose RP-GAN to adaptively weight prior constraint on corrupted deconvolutional results.
- We build global iterations for extending our algorithm to nonlinear blur.

2. Related Work

Most conventional approaches get success depending on guiding maximum a posteriori probability (MAP) process by assumed priors, such as the total variational regularizer [31, 21], Gaussian scale mixture priors [3] , normalized sparsity [8], L0 gradients [32] , dark channel prior [20], *etc*. However, since all these hand-crafted priors are designed under limited observations or restricted assumptions, these algorithms show unsatisfied results when processing the images that have fewer corresponding features. Hand-crafted priors are also proved effective for blur kernel estimation, such as edge-extraction based MAP [2, 27], gradient activation based MAP [4] and variational Bayesian methods [11, 12, 35], etc. However, their generalization capabilities are also limited by human-made assumptions .

The development of neural networks inspires many works to utilize it in conventional deblurring framework. Schuler *et al.* [22] propose a neural network to estimate the blur kernel and restore the images by deconvolution. Yan and Shao [34] build a classification network to classify the blur type and then a regression network to estimate its parameters. Sun *et al.* [26] propose a convolutional neural network (CNN) to predict the probabilistic distribution of blur. Gong *et al.* [5] propose a fully-convolutional deep neural network (FCN) to directly estimate the blur kernel in pixel level and get a higher accuracy. On the other hand, several works use neural network to learn generic image priors (explicit or implicit) for deblurring [24, 37, 23, 36]. Zhang *et al.* [36] learn denoised gradient as image prior to guide the deconvolution model. Li *et al.* [13] learn a discriminative prior for deblurring in generic scenarios. However, they have to compromise on the restoration quality when embedding an explicit prior to numerical methods. Schuler *et al.* [23] and Zhang *et al.* [37] train models with implicit priors for restoring the corrupted deconvolutional results by multi-layer perceptron (MLP) and multiple CNN respectively. Nevertheless, since trained for non-blind deconvolution, these methods are hard to be applied to blind deblurring directly. Besides, restricted by development of neural networks, their network structures are actually too simple to fit their tasks.

Recently, end-to-end learning based methods have shown great advantages for deblurring. Nah *et al.* [17] propose a multi-scale CNN for restoring the images in three different levels. Each level is responsible for processing different blur scale, from small to large. Kupyn *et al.* [9] build GAN model for deblurring directly from the blurred observations to the sharp images. Tao *et al.* [29] inherited the multi-scale structure from Nah *et al.* [17] and add long short-term memory (LSTM) in the recurrent process. End-to-end learning based methods restore the images with fewer artifacts than optimization based methods. But they highly depend on the observed data. Thus, on the one hand, they often show worse generalization ability, especially when the motion is large or complex. On the other hand, since these methods completely discard the deblurring principle, they sometimes restore unreasonable results from blur images, which may cause the restored images to be more confusing than the blurry ones.

3. Motion Flow Estimation by SEN

We propose to estimate spatially-varying motion flow map using a stacked residual network from end to end. Our motion flow map is modeled following [5]. Specifically, given an arbitrary RGB blurry image O which has the size $H * W$, the task is generating a $H * W * 2$ matrix U to represent motion flow map. U can be expressed as : $U(i, j, 1) = u_{i,j}$, $U(i, j, 2) = v_{i,j}$, $\forall i, j \in O$. u and v denote horizontal and vertical motions respectively.

The network we propose, called SEN, is mainly based on an encoder-decoder structure which has been proven useful in various vision tasks [15, 25, 28, 33, 18]. We adapt the network structure from [18]. Its detailed structure is shown in Figure 2. Unlike the symmetrical structure used in [18], we use three dilated convolution layers with increasing channels in front of the network. They help to enlarge the recep-

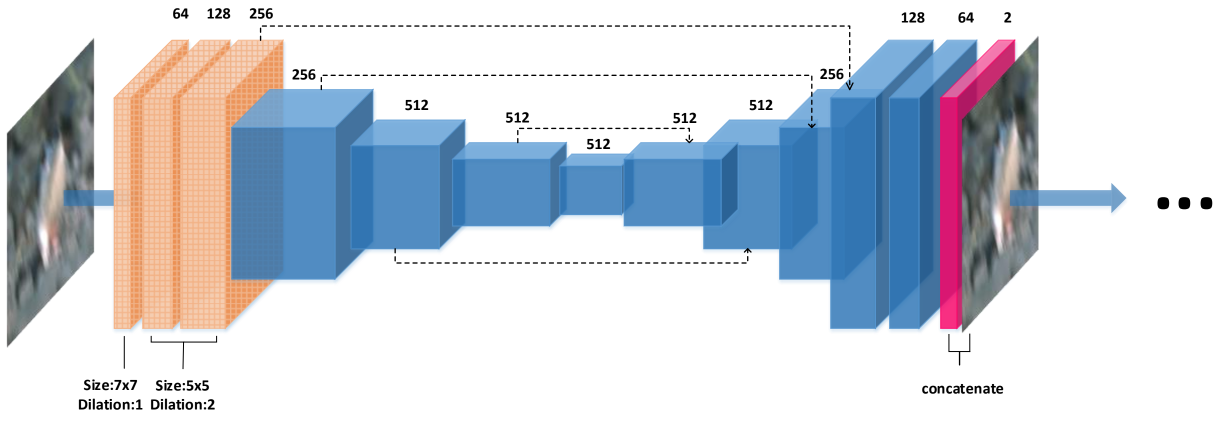

Figure 2: The first stack of SEN. Later stacks share the similar structure.

🟧 is dilated convolution. 🟦 is residual learning block, 🟥 is objective motion flow map, ⇢ is skip connection.

tive field and abstract the intact deep blur features .

Because the prior layers are more likely related to the image content but not the blur features. Skip connections are added only after dilated convolution layers. From experiments, we find the skip connections on prior layers impair the network performance because of the large difference between the image content and the motion flow map.

The stacked structure is utilized to improve the estimation accuracy. Stacked structure takes the result from the last network to serve as the input of the next one. In the paper, we build the stacked networks in the similar structure with the first one. The result from the first network concatenated with the blur image serves as the next network's input. Although more stacks can improve the estimation accuracy, they will also cause overfitting. For the balance, we take two stacks in practice.

In previous works [26, 5], the task was treated as a classification problem. However, we find that regression loss help network reach higher accuracy. Specifically, L2 loss is used for training, which can be expressed as :

$$\mathcal{L}_{L_2} = \frac{1}{N}\|U^* - U^l\|_2^2 \tag{2}$$

where U^* denotes the estimated matrix and U^l denotes the label. N is the number of all elements in the matrix.

4. RP-GAN embedded in HQ Algorithm

4.1. Half-Quadratic Splitting Algorithm(HQ Algorithm)

When knowing the motion flow maps in pixel level, the deblurring issue can be modeled as:

$$I = \arg min_I \|I * K - O\|_2^2 + \gamma p(I) \tag{3}$$

where I denotes the latent sharp image. O denotes the given blurry image. K denotes the heterogeneous motion blur kernel map with different blur kernels for each pixel in O. $p(I)$ denotes the latent prior of the image. $*$ implies general convolution operation. γ is a weighting constant. Here each nonuniform blur kernel is applied on each different pixel. The specific operation is done following [5]. In Eqn. (3), since both I and p are unknown, it's hard to solve the equation directly.

Half-quadratic splitting algorithm is a way to simplify it. By HQS, for solving Eqn. (3), we can split it into two sub-problems in Eqn. (4).

$$\begin{cases} I_{n+1}^* = \arg min_{I_{n+1}^*} \frac{\beta}{2}\|I_{n+1}^* - Z_n\|_2^2 \\ \qquad + \frac{1}{2}\|I_{n+1}^* * K - O\|_2^2 & \textcircled{1} \\ Z_{n+1} = \arg min_{Z_{n+1}} \frac{\beta}{2}\|I_{n+1}^* - Z_{n+1}\|_2^2 \\ \qquad + \gamma p(Z_{n+1}) & \textcircled{2} \end{cases} \tag{4}$$

where n is the number of iterations. I^* is the corrupted deconvolutional image, Z is an auxiliary variable initialized with observed blurry image O. β is a variable parameter. By

iteratively optimizing two sub-problems with increasing β, the solved I could be seemed as an approximate solution of Eqn. (3).
Eqn. (4)-① has the analytic solution :

$$I^*_{n+1} = [K^T K + \beta \mathcal{I}]^{-1} [\beta Z_n + K^T O] \tag{5}$$

$\mathcal{I}$ is the identity matrix.

But Eqn. (4)-②, because of the unknown image prior, can not be solved without any assumption. Therefore, training a neural network to solve Eqn. (4)-② is an appealing idea. In this way, we can learn the implicit image prior directly through data. However, Eqn. (4)-② constraints images by image prior in different levels, which controlled by parameter β. Thus we build the network with recurrent structure to weight adaptive prior during the iteration. Detailed RP-GAN are given in the next subsection.

4.2. Recurrent Prior GAN and Insights behind it

We build Recurrent Prior GAN (RP-GAN) to learn Eqn. (4)-② from end to end. The structure of RP-GAN is roughly based on [6]. Detailed structure is provided in Appendix A.

4.2.1 Recurrent Structure

When using a single neural network to learn Eqn. (4)-②, it's hard to integrate the parameter β into the network. However, the observation that network outputs clearer images when accepting better-quality deconvolutional results makes us believe network could learn to adjust the latent parameter β by itself (through perceiving variant inputs). From the above, we build the recurrent structure to adapt to the conventional iterative process. In recurrent process, the restored result of the n recurrence (level) is served as variable Z_n for Eqn. (4)-①, then it will be used to calculate the corrupted deconvolutional value I^*_{n+1} through Eqn. (5). I^*_{n+1} will then be the input of $n+1$ level in recurrent generator. The process is shown in Figure 3. If using a general GAN structure (with one generator and one adversarial discriminator) for the task, once GAN outputs the images with remained artifacts, these artifacts will then escalate in the deconvolution process and lead the inputs of the next level become even worse. The artifacts' corruption will be gradually aggravated in the process of recurrence. A way to prevent it is designing the network which favors blur more than artifacts. The advantages are twofold. On the one hand, comparing with an artifacts-corrupted intermediate result (Z), a blurry intermediate result will not cause extra artifacts in the next deconvolutional input (I), thus promises convergence. On the other hand, the residue blur can be further processed in the global iteration, but the artifacts can not. Therefore, RP-GAN could output images from blurry to clear in its recurrence. We show an example in Figure 4. The way we implement it is putting extra penalties on the artifacts. The details are shown in section 4.2.3.

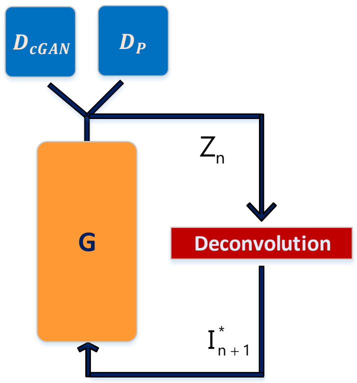

Figure 3: The recurrent process of RP-GAN. G denotes generator, D_{cGAN} and D_p are cGAN discriminator and artifacts-penalize discriminator respectively

4.2.2 Generator

The inputs of the generator is the deconvolutional image: I^* and the concatenated observed blurry image: O which is used to compensate the heavily corrupted information in I^*. We also skip connect the concatenated inputs with the last layer of the network to make the network to be more dependent on the offered information, thus to be sensitive to different inputs in different iterations.

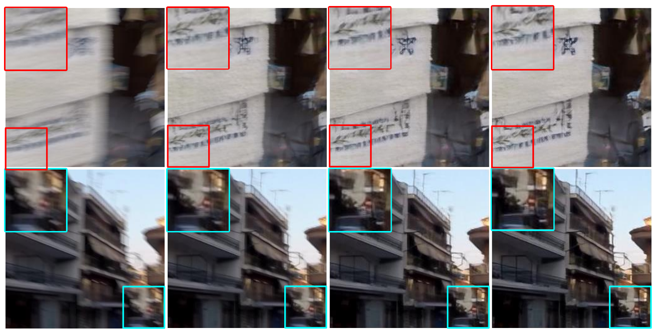

Figure 4: The images from left to right are blurry image, result of level 1, result of level 2, result of level 3 respectively. We see the RP-GAN outputs image from blurry to clear with recurrence going on.

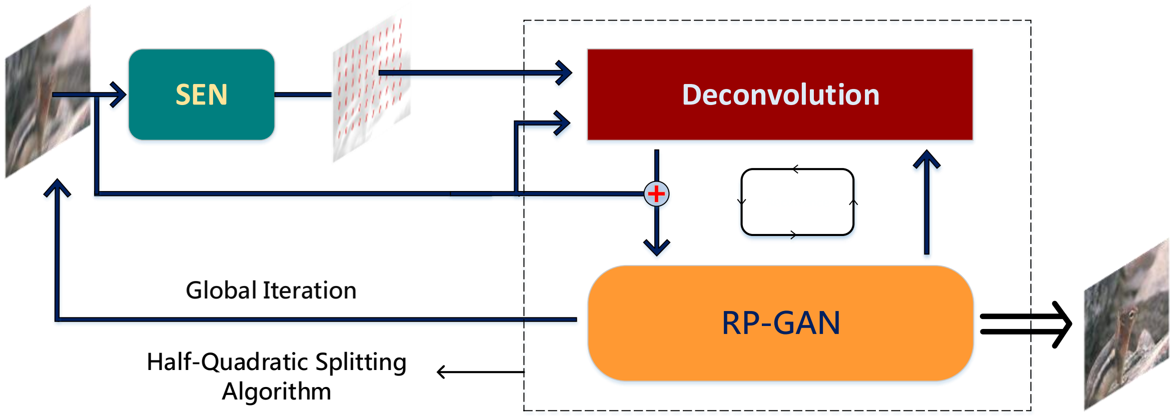

Figure 5: The deblurring system
⊕ denotes concatenate. $\Longrightarrow$ denotes system output. The blurry image is firstly sent to SEN for estimating the motion flow map. Then the blurry image can be deconvolved by the estimated motion flow map and then send to RP-GAN's recurrent process. The result of RP-GAN will serve as the blurry image and be sent to the system again in global iteration.

4.2.3 Discriminator

Two discriminators are built to ensure RP-GAN would rather contain blurry content than contain artifacts during recurrence. One discriminator plays a minimax game with G, to penalize the difference between generated images and sharp images. Another discriminator gives extra penalty to the artifacts. Two discriminators help G generate minimum artifacts while restoring clear content.

cGAN discriminator. Since we have used blurry image as additional information, we build our generative and adversarial strategy following conditional generative and adversarial net [16]. The objective of G and D in it can be expressed as:

$$\begin{aligned}\arg\min_{G}\max_{D_{cGAN}} \quad & \mathbb{E}_{I_s,O}[log(D_{cGAN}\,(I_s|O))]+ \\ & \mathbb{E}_{I^*,O}[log(1-D_{cGAN}\,(G(I^*|O)|O)]\end{aligned} \tag{6}$$

where I_s denotes the latent sharp image. O denotes the observed blurry image. I^* denotes the corrupted deconvolutional image. Generator G learns to minimize the objective against discriminator D_{cGAN}, which tries to maximize it. The generator is expected to generate images closer to natural ones through playing the minimax game continuously. The generated image will be sent to the discriminator with concatenated conditional variable. By comparing the blurry image with the generated image, cGAN discriminator can better distinguish the generated one from the sharp one.

Artifacts-penalized discriminator. For constraining artifacts, we build an extra penalty term. Artifacts-penalized discriminator penalizes G if generated image is closer to the corrupted input than the sharp image. We adapt WGAN [1] to our discriminator. WGAN discriminator is trained to approximate the Wasserstein distance, or Earth-Mover distance, between the generated distribution and the real distribution. But here, we train the artifacts-penalized discriminator as a Wasserstein distance between corrupted inputs and observed blurry images. The objective of artifacts-penalized discriminator can be expressed as :

$$W(P_{data},P_o) \approx \max_{||D_p||\leq 1} \mathbb{E}_{x\sim P_{data}}[D_p(x)] - \mathbb{E}_{y\sim P_o}[D_p(y)] \tag{7}$$

where P_{data} is the distribution of the corrupted input data and P_o is the distribution of the observed blurry image. x and y are the samples from distribution P_{data} and distribution P_o respectively. $W(P_{data}, P_o)$ denotes the Wasserstein between two distributions. D_p denotes the artifacts-penalized discriminator. $||D_p|| \leq 1$ denotes D_p should be 1-Lipschitz function. The penalty to G is expressed as:

$$\mathcal{L}_p(G, D_p) = (D_p(G(I^*|O)) - D_p(I_s))_+ \tag{8}$$

$()_+$ denotes the positive part of the content. As shown in Eqn. (7), artifacts-penalized discriminator is trained to output a larger value for artifacts-corrupted image and a smaller value for blurry image. Thus, as shown in Eqn. (8), punishment is only imposed when $D_p(G(I^*|O))$ is larger than $D_p(I_s)$, which denotes the generated image is closer to the corrupted input than the sharp image in learned measure.

4.2.4 Loss function

Total loss of RP-GAN is expressed as:

$$\begin{aligned}\mathcal{L}_{RP-GAN} =& \mathcal{L}_c\,(G) + \gamma\,\mathcal{L}_{cGAN}(G, D_{cGAN}\,)+ \\ & \lambda\mathcal{L}_P(G, D_p)\end{aligned}$$

where $\mathcal{L}_c(G)$ denotes the content loss. $\mathcal{L}_{cGAN}$ denotes the adversarial loss. $\mathcal{L}_P$ denotes extra penalty. λ and γ are weighting constants. The objective of RP-GAN is simultaneously minimizing the three items until convergence. Specifically, adversarial loss can be expressed as:

$$\mathcal{L}_{cGAN}(G, D_{cGAN}) = log(1 - D_{cGAN}\ (G(I^*|O)|O))$$

We take perceptual loss[7] as our content loss, which can be expressed as:

$$\mathcal{L}_c(G) = \frac{1}{C * H * W} \sum_{c=1,i=1,j=1}^{C,H,W} [\phi(G(I^*, O)) - \phi(I_s)]$$

where ϕ is a pretrained network for abstracting the depth value of images. C, H, W are the channel, height and weight of a hidden layer of network ϕ. Perceptual loss compares the depth value of images instead of directly penalizing the difference of pixel-level information. In our experiment, it helps to prevent the network from overfitting to the image contents and generate images with better perceptual quality. In practice, we compute the loss at layer relu2_2 of the VGG16 network. Extra penalty $\mathcal{L}_P$ is shown in Eqn. (8).

5. Nonuniform and Nonlinear Blind Deblurring

Note SEN could only produce nonuniform but linear motion flow maps. But in practice, the blur is more likely to be nonuniform and nonlinear. Since a nonliner blur kernel can be approximated by multiple linear blur kernels in different orientations, for processing the nonlinear blur images, we use the successive approximation strategy. Final results of RP-GAN will be sent to the system again for further processing the residue blur content. Through eliminating the approximated liner blur iteratively, the nonlinear blur can be gradually eliminated. As we estimate the blur kernels in pixel level, the global iterations have high efficiency. The effect of it is shown in Table 2 and Table 3. The intact system for blind deblurring is shown in Figure 5.

6. Experiments

Our networks are implemented using PyTorch deep learning framework. All models are ran on NVIDIA 1080Ti GPU.

6.1. Motion Blur Kernel Estimation

Datasets Our data is generated following [5]. For adapting to different scales of blur kernels, we generate two datasets.

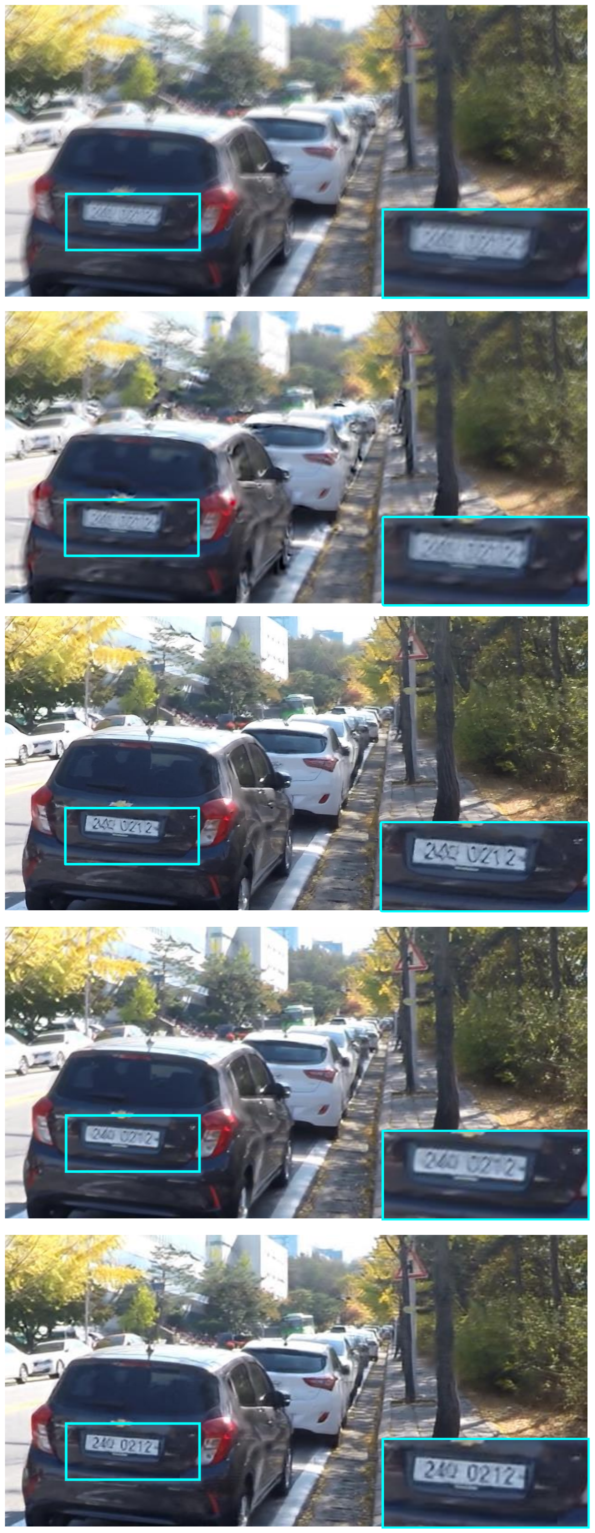

Figure 6: Test results on GOPRO dataset. Images from top to bottom are blurry image, results of Sun *et al.* [26], Nah *et al.* [17], Tao *et al.* [30] and ours respectively.

Table 1: MSE on motion flow estimation

Dataset	Sun *et al.*	Gong *et al.*	Ours
MSCOCO-23	24.34	5.54	**2.71**
MSCOCO-46	63.37	7.21	**4.24**

One is for estimating small-scale blur kernels, the ceiling of it is set as $v_{max} = u_{max} = 23$. The other is for large-scale ones, which is set as $v_{max} = u_{max} = 46$. Both datasets contain 12000 blur images which generated from 1200 sharp images from Microsoft COCO [14] (Each sharp image generates 20 blur images). The 10000 images of the datasets are used for training. The other 2000 are for testing. Two datasets are referred as MSCOCO-46 and MSCOCO-23 respectively.

Training details SEN is trained by images cropped to 256*256 pixels. We set two stacks in practice. Both two stacks are trained by batch size 16. In the first stack, learning rate is set as $1.25\,e^{-3}$ in the first 50 epochs, and linearly decreases to $1.25\,e^{-5}$ in the next 40 epochs. The second stack is trained 15 epochs by learning rate $1.25\,e^{-5}$. Both of them use Adam solver with $\beta_1 = 0.9$, $\beta_2 = 0.99$ and $\epsilon = 10^{-8}$.

Comparison We compare our method with Sun *et al.* [26] and Gong *et al.* [5]. Sun *et al.* estimated blur kernel in patch by CNN first, then further process them with markov random field for smoothness. Gong *et al.* proposed a FCN to estimate motion flow maps in pixel level and get previous state-of-the-art results. In our experiments, SEN shows higher qualitative results than previous network by comparing mean square error (MSE). The comparison is shown in Table 1.

6.2. Blind Deblurring

For evaluating our blind deblurring method, we compare our method on mainstream benchmarks with previous state-of-the-art image deblurring approaches. PSNR and SSIM metrics are used to evaluate the restored image quality.

RP-GAN training details RP-GAN is trained on two different datasets to ensure model's generalization. One contains 1000 synthetic blurred images generated following section 6.1. The other is GOPRO dataset [17], which provides 2103 blur/clear pairs for training. All deconvolutional results are solved by the estimated blur kernels from pre-trained SEN model. Batch size is set as 16. Learning rate is set as $1e^{-4}$ for both generator and discriminator in the first 110 epochs, and linearly decreases to $1e^{-6}$ in the next 80 epochs. We use the same solver as which used for SEN. For speeding up convergence, we begin the recurrence after the first 40 epochs, then update deconvolutional results every 3 epochs. The whole training process costs about 15 days.

Comparisons on benchmarks We compare our algorithm with [26] [17] and [30]. Sun *et al.* [26] learned blur kernel through CNN, then did non-blind deblurring by traditional deconvolution method. Nah *et al.* [17] and Tao *et al.* [30] are two representative end-to-end learning methods for deblurring. Both of them used multi-scale strategy and generate perceptually convincing images. Tao *et al.* got previous state-of-the-art results on mainstream benchmarks.

Table 2: Quantitative results on GOPRO testing dataset

Method	Sun *et al.*	Nah *et al.*	Tao *et al.*
PSNR	24.64	29.08	30.10
SSIM	0.8429	0.9135	0.9323
Time	20min	3.09s	**1.6s**
Method	Ours.it1	Ours.it2	**Ours.it3**
PSNR	29.86	30.35	**30.75**
SSIM	0.9314	0.9386	**0.9402**
Time	17s	32s	47s

Table 3: Quantitative results on Kohler dataset

Method	Sun *et al.*	Nah *et al.*	Tao *et al.*
PSNR	25.22	26.48	26.80
SSIM	0.7735	0.8079	0.8375
Method	Ours.it1	Ours.it2	**Ours.it3**
PSNR	27.25	28. 46	**29.19**
SSIM	0.8642	0. 8933	**0.9128**

Methods are compared on two mainstream benchmarks, GOPRO dataset [17] and Kohler dataset [10]. GOPRO dataset generates long-exposure blurry frames by averaging consecutive short-exposure frames from videos captured by high-speed cameras. The dataset provides 2103 clear/blur pairs for training and 1111 pairs for testing. The quantitative results on GOPRO testing set are listed in Table 2. Visual comparison is shown in Figure 6. More comparisons are provided in Appendix B. In Table 2 and Table 3, it1,2,3 denote the results of global iteration 1, 2 and 3 respectively. We see the results are improved with the iteration going on, and get the best result in the third iteration. Because containing the deconvolution process, our method will certainly cost more time than end-to-end learning methods. But using the deep neural network for optimization helps us take much less time than Sun *et al.*, who used a numerical method.

Kohler dataset [10] recorded and analyzed real camera motion, which is played back on a robot platform. The dataset consists of 4 images blurred with 12 different kernels for each of them. Comparisons on Kohler dataset are listed in Table 3. Visual comparison is shown in Figure 7. Comparing with end-to-end learning methods, our method restores clearer details on GOPRO test dataset. On Kohler dataset, our method outperforms others by a large margin. That is because the restoration ability of end-to-end learning based methods is only dependent on training dataset. It results in their failure of restoring extreme motions. This fault seems not so apparent on GOPRO testing dataset. But

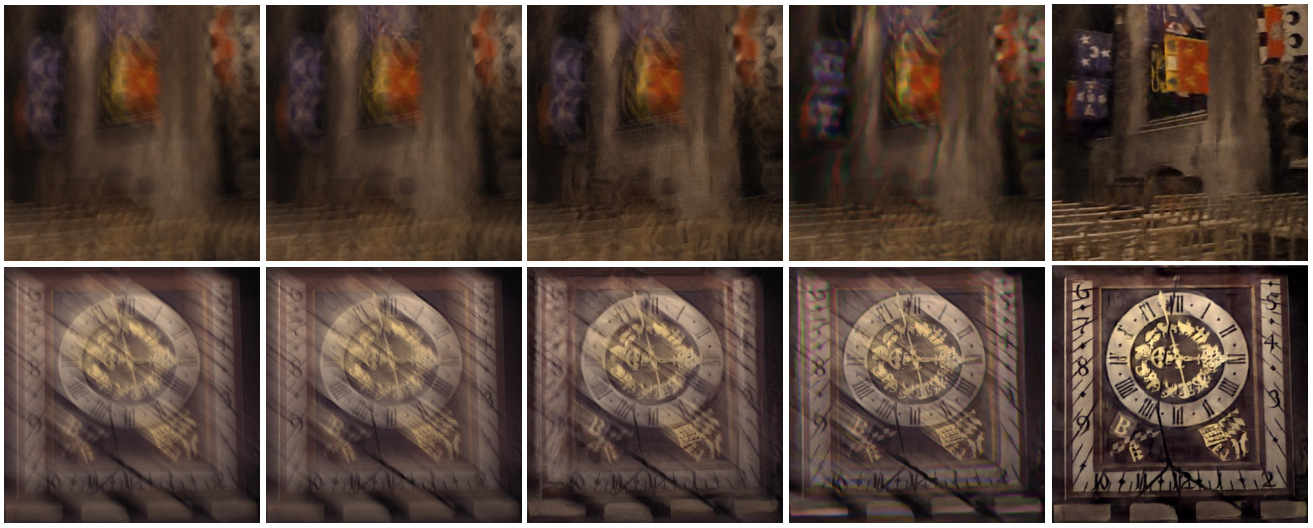

Figure 7: Test results on Kohler dataset. Images from left to right are blurry image, results of Sun *et al.* [26], Nah *et al.* [17], Tao *et al.* [30] and ours respectively.

Table 4: Comparison of different recurrent levels

Level	1	2	**3**	4	5
PSNR	28.77	29.58	**29.86**	29.83	29.84
SSIM	0.9273	0.9305	**0.9314**	0.9312	0.9312

sometimes it restores unreasonable objects when the blur content is heavy. Such as Nah *et al.* restore the slant license plate number in Figure 6. Unlike the end-to-end learning methods, we build two networks to independently estimate the motion flow and restore the deconvolutional results in the optimization framework. Because of learning a easier subtask, SEN can have a better awareness of motion than the end-to-end deblurring networks. Then RP-GAN can utilize the known motion to restore more reasonable objects. A typical example is shown in Figure 8. On the other hand, end-to-end learning based methods show worse effect on challenge blur images in Kohler dataset, which the motion is larger or more complex than which is in GOPRO dataset. Since the deblurring framework we use is universally applicable, our method shows stronger generalization capability.

Recurrent strategy As how many recurrent levels can also be a factor influencing the restored image quality. We compared the effect of setting different levels. The quantitative results of each setting are listed in Table 4. We see the results improve a lot in the first three levels, and converges later. For the best efficiency, we take three levels in the paper.

The effectiveness of artifacts-penalize discriminator RP-GAN is trained without artifacts-penalized discriminator at first. The corrupted inputs cause the results show much noise and more artifacts after recurrence. Visual comparison is shown Appendix B.

Comparison on real blurred images The comparisons with other methods on real-captured images are provided in Appendix B.

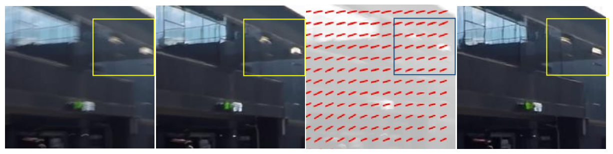

Figure 8: A typical example of the end-to-end learning methods' mistaken deblurring. Images from left to right is blurry image, result of Tao *et al.* [30], our estimated motion flow and our deblurred image respectively. We see Tao *et al.* restore the vent to a weird shape. But since we have estimated the right motion in that location, we can restore it to a reasonable shape.

7. Conclusion

In this paper, we embed the deep neural network in a conventional deblurring framework for blind deblurring. Under theoretical guidance, our method avoids the existing drawbacks of end-to-end learning methods. The restored images are more reasonable and able to restore the large and complex motion. Comparing with other methods on mainstream benchmarks, our approach outperforms the state-of-the art methods, both qualitatively and quantitatively.

References

[1] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. 2017.

[2] S. Cho and S. Lee. Fast motion deblurring. In *Acm Siggraph Asia*, pages 1–8, 2009.

[3] 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.

[4] D. Gong, M. Tan, Y. Zhang, A. V. D. Hengel, and Q. Shi. Blind image deconvolution by automatic gradient activation. In *Computer Vision and Pattern Recognition*, pages 1827–1836, 2016.

[5] D. Gong, J. Yang, L. Liu, Y. Zhang, I. Reid, C. Shen, A. V. D. Hengel, and Q. Shi. From motion blur to motion flow: A deep learning solution for removing heterogeneous motion blur. 2017.

[6] P. Isola, J. Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. pages 5967–5976, 2016.

[7] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 2016.

[8] D. Krishnan, T. Tay, and R. Fergus. Blind deconvolution using a normalized sparsity measure. In *Computer Vision and Pattern Recognition*, pages 233–240, 2011.

[9] O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. 2017.

[10] R. Khler, M. Hirsch, B. Mohler, B. Schlkopf, and S. Harmeling. *Recording and Playback of Camera Shake: Benchmarking Blind Deconvolution with a Real-World Database*. Springer Berlin Heidelberg, 2012.

[11] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. 8(1):1964–1971, 2009.

[12] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Efficient marginal likelihood optimization in blind deconvolution. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2657–2664, 2011.

[13] L. Li, J. Pan, W. S. Lai, C. Gao, N. Sang, and M. H. Yang. Learning a discriminative prior for blind image deblurring. 2018.

[14] T.-Y. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollr, and C. L. Zitnick. Microsoft coco: Common objects in context. *CoRR*, abs/1405.0312, 2014.

[15] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala. Video frame synthesis using deep voxel flow. pages 4473–4481, 2017.

[16] M. Mirza and S. Osindero. Conditional generative adversarial nets. *Computer Science*, pages 2672–2680, 2014.

[17] S. Nah, T. H. Kim, and K. M. Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. pages 257–265, 2016.

[18] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. pages 483–499, 2016.

[19] T. M. Nimisha, A. K. Singh, and A. N. Rajagopalan. Blur-invariant deep learning for blind-deblurring. In *IEEE International Conference on Computer Vision*, pages 4762–4770, 2017.

[20] J. Pan, D. Sun, H. Pfister, and M. H. Yang. Blind image deblurring using dark channel prior. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1628–1636, 2016.

[21] D. Perrone and P. Favaro. Total variation blind deconvolution: The devil is in the details. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2909–2916, 2014.

[22] C. Schuler, M. Hirsch, S. Harmeling, and B. Scholkopf. Learning to deblur. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 38(7):1439–1451, 2016.

[23] C. J. Schuler, H. C. Burger, S. Harmeling, and B. Scholkopf. A machine learning approach for non-blind image deconvolution. pages 1067–1074, 2013.

[24] S. Sreehari, S. V. Venkatakrishnan, B. Wohlberg, G. T. Buzzard, L. F. Drummy, J. P. Simmons, and C. A. Bouman. Plug-and-play priors for bright field electron tomography and sparse interpolation. *IEEE Transactions on Computational Imaging*, 2(4):408–423, 2016.

[25] S. Su, M. Delbracio, J. Wang, G. Sapiro, W. Heidrich, and O. Wang. Deep video deblurring. 2016.

[26] J. Sun, W. Cao, Z. Xu, and J. Ponce. Learning a convolutional neural network for non-uniform motion blur removal. (CVPR):769–777, 2015.

[27] L. Sun, S. Cho, J. Wang, and J. Hays. Edge-based blur kernel estimation using patch priors. In *IEEE International Conference on Computational Photography*, pages 1–8, 2013.

[28] X. Tao, H. Gao, R. Liao, J. Wang, and J. Jia. Detail-revealing deep video super-resolution. pages 4482–4490, 2017.

[29] X. Tao, H. Gao, X. Shen, J. Wang, and J. Jia. Scale-recurrent network for deep image deblurring. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[30] X. Tao, H. Gao, Y. Wang, X. Shen, J. Wang, and J. Jia. Scale-recurrent network for deep image deblurring. 2018.

[31] C. TF and W. CK. Total variation blind deconvolution. *IEEE Transactions on Image Processing*, 12(3):370–375, 1998.

[32] L. Xu, S. Zheng, and J. Jia. Unnatural l0 sparse representation for natural image deblurring. pages 1107–1114, 2013.

[33] N. Xu, B. Price, S. Cohen, and T. Huang. Deep image matting. 2017.

[34] R. Yan and L. Shao. Blind image blur estimation via deep learning. *IEEE Transactions on Image Processing*, 25(4):1910–1921, 2016.

[35] H. Zhang, D. Wipf, and Y. Zhang. Multi-image blind deblurring using a coupled adaptive sparse prior. In *Computer Vision and Pattern Recognition*, pages 1051–1058, 2013.

[36] J. Zhang, J. Pan, W. S. Lai, R. Lau, and M. H. Yang. Learning fully convolutional networks for iterative non-blind deconvolution. pages 6969–6977, 2016.

[37] K. Zhang, W. Zuo, S. Gu, and L. Zhang. Learning deep cnn denoiser prior for image restoration. pages 2808–2817, 2017.

A. RP-GAN structure

We use **CBR** $k * k * c : s$ denote the convolution layer with batch normalization and relu activation function, the convolution layer has kernel size $k * k$ and c output channels applied with stride s , use **Res1** c denote the first kind of Residual Block which has c output channels, **Res2** c denote the second kind of Residual Block which has c output channels, then has:

Res1 $c : input \rightarrow CBR\ 3 * 3 * c : 1 \rightarrow CBR\ 3 * 3 * c : 1 + input \rightarrow output$

Res2 $c : input \rightarrow CBR\ 3 * 3 * c/2 : 1 \rightarrow CBR\ 3 * 3 * c/2 : 1 \rightarrow CBR\ 3 * 3 * c : 1 + input \ ⋇ \ CBR\ 3 * 3 * c : 1 \rightarrow output$

RP-GAN

Generator : $input \rightarrow CBR\ 7 * 7 * 64 : 1 \rightarrow CBR\ 3 * 3 * 128 : 2 \rightarrow CBR\ 3 * 3 * 256 : 2 \rightarrow Res1\ 256 \rightarrow Res1\ 256 \rightarrow Res1\ 256 \rightarrow Res1\ 256 \rightarrow Res1\ 256 \rightarrow Res1\ 256 \rightarrow Res1\ 256 \rightarrow CBR\ 3 * 3 * 256 : 2 \rightarrow CBR\ 3 * 3 * 128 : 2 \rightarrow CBR\ 7 * 7 * 64 : 1 + input ⋇ Res2\ 3 \rightarrow output$

DcGAN : $input \rightarrow CBR\ 4 * 4 * 64 : 2 \rightarrow CBR\ 4 * 4 * 128 : 2 \rightarrow CBR\ 4 * 4 * 256 : 2 \rightarrow CBR\ 4 * 4 * 512 : 2 \rightarrow CBR\ 4 * 4 * 512 : 1 \rightarrow CBR\ 4 * 4 * 1 : 1 \rightarrow sigmoid \rightarrow output$

Dp : $input \rightarrow CBR\ 4 * 4 * 64 : 2 \rightarrow CBR\ 4 * 4 * 128 : 2 \rightarrow CBR\ 4 * 4 * 256 : 2 \rightarrow CBR\ 4 * 4 * 512 : 2 \rightarrow CBR\ 4 * 4 * 512 : 1 \rightarrow CBR\ 4 * 4 * 1 : 1 \rightarrow output$

B. Visual comparisons

B.1. Comparison on GOPRO dataset

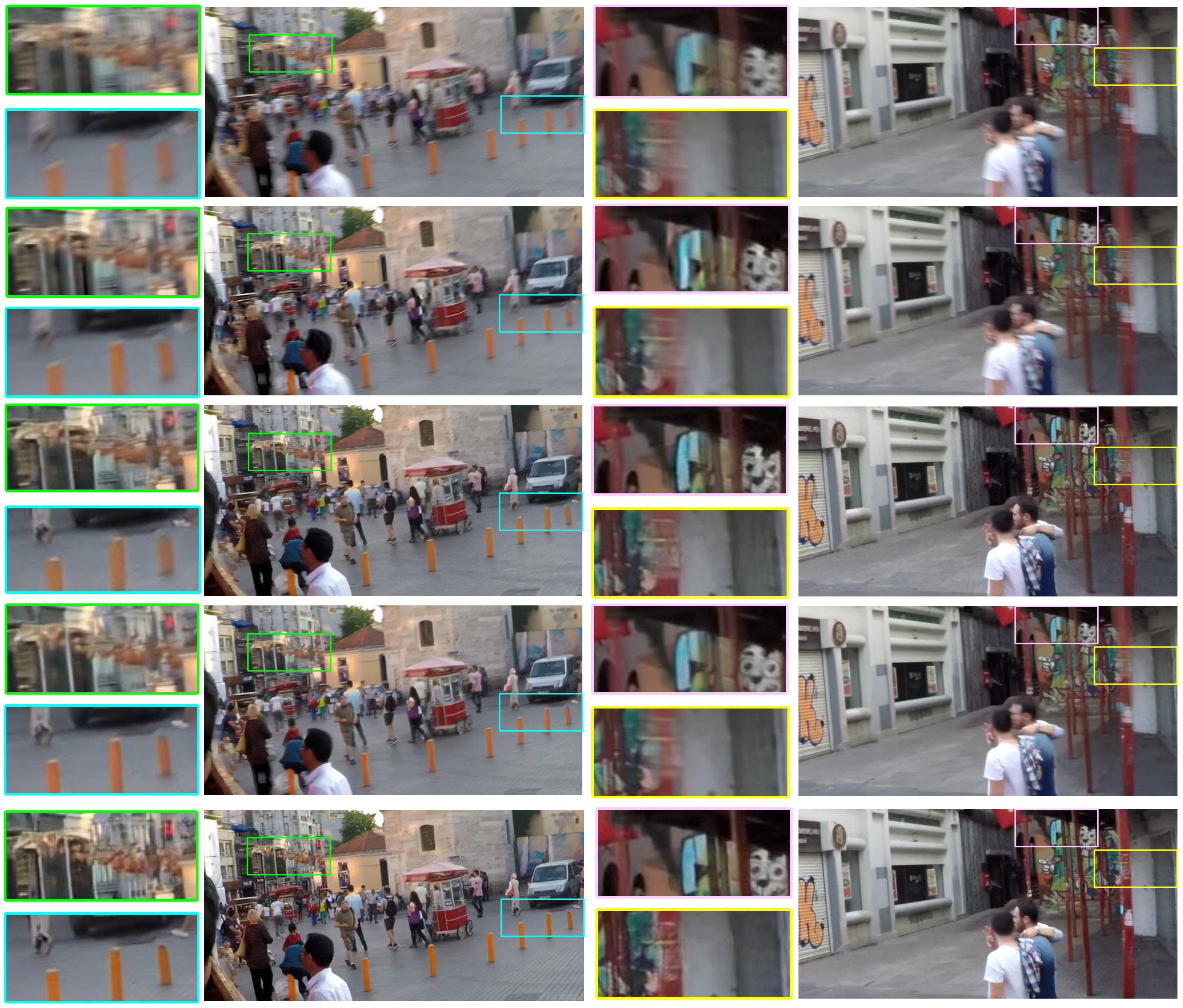

Figure 9: More test results on GOPRO dataset. Images from top to bottom are blurry image, result of Sun *et al.* [26], Nah *et al.* [17], Tao *et al.* [30] and ours respectively.

B.2. The effectiveness of artifacts-penalize discriminator

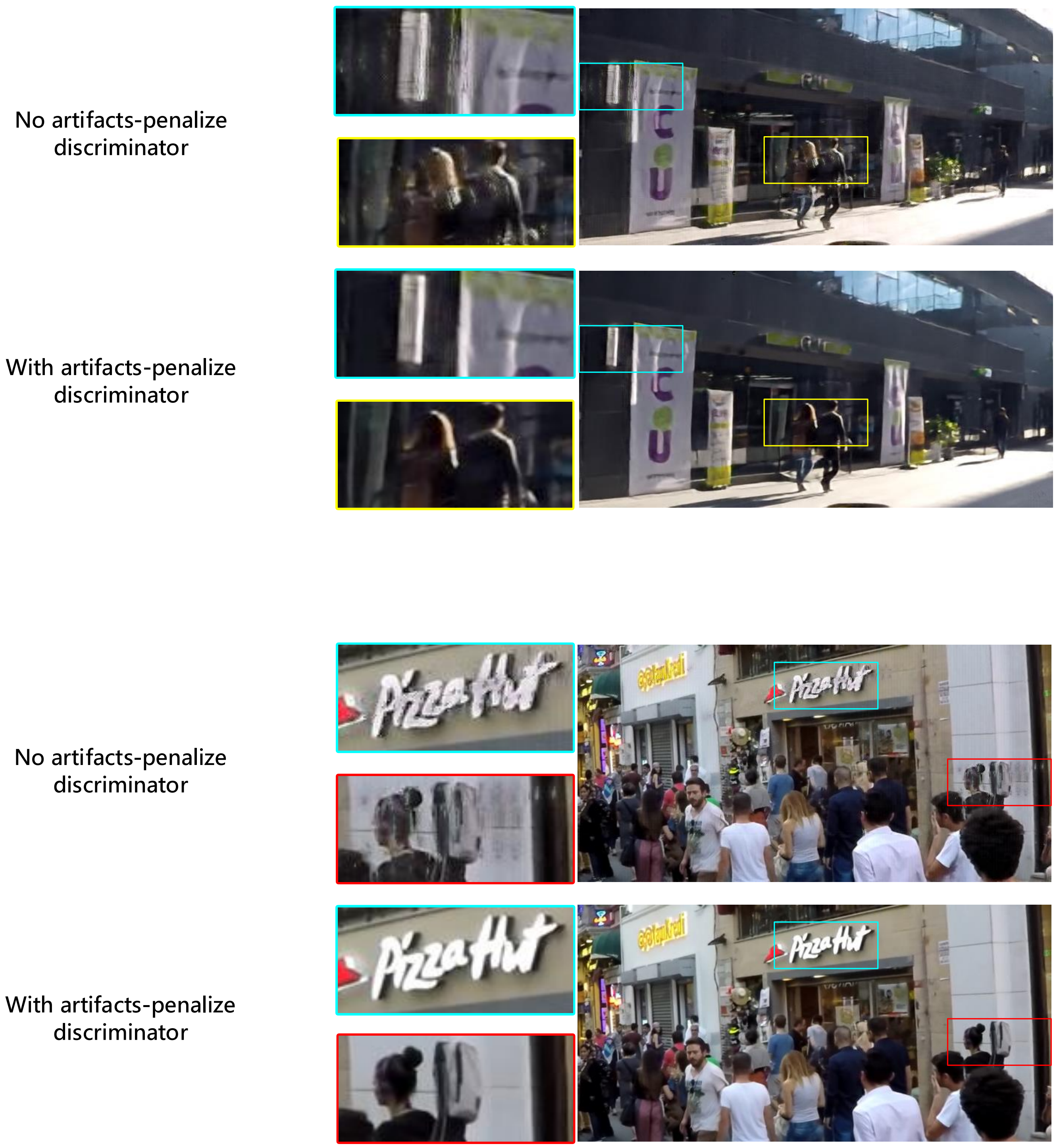
No artifacts-penalize discriminator
With artifacts-penalize discriminator
No artifacts-penalize discriminator
With artifacts-penalize discriminator

Figure 10: Comparison of using artifacts-penalize discriminator before and after. Without artifacts-penalized discriminator, the quality of images with artifacts can't be improved through recurrence. Artifacts stay or even being worse in the end.

B.3. Comparison on real blurred images

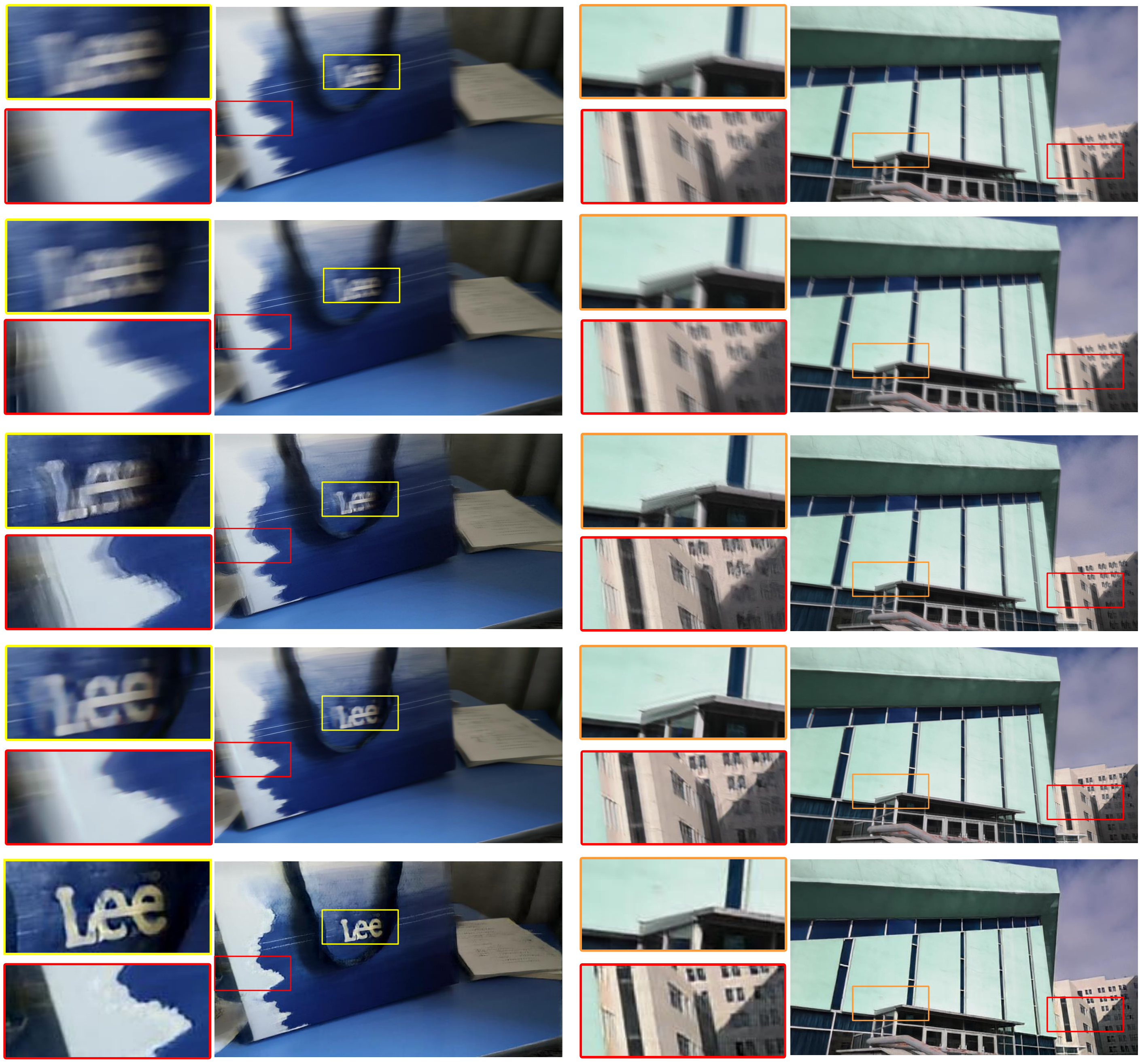

Figure 11: Comparison on real blurred images. Images from top to bottom are blurry image, result of Sun *et al.* [26], Nah *et al.* [17], Tao *et al.* [30] and ours respectively.