DEEP IMAGE DEBLURRING USING LOCAL CORRELATION BLOCK

Wei Su, Yuan Yuan, Qi Wang*

School of Computer Science and Center for OPTical IMagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University, Xi'an 710072, Shaanxi, P.R. China

ABSTRACT

Dynamic scene deblurring is a challenging problem due to the various blurry source. Many deep learning based approaches try to train end-to-end deblurring networks, and achieve successful performance. However, the architectures and parameters of these methods are unchanged after training, so they need deeper network architectures and more parameters to adapt different blurry images, which increase the computational complexity. In this paper, we propose a local correlation block (LCBlock), which can adjust the weights of features adaptively according to the blurry inputs. And we use it to construct a dynamic scene deblurring network named LCNet. Experimental results show that the proposed LCNet produces compariable performance with shorter running time and smaller network size, compared to state-of-the-art learning-based methods.

Index Terms— Dynamic scene deblurring, EncoderDecoder network, Local correlation block

1. INTRODUCTION

Owing to the existence of exposure time, motion blur always occures during taking pictures or videos. Many causes such as camera shake, depth variation and object motion could result in blur artifacts by polluting one pixel with its neighbour pixels in the imaging plane. Blur artifacts downgrad the image quality and it is harmful to other vision tasks such as object detection, classification and object tracking because of the unclear objects. Single image deblurring is to recover the unknown sharp images from the observed blurry images with additional noise. However, it is difficult to solve and highly illposed because the blur pattern and noise are unknown. Many early approaches formulate the blur model as:

$$\mathbf{b} = \mathbf{K}\mathbf{s} + \mathbf{n}, \tag{1}$$

where $\mathbf{b}$, $\mathbf{s}$ and $\mathbf{n}$ indicate the vectorized blurry image, sharp latent image, and additional noise respectively. $\mathbf{K}$ is the blur matrix. Each row of $\mathbf{K}$ represents a local blur kernel which attached to the sharp image to generate a blurry pixel. The solution space is very large and it is hard to solve the $\mathbf{s}$ and $\mathbf{K}$ with $\mathbf{b}$.

To solve this problem, some natural image priors such as total variation (TV) [1], Hyper-Laplacian prior [2], sparse image prior [3], dark channel prior [4], extreme channel prior [5] and *etc.* are introduced to the problem for regularring the solution space. However, the deblurring process needs to solve a non-convex optimization problem. It is time and memory consuming and the quality of results is extremely dependent on the hand-crafted priors. Once the priors are not suitable for the blur pattern, many artifacts such as ringing artifact would appear. Methods in [6] specify the cause of the blur to constraint the blur matrix, *i.e.* if the blur artifacts are caused by shakeness of camera, then it can be modeled as an uniform blur where the blur matrix $\mathbf{K}$ would be a circulant matrix, and thus the model can be simplified as:

$$\mathbf{b} = \mathbf{k} * \mathbf{s} + \mathbf{n}, \tag{2}$$

where * represents the convolution operation. However, there are many moving objects in dynamic scene, so Kim *et al.* [7] proposed a segment-based non-uniform debulr method, in which different segmentation regions share the same blur kernel. Some works [8, 9, 10] approximated the blur kernel to be a local linear kernel, then estimate the latent image and linear kernel jointly.

With the development of deep learning, many researchers try to introduce neural networks into the traditional optimazation based deblurring frameworks to replace some parts of the solution processes. Chakrabarti *et al.* [11] use a fully connected layer network to predict the inverse kernel in the frequency domain. Sun *et al.* [9] and Gong *et al.* [10] apply a convolution neural network (CNN) to predict the local linear blur kernels and then use non-blind deblurring methods to optimize the latent image. Li *et al.* [12] try to learn the natural image prior with a CNN, and use it as a prior regularization term in the optimazation framework. In addition, methods in [13, 14, 15] try to train an end-to-end neural network to deblur the images or videos and achieve state-of-the-art performance. These methods directly generate sharp images from blurry input images without the blur kernel estimation process. However, due to the complexity of dynamic scene deblurring, so many convolution layers should be added into the

*Corresponding author. This work was supported by the National Natural Science Foundation of China under Grant 61632018, 61825603, U1864204 and 61773316.

network to insure there is a enough large receptive field to deal with severe motion blur problems. In addition, the degrees of blur vary in different images but the parameters of deblurring network are fixed. So the current deblurring networks need amounts of parameters to adapt various blurry images, which result in the networks large in size and computationally expensive.

In this paper, we propose a novel local correlation block (LCBlock) and use it to build a dynamic scene deblurring neural network named LCNet. The LCNet is based on the encoder-decoder framework with skip connections. Several LCBlocks are inserted into the decoders. The encoders try to encode the input blurry images to latent feature maps which are combined with the image content information and blur pattern information. The LCBlocks generate weights according to the similarity of the blur pattern information and then use the generated weights to sum the content information. Following the non-local mean [16] and non-local network [17], dot-product similarity is used to calculate the similarity between each pixel and its neighbor pixels in the blur pattern feature map. For getting larger receptive field, we block two LCBlocks together. In general, the contributions are summarized as follows:

- We propose a novel LCBlock, which can extract the content features and blur pattern features. And map the similarity calculated on blur pattern features to weights for the content features adaptively.
- We build LCNet based on LCBlocks. The LCNet is a lightweight dynamic scene deblurring network and achieves better performance.
- Experimental results show that the proposed method achieves better results compared with state-of-the-art methods, and our method has faster running speed and smaller network size.

2. METHODOLOGY

In this section, we will introduce the architecture of the proposed LCBlock and LCNet in detail. In addition, the upsampling with pixelshuffle and loss function used for training will be descriped seperately.

2.1. Network Architecture

The overall architecture of the network is illustrated in Figure 1. The backbone of the LCNet is an encoder-decoder neural network with skip connections. The encoder-decoder networks are combined with encoder networks and decoder networks. The encoders usually contain several encoder blocks (EBlocks), which are used to downsample the feature maps for smaller spatial size, and increase the channels numbers. The decoders which usually contain several decoder blocks

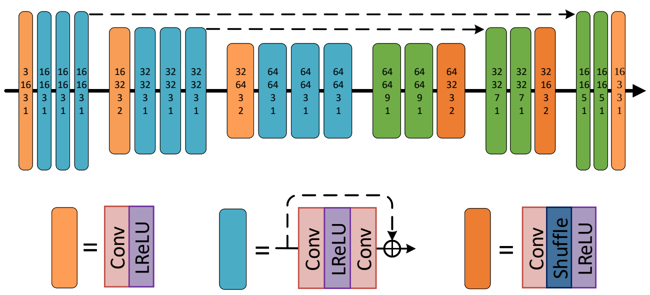

Fig. 1. The architecture of LCNet. The modules colored in green are LCBlocks. The LReLU, Conv and Shuffle represent LeakyReLU, Convolution and Pixel Shuffle respectively. The skip connections are shown as dashed lines. The four numbers in each module are in channels, out channels, kernel size and stride respectively.

(DBlocks) just do the opposite, trying to reduce channels but increasing the spatial size. Residual blocks are added to the decoders and skip connections are used to combine different level feature maps in the encoder and decoder correspondingly. To sum up, there are two advantages of using encoder-decoder network with skip connections to deblur dynamic scene images. The first one is the downsampling and upsampling operations can reduce the amount of calculation in the middle layer and enlarge the receptive field which is very essential for deblur task. And the second one is that only residuals need to learn which is suitable for image-to-image translation tasks.

In the network, we use three EBlocks and three DBlocks. Each EBblock consists of a convolution layer and three residual blocks (res-blocks) . Each DBlock contains two LCBlocks and a convolution layer followed by a pixel shuffle layer. The downsampling rate and upsampling rate are set to 0.5 and 2 respectively. Instead of the res-blocks defined in [18], we use modified res-blocks similar to [13], which is better for image restoration problems by dropping normalization layers, post-activation , and replacing ReLU with LeakyReLU. The kernel size is set to 3 for all convolution layers in the LCNet as we found it is easier to train than that with kernel size 5. The negative-slops of all LeakyReLU are set to 0.2.

2.2. LCBlock

LCBlock is the key component in the construction of deblurring network. The architecture ot LCBlock is illustrated in Figure 2. It contains three convolutions. The first two are used to extract image content features and blurry pattern features seperately, and then squeeze channels of feature maps to reduce the amount of calculation. The last convolution is used to restore the number of channels to their original size. Different from calculating similarity with all elements [17], we only calculate the similarity of elements in the sliding win-

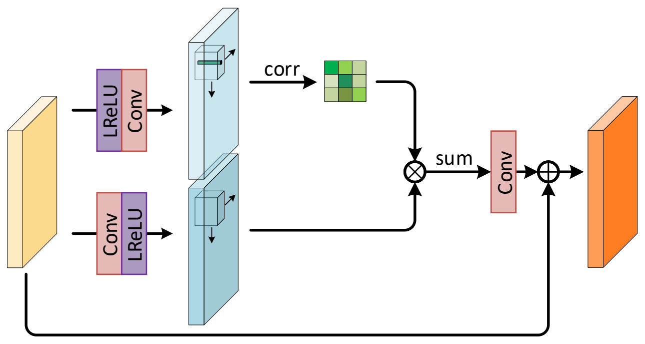

Fig. 2. The architecture of LCBlock. The top feature map represents the blurry pattern feature. The bottom feature map is the content feature. The green cuboid represents centeral elements x_i in the sliding block. The colored grid is the weight map generated by correlation operations.

dow. In each sliding window of the blurry pattern features, we calculate the similarity between the central element with other elements and normalize the similarity to generate normalized weight map. Then perform weighted summation in the corresponding local regions of the content features. The process can be described as:

$$w_j = \frac{exp(x_i^T x_j)}{\sum\limits_{j \in \Omega_i} exp(x_i^T x_j)}, \tag{3}$$

$$y_i = \sum_{j \in \Omega_i} w_j c_j, \tag{4}$$

where w_j is the normalized weight. x_j represents one element of the blurry pattern feature map and c_j represents one element of the content feature map. y_i represents the weighted summation. Ω_i indicates the indices within the local region centered on x_i.

Similar to [17], a short cut connection is added between the input and output feature maps, so it can be inserted into any neural network without affection on the original structure. We use correlation operation to calculate the similarity map. Because one input of the correlation is only one element and each element is a vector, so the correlation operation can be simplified as dot product.

2.3. Upsampling with Pixelshuffle

There are many choices for upsampling in the DBlocks. Parameterless upsampling methods such as bilinear interpolation, nearest neighbor interpolation are usually used as resize methods in image processing. And there are also several upsampling methods with parameters such as transposed convolution, sub-pixel convolution and *et al.*. These convolution while up-sampling methods are always used in image super-resolution and show good performance. Pixelshuffle is an effective way to implement sub-pixel convolution which just rearange pixels in the spatial dimension according to the sampling rate.

Although the transposition convolution has been applied to many tasks and performances well, some studies have shown that using transposed convolution for upsampling is easy to generate images with checkerboard artifacts, which not only happens in the training process, but also in the final results. To avoid the artifacts, Odena *et al.* [19] proposed two alternative approaches. One way is to separate upsampling and convolution, *i.e.* first scale the image using bilinear interpolation or nearest neighbor interpolation and then perform a convolution operation. Another way is to use transposed convolution and ensure that the kernel size can be divided by the stride. This is equivalent to the sub-pixel convolution. At the same computational complexity, sub-pixel convolution has more parameters, so it has better modeling power compared with the standard resize convolutions. So in this paper, we use pixel shuffle as upsampling operations in the DBlocks for getting better results.

2.4. Loss Functions

The loss functions used in this paper is composed with pixel-wise loss and content loss, which is formulated as:

$$\mathcal{L} = \mathcal{L}_{pixel} + \lambda \cdot \mathcal{L}_{content}, \tag{5}$$

where λ is set to 0.1 in our experiments.

Pixel loss. Two classical pixel-level loss functions are MSE and MAE loss, which are also referred as L1 and L2 loss respectively. However, models trained with L2 loss usually generate slightly blurry results. So we use L1 loss as the pixel loss instead, which is formulated as:

$$\mathcal{L}_{pixel} = \frac{1}{N_p} \left\| L - S \right\|_1, \tag{6}$$

where L and S represent the predicted results of LCNet and ground truth images respectively. N_p is the numbers of elements in the L and S.

Content loss. Here, we refer to the loss calculated on semantic features as content loss, such as Patch-GAN and perceptron loss. Each element in the semantic feature corresponds to a local region of the original image, so this loss can focus on restoring the general content and conduct to the recovery of image structures. Like [15], we use the perceptual loss as content loss, which is formulated as:

$$\mathcal{L}_{content} = \frac{1}{2N_c} \left\| \phi_i(L) - \phi_i(S) \right\|_F^2, \tag{7}$$

where N_c is the numbers of elements, and ϕ_i represents the feature map obtained by the i-th layer of VGG-19.

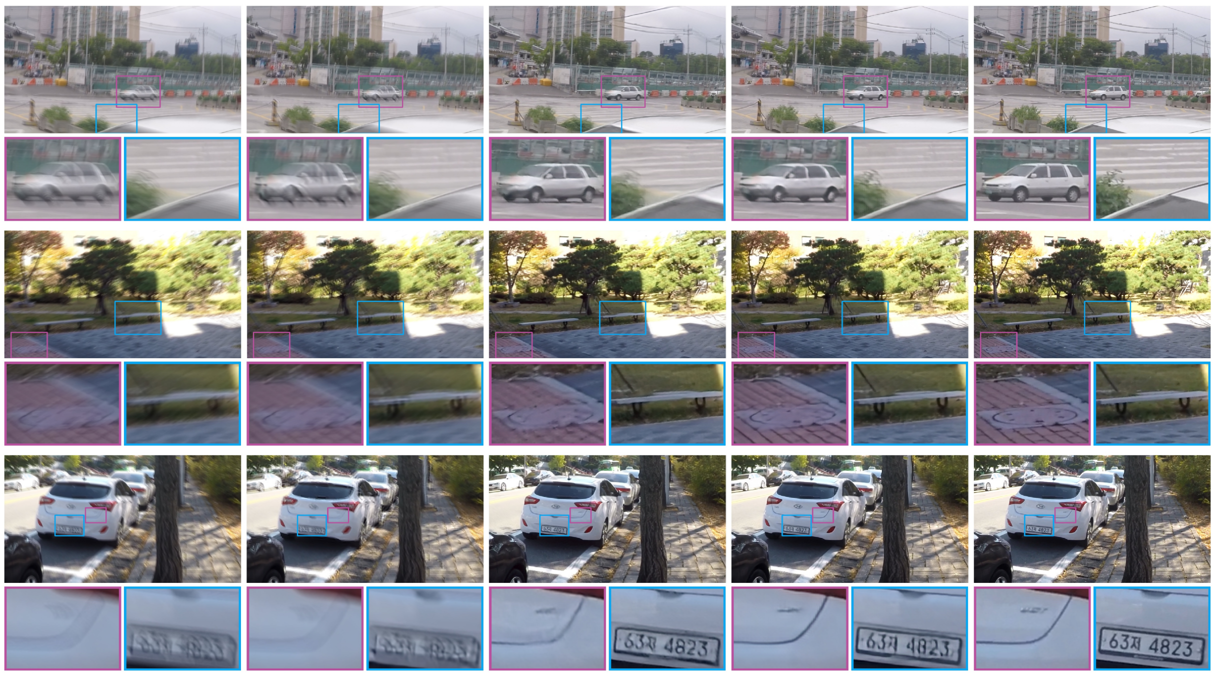

Fig. 3. Deblurring results on the GOPRO test dataset [13]. From left to right: blurry images, results of Sun *et al.* [9], results of Nah *et al.* [13], results of the proposed method, and ground truth images.

3. EXPERIMENTS

In this section, we will show the detailed training configuration and evaluations. All experiments are performed with an i7-6800K CPU and four NVIDIA Geforce GTX 1080Ti GPUs. The model is implemented with the Pytorch 1.0.0.

3.1. Training Details

For fair comparison, we use the GOPRO dataset proposed by [13], and similarly we use 2103 image pairs for training and 1111 image pairs for testing. Each training batch contains 32 image patches. Each patch is randomly cropped and augmentated by randomly flipping and channel shuffling. The size of patch is 200×200 in our experiments.

The parameters of LCNet are initialized with Xavier [20] and updated with Adam [21] optimizer. The learning rate is initialized as 0.001 and decreases by multiplying 0.1 every 500 epochs. 1500 epochs are used for training and experiments show that it is enought to converage the network.

3.2. Evaluation

The trained deblurring network are compared with state-of-the-art deep learning based deblurring methods [9] and [13]. The PSNR, SSIM, model size and running time are evaluated.

The quantitative evaluations are shown in Table 1. It can be seen that our approach achieves the best performance, except for the PSNR item. The size of LCNet is 100× smaller, and the runtime is 150× shorter compared with Nah *et al.*. The qualitative evaluations are shown in Figure 3. From it we can see that the method of Sun *et al.* fails to deblur the blurry images, although it is a non-uniform deblurring method. Compared with Nah *et al.*, our method produced the same or even better results. The recovered images of our method is more clear and better in structures.

Table 1. Quantitative Results on GOPRO Test Dataset

Methods	PSNR	SSIM	Size	Runtime
Kim *et al.*	23.64	0.8239	-	1h
Sun *et al.*	24.64	0.8429	<u>51.5</u>	20min
Nah *et al.*	**29.08**	<u>0.9135</u>	289.5	<u>3.1s</u>
Ours	<u>28.69</u>	**0.9204**	**2.7**	**0.02s**

4. CONCLUSION

In this paper, we propose a novel weight adaptive module named LCBlock, and use it to build a dynamic scene deblurring network called LCNet, which is a encoder-decoder framework. We demonstrate how the LCBlock can adaptively adjust weights according to the different blurry images. Experimental results show that the proposed method achieve compariable performance with state-of-the-art deblurring methods, but with less parameters and shorter running time benefited from the LCBlock.

5. REFERENCES

[1] Tony F Chan and Chiu-Kwong Wong, "Total variation blind deconvolution," *IEEE transactions on Image Processing*, vol. 7, no. 3, pp. 370–375, 1998.

[2] Dilip Krishnan and Rob Fergus, "Fast image deconvolution using hyper-laplacian priors," in *Advances in neural information processing systems*, 2009, pp. 1033–1041.

[3] Anat Levin, Yair Weiss, Fredo Durand, and William T Freeman, "Understanding and evaluating blind deconvolution algorithms," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 1964–1971.

[4] Jinshan Pan, Deqing Sun, Hanspeter Pfister, and Ming-Hsuan Yang, "Blind image deblurring using dark channel prior," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1628–1636.

[5] Yanyang Yan, Wenqi Ren, Yuanfang Guo, Rui Wang, and Xiaochun Cao, "Image deblurring via extreme channels prior," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4003–4011.

[6] Li Xu and Jiaya Jia, "Two-phase kernel estimation for robust motion deblurring," in *European conference on computer vision*. Springer, 2010, pp. 157–170.

[7] Tae Hyun Kim, Byeongjoo Ahn, and Kyoung Mu Lee, "Dynamic scene deblurring," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 3160–3167.

[8] Tae Hyun Kim and Kyoung Mu Lee, "Segmentation-free dynamic scene deblurring," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2766–2773.

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

[10] Dong Gong, Jie Yang, Lingqiao Liu, Yanning Zhang, Ian Reid, Chunhua Shen, Anton Van Den Hengel, and Qinfeng Shi, "From motion blur to motion flow: a deep learning solution for removing heterogeneous motion blur," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2319–2328.

[11] Ayan Chakrabarti, "A neural approach to blind motion deblurring," in *European conference on computer vision*. Springer, 2016, pp. 221–235.

[12] Lerenhan Li, Jinshan Pan, Wei-Sheng Lai, Changxin Gao, Nong Sang, and Ming-Hsuan Yang, "Learning a discriminative prior for blind image deblurring," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6616–6625.

[13] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee, "Deep multi-scale convolutional neural network for dynamic scene deblurring," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3883–3891.

[14] Sainandan Ramakrishnan, Shubham Pachori, Aalok Gangopadhyay, and Shanmuganathan Raman, "Deep generative filter for motion deblurring," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2993–3000.

[15] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiří Matas, "Deblurgan: Blind motion deblurring using conditional adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8183–8192.

[16] Antoni Buades, Bartomeu Coll, and J-M Morel, "A non-local algorithm for image denoising," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. IEEE, 2005, vol. 2, pp. 60–65.

[17] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He, "Non-local neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7794–7803.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[19] Augustus Odena, Vincent Dumoulin, and Chris Olah, "Deconvolution and checkerboard artifacts," *Distill*, vol. 1, no. 10, pp. e3, 2016.

[20] Xavier Glorot and Yoshua Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[21] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.