XYDeblur: Divide and Conquer for Single Image Deblurring

Seo-Won Ji[*, 1] Jeongmin Lee[*, 1] Seung-Wook Kim[*, 2]
Jun-Pyo Hong[1] Seung-Jin Baek[1] Seung-Won Jung[†, 1] Sung-Jea Ko[1]
[1]Korea University, [2]Pukyong National University

Abstract

Many convolutional neural networks (CNNs) for single image deblurring employ a U-Net structure to estimate latent sharp images. Having long been proven to be effective in image restoration tasks, a single lane of encoder-decoder architecture overlooks the characteristic of deblurring, where a blurry image is generated from complicated blur kernels caused by tangled motions. Toward an effective network architecture for single image deblurring, we present complemental sub-solution learning with a one-encoder-two-decoder architecture. Observing that multiple decoders successfully learn to decompose encoded feature information into directional components, we further improve both the network efficiency and the deblurring performance by rotating and sharing kernels exploited in the decoders, which prevents the decoders from separating unnecessary components such as color shift. As a result, our proposed network shows superior results compared to U-Net while preserving the network parameters, and using the proposed network as the base network can improve the performance of existing state-of-the-art deblurring networks.

1. Introduction

Image deblurring is a fundamental image restoration problem in image processing and computer vision, which aims to recover a sharp image from a blurry image caused by a camera or objects in motion. Blurry images affect not only the perceptual image quality but also the performance of various applications such as object detection, image segmentation, and visual odometry. Therefore, despite it being a classical restoration task, image deblurring is still being actively researched.

In general, the degradation model is formulated as follows:

$$\mathbf{x} = \mathbf{y} * \mathbf{k} + \mathbf{n}, \tag{1}$$

where $\mathbf{x}$, $\mathbf{y}$, $\mathbf{k}$, $\mathbf{n}$, and $*$ denote the blurry image (observation), sharp image (latent image), blur kernel, additive random noise, and 2D convolution operation, respectively. Since only the blurry image $\mathbf{x}$ is given and the other terms unknown, solving Eq. (1) is considered to be ill-posed, having multiple solutions to a problem. In traditional studies, researchers employed blur kernel estimation followed by deconvolution or regularization using natural image priors in attempts to handle the ill-posedness of the deblurring problem [8, 18, 28, 30, 32]. Thanks to the tremendous advances in deep learning and large-scale data accessibility, numerous methods using convolutional neural networks (CNNs) have recently been proposed and have achieved great success [2, 9, 10, 16, 27, 29, 31, 33, 34]. Many deblurring methods adopt U-Net [25] as their base architecture, which has achieved state-of-the-art performance. U-Net consists of a common CNN-based encoder, successively decreasing the feature resolution to extract high-level image context, and a symmetrically structured decoder, increasing the feature resolution back to the input resolution to generate a restored image. Since the local information at each feature resolution in the encoder is transferred to the one at the corresponding resolution in the decoder via skip connection, U-Net can effectively handle multi-scale degradation of images.

Motion blur is originated from multiple tangled motions of moving objects in the scene and/or camera shakes. We claim that the common standard of using the unitary lane of U-Net is insufficient for handling the complicated nature of image blurs. We thus propose a simple but effective solution to this problem by dividing the original problem into multiple sub-problems [21]. In terms of network design objective, splitting a decoder can divide the deblurring problem into sub-problems; in other words, the solution space required for deblurring can be decomposed by explicitly separated decoder networks. The proposed network consists of a single encoder and two decoders as shown in Fig. 1, which induces two decoders to separately solve two sub-problems of deblurring. One decoder implicitly generates a principal residual in the given 2D scene, and the other one generates the complement residual of its separated decoder.

*Equal contribution
†Corresponding author: Seung-Won Jung (swjung83@korea.ac.kr)

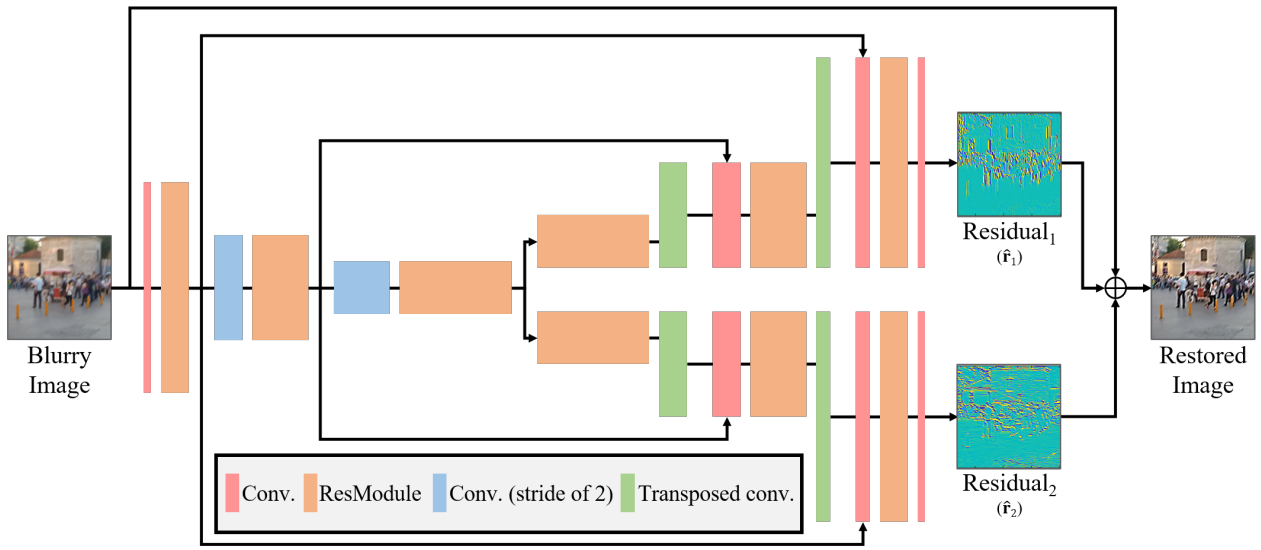

Figure 1. Detailed architecture of XYDeblur.

We observe that without any explicit supervision, the principal and its complement residuals contain blurred edges along the direction of horizontal and vertical axes in an image plane, respectively, which is why we call the proposed network *XYDeblur*. Based on this observation, we further improve the one-encoder-two-decoder structure of XYDeblur by spatially rotating the convolutional kernels in one decoder and sharing the parameters with the other decoder. Unlike conventional parameter sharing approaches in CNN that sacrifice the performance for network efficiency [1, 17], the separated decoders with shared and rotated parameters not only reduce the number of parameters but also improve the performance by eliminating undesired disentanglement of features that are irrelevant to the deblurring task.

We demonstrate the effectiveness of XYDeblur as compared to U-Net while consuming the same network parameters. We also show the extensibility of the proposed approach by substituting the base structure in state-of-the-art deblurring networks with the proposed architecture. Experimental results validate that the proposed approach successfully guides the network to learn complementary sub-solutions and improves the deblurring performance. The contribution of the proposed method can be summarized as follows:

1. To the best of our knowledge, we are the first to introduce a one-encoder-two-decoder architecture in image deblurring that derives each output from two decoders to have complementary sub-solutions, which are visually orthogonal in the spatial domain.

2. XYDeblur shares rotated convolutional kernels from one decoder with the other, thereby substantially improving the deblurring performance while using the same number of parameters as the standard U-Net architecture.

3. The proposed network can be implemented in many U-Net-based state-of-the-art networks without increasing the model size, and extensive experimental results show that substituting the U-Net with our proposed network can improve the deblurring performance.

2. Related Works

2.1. Image deblurring using U-Net

Most image restoration methods consider the multi-scale representation of image degradation as their key problem [5, 12, 16, 19, 35]. With a U-Net architecture, several image restoration models estimate a latent image iteratively as follows:

$$\{\hat{\mathbf{y}}_{k+1}, \mathbf{h}_{k+1}\} = \mathcal{F}_{k+1}(\mathbf{x}, \{\hat{\mathbf{y}}_k, \mathbf{h}_k\}; \Theta_{k+1}) \quad \text{for } k \in [0, K-1], \tag{2}$$

where $\mathbf{y}_{k+1}$, $\mathbf{h}_{k+1}$, and $\mathbf{x}$ denote the estimated image at the $(k+1)$-th iteration, the hidden features at the $(k+1)$-th iteration to be transferred to the next stage, and the input blurry image, respectively, and $\mathcal{F}_{k+1}(\cdot)$ is the U-Net-based model with the parameter set Θ_{k+1} at the $(k+1)$-th iteration.

Many works on image deblurring also follow such trends [2, 16, 27, 29, 33, 34], in which the difference between the estimation and the ground-truth image is gradually predicted by applying U-Nets from the small-to-large scale, or vice versa. In [2, 16, 29], a coarse-to-fine approach is adopted by transferring the deblurred information for the down-sampled image resolution to the larger scale image input. In [27, 33, 34], the deblurring networks take finely-split multi-patch hierarchy as their inputs and transfer the

sharpened information to the next stage. At the next stage, wider patches, which are made up of several patches from the previous stage, are recovered.

2.2. One-Encoder-Multiple-Decoder architecture

In multi-task learning, a multi-head decoder with a single encoder is a common network architecture. The single encoder extracts feature embedding having jointly tangled information of the input, and the multiple decoders perform their assigned tasks, which are supervised by given labels. For example, an image decomposition problem, which is one of the multi-task learning problems, can be dealt with the one-encoder-multiple-decoder architecture [6, 13]. To train those multiple decoders, the labels for each task are given [6] or the complicated loss functions are designed using the given data and prior knowledge [13]. In generative models, an additional decoder to a single encoder-decoder network is often employed to guide the encoder [11] or the other decoder [26]. Unlike our proposed method, those methods temporarily adopt a one-encoder-two-decoder architecture to regulate the learning process and utilize the conventional U-Net structure for test.

3. Proposed Method

3.1. Architecture of the proposed XYDeblur

Fig. 1 illustrates the proposed one-encoder-two-decoder architecture. We design our network to estimate the residual image $\mathbf{r} = \mathbf{y} - \mathbf{x}$ from the blurry image. Formally, the residual estimation, $\hat{\mathbf{r}}$, can be obtained via the proposed network $\mathcal{F}$ with the learnable parameters Θ as follows:

$$\hat{\mathbf{r}} = \mathcal{F}(\mathbf{x}; \Theta). \tag{3}$$

We obtain the residual estimation using two separate decoder networks $\mathcal{D}_{\text{hor}}$ and $\mathcal{D}_{\text{ver}}$ as follows:

$$\hat{\mathbf{r}} = \mathcal{D}_{\text{hor}}(\mathbf{z}; \Theta_{\text{hor}}) + \mathcal{D}_{\text{ver}}(\mathbf{z}; \Theta_{\text{ver}}), \tag{4}$$

where Θ_{hor} and Θ_{ver} are the network parameters of $\mathcal{D}_{\text{hor}}$ and $\mathcal{D}_{\text{ver}}$, respectively, and $\mathbf{z}$ represents the encoded image feature. Here, the encoded image feature is obtained by

$$\mathbf{z} = \mathcal{E}(\mathbf{x}; \Theta_{\text{e}}), \tag{5}$$

where $\mathcal{E}$ denotes the encoder with the parameter of Θ_{e}. It is worth noting that the two decoders $\mathcal{D}_{\text{hor}}$ and $\mathcal{D}_{\text{ver}}$ have the same structure. If we use the encoder and only one of the decoders, the network structure reduces to a common U-Net structure.

3.2. Separated decoders: 2D blur decoupling

The proposed network forces the information of the blur degradation in the encoded feature $\mathbf{z}$ to be divided through the two separated decoders. According to the linear span theory [14, 15], the linear reconstruction composed of multiple independent regression outputs can encompass a larger output space because the independent regression networks learn the complementary features to span a whole output space. Ideally, it is desired that the two decoders result in independent regression outputs to maximize the solution space. In other words, it is expected that one decoder produces the blur residual along the principal axis, *i.e.*, the degradation components with the largest variation, and the other one generates the complementary residual to complete the linear reconstruction.

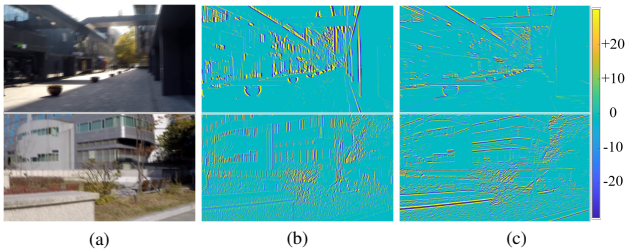

Figure 2. Sample images and sub-solutions for each decoder: (a) Blurry images; (b) results of the first decoder ($\hat{\mathbf{r}}_1$); (c) results of the second decoder ($\hat{\mathbf{r}}_2$).

Fig. 2 depicts an example of the estimated residuals from the two decoders, $\mathcal{D}_{\text{hor}}$ and $\mathcal{D}_{\text{ver}}$.[1] Interestingly, the two residuals obtained using the separated decoders contain the horizontal and vertical motion components. This example implies that the proposed design enables the network to decompose the information inherent in the encoded features into the ones along the x- and y-axes without applying any explicit constraint. This motivates us to propose the scheme to share the parameters of the decoders for more efficient deblurring.

3.3. Spatial kernel rotation for parameter sharing

Even though the separated decoders can successfully decouple features related to deblurring along x- and y-axes, there is also a possibility that undesirable information decoupling could occur since we do not give any supervision to the network. This is problematic in terms of network capacity [3] and solution space [14, 15] of the network because the undesired division of complementary features limits the potential solution space and degrades the network performance. For example, Figs. 3(a)-(c) illustrate the color shift caused by the two separated decoders. Specifically, the two decoders, U-Net^{2D}, produce the results with the color temperatures shifted to the lower and higher levels, respectively. Such color shift hinders the network from fully exploring the solution space and does not provide any clues for deblur-

[1] The network was trained on the GoPro dataset [16]. More detailed experimental setup can be found in Sec. 4.1.

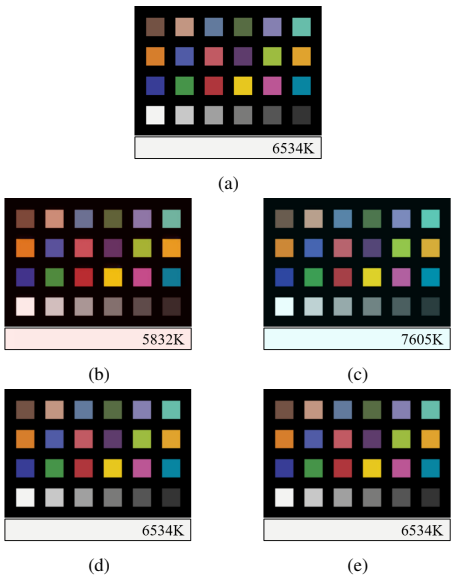

Figure 3. Input ColorChecker image and resultant images. The magnified color white of the chart and its color temperature are shown in the bottom row. (a) ColorChecker image; (b) input + $\hat{\mathbf{r}}_1$ of U-Net^{2D}; (c) input + $\hat{\mathbf{r}}_2$ of U-Net^{2D}; (d) input + $\hat{\mathbf{r}}_1$ of XYDeblur; (e) input + $\hat{\mathbf{r}}_2$ of XYDeblur.

ring (the color shift will be compensated when the residuals are combined).

To solve this problem, we employ the properties of the separated decoders. Since the two decoders can learn the inherent information along the axes at a right angle, we share the network parameter of the two decoders as follows:

$$\hat{\mathbf{r}} = \mathcal{D}_{\text{hor}}(\mathbf{z}; \Theta_{\text{d}}) + \mathcal{D}_{\text{ver}}(\mathbf{z}; \Theta_{\text{rd}}), \tag{6}$$

where Θ_{rd} is the network parameter of Θ_{d} rotated 90 degrees counterclockwise. We select $\mathcal{D}_{\text{hor}}$ as a primary decoder. On average, the resultant residual from $\mathcal{D}_{\text{hor}}$ has more signal power than that from $\mathcal{D}_{\text{ver}}$, which means that the larger variations occur along the horizontal axis as compared to the vertical axis. This is natural as a camera can rotate full 360 degrees around the yaw axis but the pitch rotation is rather limited while photographing. The proposed network design with Eq. (6) has two advantages. First, the decoders are forced to share the other information except for the one along the axial directions, which regularizes the learning process of the network. Thus, the network can focus only on eliminating blur components. As shown in Figs. 3(d) and (e), sharing the rotated kernels guides the network to learn complementary features related to the horizontal and vertical axes and no longer yields unnecessary color shifts. Second, the required parameters for the two separated decoders are halved. Consequently, XYDeblur not only improves the efficiency but also the deblurring performance, which can be seen in Table 1.

3.4. Extension to existing deblurring networks

XYDeblur can be applied to any conventional networks based on U-Net such as [2, 34]. For example, if we extend the proposed network to the iterative deblurring method explained in Eq. (2), the estimation at the $(k+1)$-th iteration can be obtained as follows:

$$\begin{aligned}\hat{\mathbf{y}}_{k+1} = \mathbf{x} &+ \mathcal{D}_{\text{hor}}^{(k+1)}(\bar{\mathbf{z}}_{k+1}; \Theta_{\text{d}}^{(k+1)}) \\ &+ \mathcal{D}_{\text{ver}}^{(k+1)}(\bar{\mathbf{z}}_{k+1}; \Theta_{\text{rd}}^{(k+1)}),\end{aligned} \tag{7}$$

$$\bar{\mathbf{z}}_{k+1} = \mathcal{T}(\bar{\mathbf{z}}_k, \mathbf{z}_{k+1}), \tag{8}$$

$$\mathbf{z}_{k+1} = \mathcal{E}^{(k+1)}(\hat{\mathbf{y}}_k; \Theta_{\text{e}}), \tag{9}$$

where $\mathcal{T}(\cdot,\cdot)$ is a combining function that aggregates the encoded feature at the current iteration, $\mathbf{z}_{k+1}$, and the transferred features from the previous iteration, $\bar{\mathbf{z}}_k$. In the following section, we compare the U-Net and XYDeblur using several top-performing deblurring methods to confirm the efficiency and effectiveness of the proposed method.

4. Experiments

In this section, we first describe the implementation details of XYDeblur. We validate the effectiveness of the proposed method by conducting experiments on the following two scenarios: 1) the vanilla proposed network; 2) substituting the U-Net in top-performing methods with XYDeblur. Extensive experimental results for the proposed method can be seen in the supplemental material.

4.1. Experimental setting

Datasets. For training of the networks, GoPro training dataset [16] was used. GoPro dataset is a realistic dataset with blurry images generated by averaging consecutive image frames. It contains 2,103 pairs of blurry and sharp images from 22 different scenes. The images were randomly cropped into 256×256 patches for training. For testing, GoPro test dataset with 1,111 image pairs was used, and 980 pairs of RealBlur test dataset [24], which is a dataset with real blurred images, was also used.

Implementation details. The detailed architecture of XYDeblur is described in Fig. 1. Each ResModule consists of six residual blocks with batch normalization excluded from the original residual block [16]. The number of channels in the feature map starts at 32 and is doubled (halved) every time the resolution of the feature map is halved (doubled). All the experiments were conducted on a PC with an Intel i7-8700 CPU, 32GB of RAM, and an NVIDIA Titan Xp GPU. We used the PyTorch [20] to implement the proposed method. For training, we adopted the Adam [7] optimizer with a batch size of four. The learning rate was initialized as 10^{-4}, which was halved after every 500 epochs.

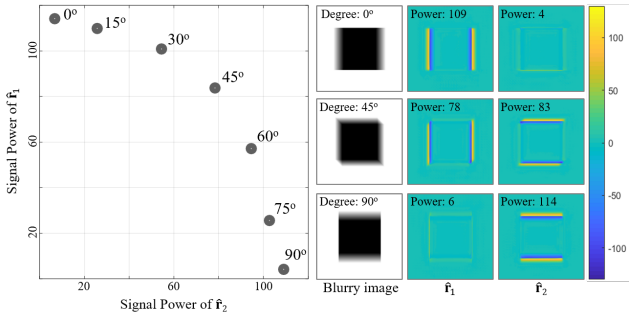

Figure 4. Signal power of $\hat{\mathbf{r}}_1$ and $\hat{\mathbf{r}}_2$ for different degrees of blur: from 0° (horizontal) to 90° (vertical).

The ℓ_1 criterion was applied as a loss function, which is formulated as

$$L = \frac{1}{N}\sum_{n=1}^{N}\left\|\hat{\mathbf{y}}^{(n)} - \mathbf{y}^{(n)}\right\|_1, \tag{10}$$

where $\mathbf{y}^{(n)}$ is the nth target sharp image, $\hat{\mathbf{y}}^{(n)}$ is the corresponding estimation, and N is the number of samples in a mini-batch. The proposed network was trained for 1,300 epochs and took about 65 hours. For state-of-the-art networks, each network and its variations were trained until the peak signal-to-noise ratio (PSNR) of their baseline network reached its reported rate.

4.2. Proposed network - standalone

To analyze the effectiveness of the proposed approach, three experiments were conducted as follows: 1) the analysis of sub-solutions to confirm whether the proposed network learns complementary solutions as intended, 2) the investigation for a suitable number of decoders and the verification of the rotated-parameter sharing, and 3) the examination to ensure that the proposed approach solves the deblurring problem in a stable and robust manner in different cases.

The impact of the one-encoder-two-decoder architecture. We attempt to validate that the proposed approach induces the network to regress two complementary sub-solutions as intended. To this end, the synthetically blurred images with varying blur directions from 0° (horizontal) to 90° (vertical) were used for test. To quantitatively measure how much high-frequency components are restored by the two decoders, $\mathcal{D}_{\text{hor}}$ and $\mathcal{D}_{\text{ver}}$, the signal power [22] is computed as follows:

$$\text{Power} = \frac{1}{M}\|\mathbf{s}\|_2^2, \tag{11}$$

where $\mathbf{s}$ and M are the vector/matrix form of discrete signal and the number of elements in $\mathbf{s}$, respectively. Fig. 4 shows the signal power of $\hat{\mathbf{r}}_1$ and $\hat{\mathbf{r}}_2$ for the images blurred in seven different directions. As expected, the closer the blur direction is to the horizontal direction, the signal power of $\hat{\mathbf{r}}_1$ increases; and the closer to the vertical direction, the signal power of $\hat{\mathbf{r}}_2$ increases. The input blurry images with blur direction of 0°, 45°, and 90° and their corresponding $\hat{\mathbf{r}}_1$ and $\hat{\mathbf{r}}_2$ results are depicted in the last three columns of Fig. 4. This result shows that the two decoders successfully decouple the blur components according to the direction of the blur.

Table 1. Performance of various number of decoders

Method	GoPro		Complexity*	
	PSNR	SSIM	Params	GMACs
U-Net	30.61	0.9458	4.92	687.93
U-Net^{2D}	30.90	0.9489	7.42	1059.77
U-Net^{3D}	30.99	0.9496	9.92	1431.61
U-Net^{4D}	31.04	0.9506	12.42	1803.45
XYDeblur	30.97	0.9502	4.92	1059.77

* The number of parameters is measured in million.
* GMACs is estimated for the input size of 720P.

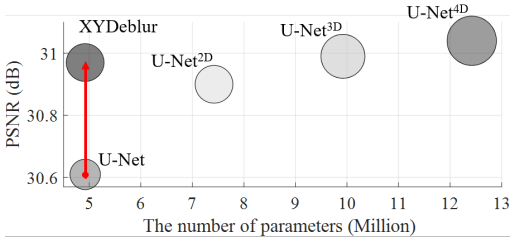

Figure 5. Various number of decoders and their performance. The radius of the circle indicates GMACs of the network.

The proper number of decoders and the impact of sharing the rotated parameters. Increasing the number of decoders leads to the division of a problem into multiple sub-problems and potentially expands the coverage of the solution space of the network. To justify that the use of two decoders is sufficient for the proposed network, we trained U-Nets with one to four decoders referred to as U-Net, U-Net^{2D}, U-Net^{3D}, and U-Net^{4D}, respectively.

As shown in Table 1 and Fig. 5, the deblurring performance improves as the number of decoders used in the network increases. However, unlike the network complexity, which linearly increases as the number of decoders increases, the deblurring performance improvement is abated rapidly after the point when there are two decoders.

In addition, despite the proposed network having the same network structure as U-Net^{2D}, Table 1 shows that sharing the rotated parameters in the proposed network reduces the network complexity almost by half and even enhances the performance. According to [15], when each branch of a network occupies a feature space (or solution space) that

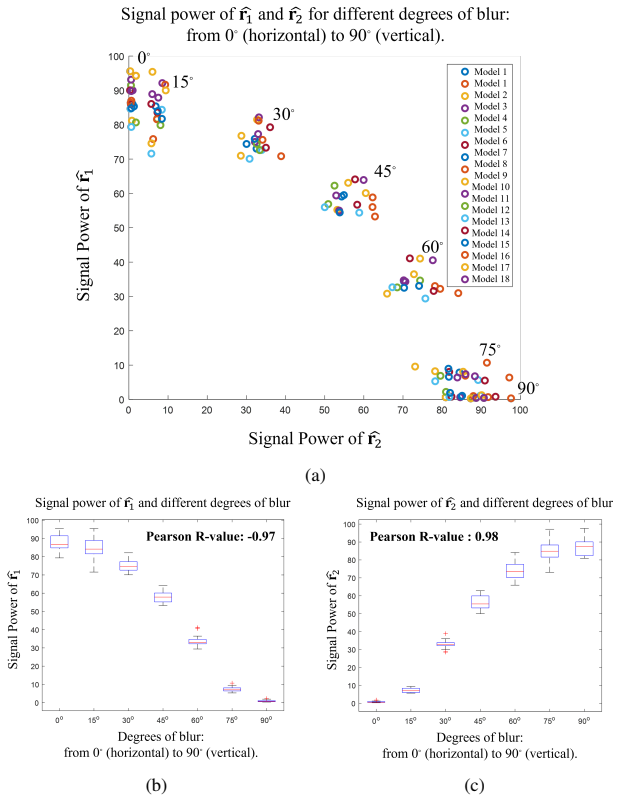

Figure 6. Statistics to certify the consistency of the proposed approach. (a) Signal power of $\hat{\mathbf{r}}_1$ and $\hat{\mathbf{r}}_2$ of different 18 models; (b) correlation between the power of $\hat{\mathbf{r}}_1$ and the degree of blur; (c) correlation between the power of $\hat{\mathbf{r}}_2$ and the degree of blur.

is complementary to each other, it can be regarded that the capacity of the network is most efficiently exploited. In this sense, both U-Net^{2D} and the proposed XYDeblur employ two decoders that learn complementary features and ultimately expand the solution space of the networks. However, while the two networks use the same number of decoders, Figs. 3 shows that sharing the rotated parameters guides the network to eliminate the division of complementary features other than directional components related to the rotation of parameters, which is the most relevant to deblurring. As a result, each decoder in XYDeblur can utilize the same amount of network capacity as decoders in U-Net^{2D} but use the network space solely on deblurring. The positive effects brought by sharing of the rotated parameters are reflected in the performance between the two networks.

The consistency of the proposed approach. To ensure that the proposed model learn the complementary residuals in a stable and robust manner, we randomly divided the GoPro training DB into three and trained 18 models (six models for each divided DB) for 900 epochs using random initial weights with different seeds. Using the same sample image shown in Fig. 4, we measured the signal power of $\hat{\mathbf{r}}_1$ and that of $\hat{\mathbf{r}}_2$ for different motion blur along with the seven directions. Through this experiment, it can be confirmed that even if the model is initialized with different weights and trained on different DB, the network consistently learns to solve the deblurring problem in the complementary manner that divides the problem along the vertical and horizontal axes, as shown in Fig. 6 (a). The Pearson's R values of the signal power of the decoder $\hat{\mathbf{r}}_1$ (-0.97) and the decoder $\hat{\mathbf{r}}_2$ (0.98) with respect to the blur angle further support our claim, as shown in Figs. 6 (b) and (c).

4.3. State-of-the-Art deblurring networks with XYDeblur architecture

To further validate the applicability and extensibility, we applied the proposed network to more complicated state-of-the-art image deblurring networks. Among the top-performing single image deblurring networks in which the authors released the source code, PSS-NSC [2] and DMPHN [34] were chosen, and the first end-to-end deblurring method MSCNN [16] was also used for the experiment. For each network, the original network and the following three network variations were tested: Network$^{\text{Channel}\uparrow}$, Network$^{\text{Layer}\uparrow}$, and Network$^{\text{Ours}}$. In the case of the DMPHN in which various network candidates exist, the one with the lowest complexity was used. Based on the VRAM usage of the Network$^{\text{Ours}}$, Network$^{\text{Channel}\uparrow}$ and Network$^{\text{Layer}\uparrow}$ were constructed to occupy similar VRAM by increasing the number of channels in the feature maps and the number of the convolutional layers located in the decoders, respectively.[2] Each baseline network was trained until it reached its reported PSNR, and the three network variations were trained using the same number of epochs of their baseline network.

GoPro. The results are listed and visualized in Table 2 and Fig. 7 (a). Network$^{\text{Ours}}$ required the fewest number of parameters along with the baseline network, but it showed the highest performance in comparison with the other network variations. On the contrary, despite a significant increase of network parameters, Network$^{\text{Channel}\uparrow}$ and Network$^{\text{Layer}\uparrow}$ did not show consistent performance improvements over the baseline networks. For DMPHN and MSCNN, DMPHN$^{\text{Channel}\uparrow}$ and MSCNN$^{\text{Layer}\uparrow}$ recorded decreased PSNR compared to its baseline model. The results indicate that simply increasing the number of convolutions or feature channels in the decoders cannot effectively utilize the information in the encoded features, primarily due to over-fitting caused by over-parameterization. On the other hand, XYDeblur consistently improves the baseline deblurring networks while maintaining the number of network pa-

[2]More detailed experimental settings can be found in the supplemental material.

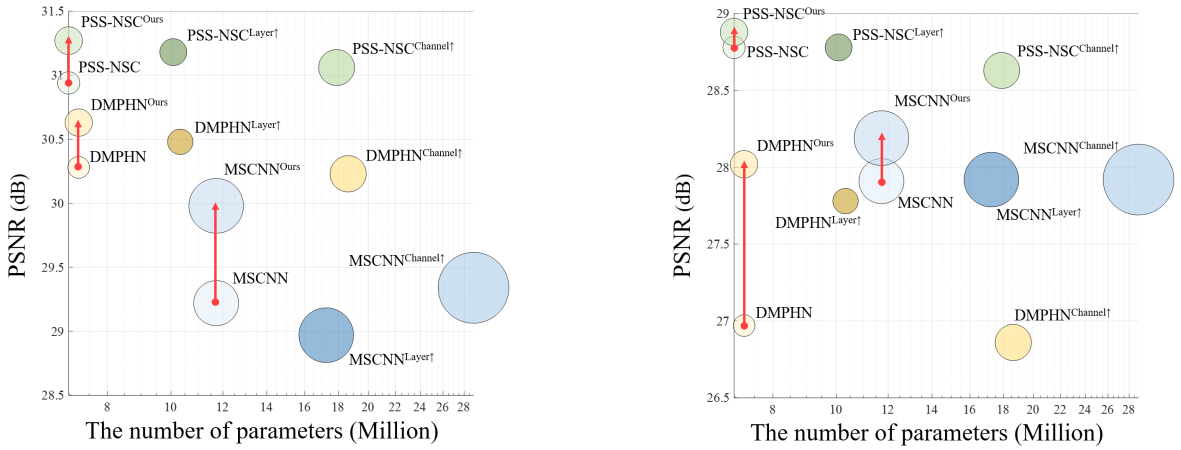

Figure 7. Effectiveness of the proposed approach on the state-of-the-art networks. The radius of the circle indicates GMACs of the network. (a) Performance changes in GoPro test datasets; (b) performance changes in RealBlur test datasets.

Table 2. The proposed approach in the top-performing networks

Method	Variations	GoPro		RealBlur		Complexity†		
		PSNR	SSIM	PSNR	SSIM	Params	GMACs	VRAM
PSS-NSC	Baseline*	$30.94_{(30.92)}$	0.9494	28.78	0.8716	6.98	1167.24	3.79
	Channel↑	31.06	0.9509	28.63	0.8667	17.89	3020.81	5.97
	Layer↑	31.18	0.9524	28.78	0.8721	10.08	1703.95	5.55
	Ours	31.27	0.9531	28.88	0.8765	6.98	1784.39	5.68
DMPHN	Baseline*	$30.28_{(30.25)}$	0.9408	26.97	0.8170	7.23	1100.18	1.62
	Channel↑	30.23	0.9414	26.86	0.8201	18.63	3060.28	2.52
	Layer↑	30.48	0.9443	27.78	0.8366	10.33	1509.11	2.30
	Ours	30.63	0.9458	28.02	0.8459	7.23	1754.07	2.41
MSCNN	Baseline*	$29.22_{(29.08)}$	0.9273	27.91	0.8442	11.72	4728.69	3.21
	Channel↑	29.34	0.9292	27.92	0.8432	28.94	11673.07	4.84
	Layer↑	28.97	0.9250	27.92	0.8449	17.25	6960.32	4.71
	Ours	29.98	0.9382	28.19	0.8550	11.72	6966.13	4.78

* The PSNR in parentheses is the value reported by the authors in their original paper.
† The number of parameters is measured in million.
† GMACs is estimated for the input resolution of 720P (1280×720).
† The usage of VRAM is measured in GB with the input size of 256×256.
The increased and decreased PSNR and SSIM value compared to the baseline method are shown in red and blue, respectively.

rameters. Fig. 8 shows the experimental results for three networks in the GoPro test data. When the proposed approach is applied to the conventional networks, it induces the networks to solve the deblurring problem by dividing it in a complementary manner (see $\hat{\mathbf{r}}_1$ and $\hat{\mathbf{r}}_2$ in Fig. 8), and it can be confirmed that qualitatively better results are obtained compared to the baseline network. [3]

RealBlur. Unless the training and test data are collected from the same distribution, even negligible data differences can significantly degrade the performance of top-performing deep learning networks [23]. This generalization problem is observed more severely as the structure of the network becomes more complex than necessary [4]. Therefore, the generalization ability of a network is one of the essential parts of evaluating deep-learning-based approaches. To confirm the generalization ability of the proposed network architecture on the state-of-the-art networks, we set a domain gap between training and test sets. In this experiment, we trained the network variations using the GoPro dataset and tested the networks with the RealBlur dataset. The results are shown in Table 2 and Fig. 7 (b). As compared to the previous experiment on the GoPro test set, the performance results for all networks are degraded. PSS-NSC$^{\text{Channel}\uparrow}$ and DMPHN$^{\text{Channel}\uparrow}$ even show lower performance than their baseline networks. On the contrary, the

[3] More experimental results can be found in the supplemental material.

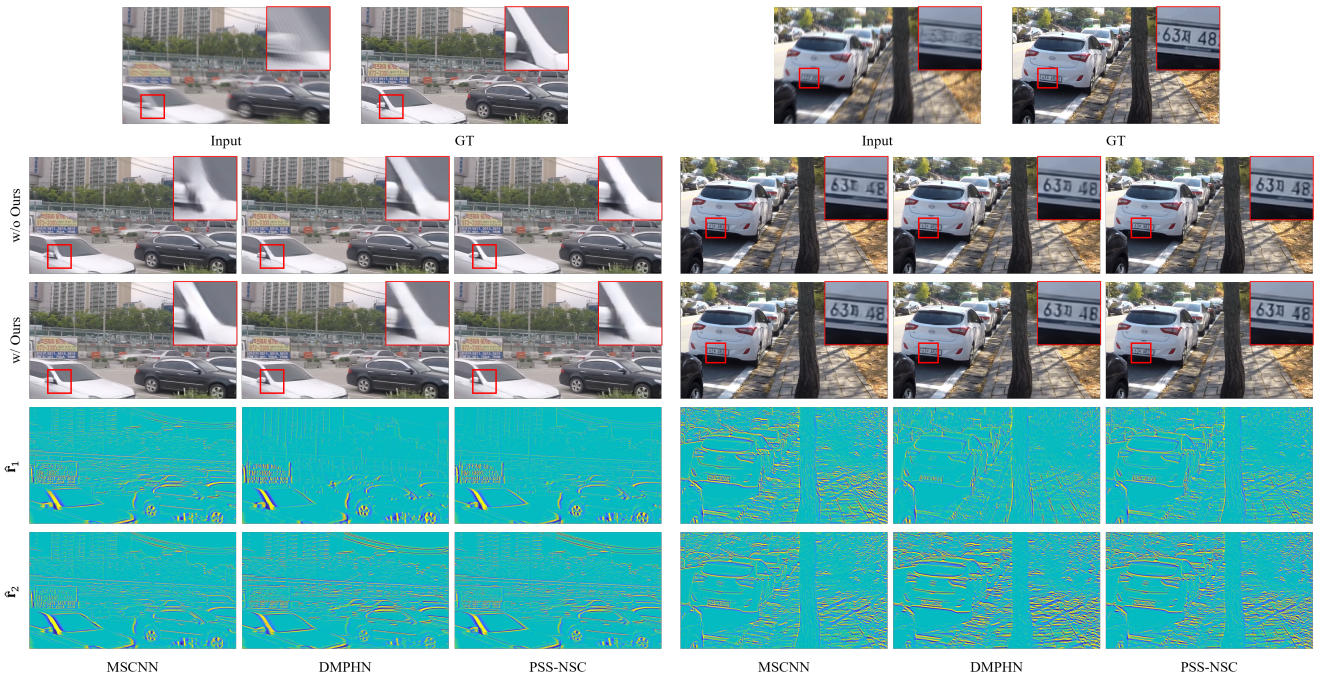

Figure 8. Experimental results for MSCNN, DMPHN, PSS-NSC with and without the proposed approach from the GoPro dataset.

proposed approach consistently recorded performance improvement in terms of both PSNR and SSIM as compared with the baseline networks.

5. Conclusion

In this paper, we propose a novel one-encoder-two-decoder network for single image deblurring. We showed that the proposed architecture guides the network to divide the original deblurring problem into two sub-problems by separating the blur residuals along the x-axis and y-axis, each of which is assigned to each decoder. In addition, we share the rotated parameters of one decoder with the other to prevent any undesired division of complementary features. Extensive performance evaluation demonstrated that XY-Deblur outperforms the conventional U-Net structure while consuming the same number of parameters. Also, we confirmed that the performance of the state-of-the-art networks can be further boosted by substituting their base structure with the proposed network.

6. Limitation

These are the possible limitations of the proposed image deblurring method.

1. Although XYDeblur maintains the number of parameters used in the network and improves the performance, the increase in the number of GMACs and the VRAM usage is inevitable. If the network is used in circumstances where the computational capacity is limited, overall adjustment of the network might be required.

2. To apply the proposed network, learning the residual image $\mathbf{r} = \mathbf{y} - \mathbf{x}$ is essential since the two decoders attempt to reconstruct the lost high-frequency components in a complementary manner. Without residual learning, the entire image including the lost high-frequency information should be estimated, which could overload the network and eventually hinder the deblurring performance.

3. Using the proposed network as a base network architecture can be constrained when the deblurring network has a complicated architecture. Since we construct the proposed network based on a very simple U-Net structure, if a deblurring network contains a complex decoder design, *e.g.*, transfer blocks in decoders, it can be necessary to further redesign or modify the architectures in the two decoders of the proposed network to implement the desired network.

Acknowledgement This work was supported by Institute of Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (2014-3-00077, Development of global multi-target tracking and event prediction techniques based on real-time large-scale video analysis). This work is supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2020R1A4A4079705)

References

[1] Dawei Dai, Liping Yu, and Hui Wei. Parameters sharing in residual neural networks. *Neural Processing Letters*, pages 1–18, 2019. 2

[2] Hongyun Gao, Xin Tao, Xiaoyong Shen, and Jiaya Jia. Dynamic scene deblurring with parameter selective sharing and nested skip connections. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3848–3856, 2019. 1, 2, 4, 6

[3] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep Learning*, volume 1. MIT Press Cambridge, 2016. 3

[4] Douglas M Hawkins. The problem of overfitting. *Journal of chemical information and computer sciences*, 44(1):1–12, 2004. 7

[5] Huaibo Huang, Ran He, Zhenan Sun, and Tieniu Tan. Wavelet-srnet: A wavelet-based cnn for multi-scale face super resolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1689–1697, 2017. 2

[6] Zhuo Hui, Ayan Chakrabarti, Kalyan Sunkavalli, and Aswin C Sankaranarayanan. Learning to separate multiple illuminants in a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3780–3789, 2019. 3

[7] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*, 2015. 4

[8] Dilip Krishnan, Terence Tay, and Rob Fergus. Blind deconvolution using a normalized sparsity measure. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 233–240, 2011. 1

[9] 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*, pages 8183–8192, 2018. 1

[10] Orest Kupyn, Tetiana Martyniuk, Junru Wu, and Zhangyang Wang. Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8878–8887, 2019. 1

[11] Wonkwang Lee, Donggyun Kim, Seunghoon Hong, and Honglak Lee. High-fidelity synthesis with disentangled representation. In *Proceedings of the European Conference on Computer Vision*, pages 157–174, 2020. 3

[12] Juncheng Li, Faming Fang, Kangfu Mei, and Guixu Zhang. Multi-scale residual network for image super-resolution. In *Proceedings of the European Conference on Computer Vision*, pages 517–532, 2018. 2

[13] Zhengqi Li and Noah Snavely. Learning intrinsic image decomposition from watching the world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9039–9048, 2018. 3

[14] Chang Liu, Wei Ke, Fei Qin, and Qixiang Ye. Linear span network for object skeleton detection. In *Proceedings of the European Conference on Computer Vision*, pages 133–148, 2018. 3

[15] Chang Liu, Yunjie Tian, Jianbin Jiao, and Qixiang Ye. Adaptive linear span network for object skeleton detection. *arXiv preprint arXiv:2011.03972*, 2020. 3, 5

[16] 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*, pages 3883–3891, 2017. 1, 2, 3, 4, 6

[17] Jordan Ott, Erik Linstead, Nicholas LaHaye, and Pierre Baldi. Learning in the machine: To share or not to share? *Neural Networks*, 126:235–249, 2020. 2

[18] 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*, pages 1628–1636, 2016. 1

[19] Vardan Papyan and Michael Elad. Multi-scale patch-based image restoration. *IEEE Transactions on Image Processing*, 25(1):249–261, 2015. 2

[20] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 4

[21] Surya N Patnaik, James D Guptill, and Dale A Hopkins. Subproblem optimization with regression and neural network approximators. *Computer Methods in Applied Mechanics and Engineering*, 194(30-33):3359–3373, 2005. 1

[22] John G Proakis. *Digital Signal Processing: Principles Algorithms and Applications*. Pearson Education India, 2001. 5

[23] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5389–5400. PMLR, 09–15 Jun 2019. 7

[24] Jaesung Rim, Haeyun Lee, Jucheol Won, and Sunghyun Cho. Real-world blur dataset for learning and benchmarking deblurring algorithms. In *Proceedings of the European Conference on Computer Vision*, pages 184–201, 2020. 4

[25] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241, 2015. 1

[26] Min-cheol Sagong, Yong-goo Shin, Seung-wook Kim, Seung Park, and Sung-jea Ko. Pepsi: Fast image inpainting with parallel decoding network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11360–11368, 2019. 3

[27] Maitreya Suin, Kuldeep Purohit, and AN Rajagopalan. Spatially-attentive patch-hierarchical network for adaptive motion deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3606–3615, 2020. 1, 2

[28] Libin Sun, Sunghyun Cho, Jue Wang, and James Hays. Edge-based blur kernel estimation using patch priors. In *Pro-*

ceedings of the IEEE International Conference on Computational Photography, pages 1–8, 2013. 1

[29] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8174–8182, 2018. 1, 2

[30] Oliver Whyte, Josef Sivic, and Andrew Zisserman. Deblurring shaken and partially saturated images. *International Journal of Computer Vision*, 110(2):185–201, 2014. 1

[31] Li Xu, Jimmy S Ren, Ce Liu, and Jiaya Jia. Deep convolutional neural network for image deconvolution. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 1790–1798, 2014. 1

[32] 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*, pages 1107–1114, 2013. 1

[33] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 1, 2

[34] Hongguang Zhang, Yuchao Dai, Hongdong Li, and Piotr Koniusz. Deep stacked hierarchical multi-patch network for image deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5978–5986, 2019. 1, 2, 4, 6

[35] Haichao Zhang and Jianchao Yang. Scale adaptive blind deblurring. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 3005–3013, 2014. 2