GAN-based image deblurring using DCT loss with customized datasets

HIROKI TOMOSADA¹(Student Member, IEEE), TAKAHIRO KUDO², TAKANORI FUJISAWA³, AND MASAAKI IKEHARA⁴(Senior Member, IEEE)
Department of Electronics and Electrical Engineering, Keio University, Yokohama 223-8522, Japan
Corresponding author: Hiroki Tomosada, Takahiro Kudo, Masaaki Ikehara(e-mail: tomosada,kudo,ikehara@tkhm.elec.keio.ac.jp)

ABSTRACT In this paper, we propose a high quality image deblurring method that uses discrete cosine transform (DCT) and requires less computational complexity. We train our model on a new dataset which is customized to include images with large motion blurs. Recently, Convolutional Neural Network (CNN) and Generative Adversarial Network (GAN) based algorithms have been proposed for image deblurring. Moreover, multi-scale and multi-patch architectures of CNN restore blurred images clearly and suppress more ringing or blocking artifacts, but they take a longer time to process. To improve the quality of deblurred images and reduce the computational time, we propose a method called “DeblurDCTGAN” that preserves texture and suppresses ringing artifacts in the restored image without multi-scale or multi-patch architecture using DCT based loss. This loss compares the restored image and the ground truth image in the frequency domain. With this loss, DeblurDCTGAN can reduce block noise and ringing artifacts while maintaining deblurring performance. Our experimental results show that DeblurDCTGAN gets the highest performances in terms of PSNR, SSIM, and running time compared with conventional methods. In terms of real image datasets, DeblurDCTGAN shows a better performance by using a customized training dataset made from GoPro, DVD, NFS and HIDE training datasets. Experimented code with pre-trained weights, datasets and results are available at https://github.com/Hiroki-Tomosada/DCTGAN-master

INDEX TERMS Image deblurring, Blind deconvolution, Non-Uniform, GAN, DCT(Discrete Cosine Transform).

I. INTRODUCTION
There are many chances to take photography by smartphones and cameras like GoPro. We are sometimes disappointed that the taking photos are out of focus or include motion blur caused by the motion of the camera and the subject. On security cameras, the motion blur is the apparent streaking of moving objects in a photograph or a sequence of frames. It results when the image being recorded changes during the recording of a single exposure, due to rapid movement or long exposure. Image deblurring to restore a sharp latent image from the degraded image has been an important topic in Computer Vision and Artificial Intelligence because the number of imaging devices has increased and HD displays have become widespread. The demand for image deblurring methods with high precision is increasing more and more in the past several years [1]. Image deblurring problems consists of two kinds of problem settings, which are the estimation of blur kernel and the deconvolution approach. Furthermore, Image deconvolution is devided into two types of approaches: non-blind and blind. Non-blind deconvolution is a classic inverse problem which assumes that the blur kernel is known and still hard to solve without some image priors. In contrast, blind deconvolution assumes that the kernel is unknown. For this reason, the blind deconvolution problem is a strongly ill-posed problem and very difficult to solve that because of little information on both the image and the kernel. A large number of single image blind deblurring methods have been proposed, which are divided into two categories, i.e., optimization-based and learning-based methods. Although optimization-based methods [2] [3] are very interesting, they need complex iterative calculation with some image and blur kernel priors. Then we focus on the recent learning-based methods.

A large number of CNN-based algorithms have been proposed for image restoration. CNN has been used for image deconvolution because it can flexibly deal with complex
changes in images. Similarly, there are some methods for image deblurring. However, the number of end-to-end methods for motion deblurring is still less because the problem setting is very difficult.

In the CNN-based methods, Generative Adversarial Network (GAN) based methods have been proposed in the field of image processing, such as SRGAN [4], CycleGAN [5], CDcGAN [6], and so on. GAN-based methods are effective to generate more realistic images than general CNN based methods. For the image deblurring task, Kupyn et al. [7] have proposed DeblurGAN which is the most famous GAN based method and extended to DeblurGAN-version2 (DeblurGANv2) [8]. These methods achieve state-of-the-art performance in objective and subjective evaluations by using the characteristics of GAN well. However, these methods often destroy the details of the deblurred image, and cause ringing or blocking artifacts.

Xin et al [9] have proposed SRN(Scale-Recurrent Network) architecture which achieves extremely high deblurring performance among end-to-end CNN-based methods. This network uses a multi-scale architecture with an LSTM (Long short-term memory) which is a type of RNN(Recurrent Neural Network) architecture. Other multi-scale structures have also been proposed recently [10] [11]. Although these methods exhibit high performance on image deblurring, it takes a long time to process because of multi-scale structure. To this end, we propose a method that preserves texture and suppresses ringing artifacts in the restored image without a multi-scale structure.

In this paper, we apply the GAN architecture to retain the details in the restored image. To train our model, we propose to use a DCT loss which was first introduced in the GAN-based method for image super-resolution [12]. This loss uses the discrete cosine transform (DCT) to compare the frequency components of the ground truth and restored images. The DCT loss is expected to suppress the extra frequency components created by the generator by comparing the frequency components of the images directly. In this paper, we introduce DCT loss in the generator instead of the discriminator and call DCT generator loss. Thus, DeblurDCTGAN can reduce ringing and blocking artifacts while maintaining deblurring performance. Also, we use a customized dataset for training in order to remove long blurs in the real images. This paper is the extended version of [13] to improve the deblurring performance using a customized dataset.

Both objective and subjective results in the experiments using some datasets show that DeblurDCTGAN can retain the details of the restored image while suppressing ringing artifacts and excessive patterns. Furthermore, our proposed method has the shortest execution time.

Our contributions are summarized in three aspects.

- We use GAN with an encoder-decoder structure to speed up the deblurring.
- We introduce DCT generator loss in the generator to reduce the ringing and blocking artifacts while keeping the details in the deblurred image.
- We customize the datasets to restore the real images with long blurs.

The remainder of this paper is organized as follows. First, Sec.II introduces the background of deblurring and related works. Sec.III explains the details of DeblurDCTGAN. Next, we compare the results of the proposed method with conventional methods for objective and subjective experiments in Sec.IV. Finally, Sec.V presents our conclusion.

II. BACKGROUND AND RELATED WORKS
A. NON-BLIND AND BLIND DECONVOLUTION

2D convolution in image processing is defined by the following equation

$$y = x \otimes k + n$$  \hspace{1cm}(1)$$

where $y$ and $x$ are the blurred and original latent images, respectively. $k$ and $n$ are the blur kernel and additive noise, respectively. Image deconvolution is classified into two types of approaches: non-blind and blind. Non-blind deconvolution is to recover a sharp latent image $x$ from a blurred image $y$ and a given blur estimate $k$, which is a classic image restoration problem. In contrast, blind deconvolution assumes that the kernel $k$ is unknown.

For the non-blind deconvolution problem, many approaches have been proposed [19]–[22]. On the other hand, it is very difficult to solve the blind deconvolution problem because of little information about the original image $x$ and the kernel $k$. In general, blind deconvolution aims to recover the blur kernel as well as the clean image. Since the dimension of the kernel is much smaller than the image size, one can better constrain the estimation of the blur kernel rather than the image. Hence, most existing blind deconvolution approaches [2] [3] try to recover an accurate motion estimate, recover the sharp image from the estimated motion kernel, and solve these problems alternatively.
B. GAN-BASED DEBLURRING

Generative Adversarial Network (GAN) framework is first proposed by Goodfellow et al. [23] and widely-used deep network architecture, which includes two subnetworks, pitting one against the other (that is, “adversarial”). In the CNN-based methods, Generative Adversarial Network (GAN) based methods have been proposed to generate more realistic images in image processing than general CNN-based methods.

Generally, the learning of GAN is performed using Adversarial Loss which uses the Binary Cross Entropy loss. Adversarial loss can be described as

\[ L_{adv} = \min_{\Theta_G} \max_{\Theta_D} \mathbb{E}[\log(D(x))] + \mathbb{E}[\log(1 - D(G(y)))] \]  

where \( \Theta_G \) and \( \Theta_D \) are the weights and biases of the Generator and the Discriminator. The Generator updates the weights and biases to obtain an output signal \( G(y) \) that cannot be discriminated by the Discriminator. In image processing, Content loss is added to the Adversarial loss similar to SRGAN [4]. Content loss calculates the difference between the output feature map of \( G(y) \) and \( x \) which are determined through VGG16 or VGG19. By adding this loss, the output images will have more subjective characteristics.

For the deblurring task, Kupyn et al. have proposed DeblurGAN [7] which is the most famous GAN-based method and extended to DeblurGAN-version2 (DeblurGANv2) [8]. These methods achieve state-of-the-art performance in objective and subjective evaluations. DeblurGANv2 shows better results than DeblurGAN because of double scale discriminator and Feature Pyramid Network for a wide range of backbones.

C. CNN-BASED DEBLURRING USING MULTI-SCALE ARCHITECTURE

Recently, CNN-based algorithms have been applied for image processing tasks since CNN can deal with complex changes of the images. Xin et al. [9] has proposed Scale-recurrent Network (SRN) as one of the highest performance methods in the deblurring task. SRN achieves extremely high deblurring performance by combining a multi-scale structure and a Convolutional LSTM layer [26] and adopts a symmetric CNN Encoder-Decoder structure of network [27].

D. OTHER HIGH PERFORMANCE CNN-BASED METHODS IN DEBLURRING

Lately, there has been very deep CNNs which can strongly restore the images. Multi-level CNN model called Deep Multi-Patch Hierarchical Network (DMPHN) [10] [28] [29]...
uses multi-patch hierarchy as input. In Zhang’s method, Stacked Multi-Patch Network called Stack(4)-DMPHN re-
stores blurred images most precisely. Also, Scale-Iterative Upscaling Network (SIUN) [30] recovers images in an
iterative manner. This network consists of a simple encoder-
decoder structure and a Super-Resolution Network (SR). The
encoder-decoder structure is a regular type of U-Net structure
[31]. Super-Resolution Network uses residual dense network
(RDN) combined with a U-Net. In the SR, the input image is
up-scaled and the output of the network is passed to encoder-
dercoder network. By iteratively repeating the above process,
SIUN can generate a sharp image. These methods can generate
sharp images from images with large blurs but the execution
time is extremely long because they use a very deep CNN
architecture.

| Validation Dataset | Train Dataset |
|--------------------|---------------|
|                    | GoPro PSNR    | DVD PSNR | NFS PSNR | HIDE PSNR | Custom PSNR |
| GoPro              | 30.08         | 28.52    | 27.96    | 30.16      | 30.51        |
| SSIM               | 0.9373        | 0.9077   | 0.8987   | 0.9399     | 0.9434       |
| DVD                | 27.78         | 31.25    | 29.69    | 27.79      | 30.27        |
| SSIM               | 0.8781        | 0.9346   | 0.9320   | 0.9829     | 0.9724       |
| NFS                | 31.72         | 34.08    | 34.03    | 31.41      | 34.04        |
| SSIM               | 0.9492        | 0.9631   | 0.9668   | 0.9497     | 0.9642       |
| HIDE               | 27.59         | 26.89    | 26.07    | 28.81      | 29.02        |
| SSIM               | 0.9131        | 0.8890   | 0.8775   | 0.9320     | 0.9342       |

III. PROPOSED METHOD
A. PROBLEMS OF CONVENTIONAL METHODS
Conventional methods such as SRN [9], DMPHN [10] and
SIUN [30], achieve high deblurring performance on test
images. However, there are still the following problems.

- Although multi-scale processing in SRN and multi-patch
  processing in DMPHN can remove large image blur, they
  often lose the details in the deblurred images.
- Multi-scale and multi-patch architectures take a long
time to deblur.
- Ringing or blocking artifacts might remain in the
deblurred image when the strong blur is removed.
- Large motion blur in real images can not be removed.

To overcome these problems, we propose a method named
"DeblurDCTGAN" that preserves texture and suppresses
ringing and blocking artifacts without using a multi-scale
and multi-patch networks.

B. OVERVIEW OF THE PROPOSED METHOD
Fig.1 shows the network structure of the proposed Deblur-
DCTGAN, which apply the architecture of GAN because it
can keep the details in the deblurred image. Also, we use a
customized dataset to include images with large motion blurs
to improve the performance.

1) Generator
Fig.2 shows the CNN architecture of the Generator and
Discriminator. We utilize an encoder-decoder structure which
is proven to be effective for the deblurring task and has been
used in SRN, DeblurGAN, and DeblurGANv2. However, we
do not adopt multi-scale scheme which is used in conventional
methods to speed up the computation. The proposed network
consists of InBlock, Encoder Block (EBlock), Residual Block
(ResBlock), Decoder Block (DBlock) and OutBlock. In this
network, the first and last convolution layers use $7 \times 7$ kernels,
and other convolution layers are $3 \times 3$. Instead of ReLU
activation, Parametric ReLU (PReLU) [32] is employed for
updating the optimal parameters in the training to prevent
overfitting except for EBlock#1. Only in the EBlock#1, we
use Leaky ReLU because it generate highest performances
of PSNR and SSIM. The stride for convolution layer in EBlock
is 2. As shown in Fig.2, EBlock and DBlock include batch
normalization in order to easily train and DBlock is symmetric
to EBlock. The number of ResBlock is 11. Each ResBlock is
adopted from the original ResNet [33] (But the activation
function is PReLU). Similarly to the DeblurGAN, Tanh is
used to normalize the final output.

2) Discriminator
Fig.2 includes the architecture of the discriminator. In this
architecture, a dense layer is a fully connected layer, and the
number of the Convblocks is 5.

3) Usage of DCT
To train the network, we transform the image into frequency
domain and use it as one of the losses. This approach refers
to the GAN-based method for image super resolution which
is proposed by Lee [12]. Both the deblurred image from the
generator $G(y)$ and the sharp latent image $x$ are converted
to grayscale images and then transformed into the frequency
domain with the discrete cosine transform (DCT). For the
dCT generator loss $L_{DCT}$, we use Least Absolute Difference
between the absolute values of these transformed images as

$$L_{DCT} = ||DCT(G(y))|| - ||DCT(x)||$$  \hspace{1cm} (3)\

where DCT($G(y)$), DCT($x$) is the frequency representation of
our network output $G(y)$ and the sharp latent image $x$
transformed by DCT, respectively. Note that DCT domain
generator is used instead of DCT domain discriminator used in
[12]. We tried faithfully to reproduce Lee’s method which uses
a discriminator to distinguish the frequency domain of the
ground truth image from that of the output image. However,
we found that the comparison between these images in the
frequency domain with Least Absolute Deviations is simpler
and produces better performances in our network than DCT
domain discriminator. By minimizing the difference of the
DCT component between the ground truth and deblurred
images, the DCT components of the blurred image are
approximated to that of the ground truth image. Therefore
the extra DCT components caused by block noise and ringing
artifacts in the deblurred image are reduced because these
artifacts include the specific frequency components. As a
result, the block noise and ringing artifacts can be reduced
TABLE 2. PSNR and SSIM comparison of using different loss functions

| Loss                   | N    | D    | A    | V    | DA   | AV   | DV   | Proposed |
|------------------------|------|------|------|------|------|------|------|----------|
| DCT generator loss     | ×    | ×    | ✓    | ✓    | ×    | ✔    | ×    |          |
| Adversarial loss       | ×    | ×    | ✓    | ×    | ✔    | ×    | ×    |          |
| VGG loss               | ×    | ×    | ×    | ×    | ✔    | ×    | ×    |          |
| PSNR                   | 30.22| 30.39| 30.24| 30.34| 30.44| 30.35| 30.33| 30.50    |

(a) Without Adversarial loss  
(b) Without DCT loss  
(c) DeblurDCTGAN

FIGURE 4. Subjective comparison of using different loss functions

TABLE 3. Averaged measurement results of GoPro, DVD, NFS and HIDE test datasets

| Method                | Time         | GoPro | DVD | NFS | HIDE |
|-----------------------|--------------|-------|-----|-----|------|
|                       |              | PSNR  | SSIM| PSNR| SSIM |
| Blurred               | -            | 25.64 | 0.8580 | 26.97 | 0.8462 | 32.09 | 0.9426 | 23.95 | 0.8299 |
| SRN [9]               | 0.452216s    | 30.25 | 0.9397 | 29.37 | 0.9110 | 32.58 | 0.9589 | 28.36 | 0.9208 |
| DeblurGAN [7]         | 0.169470s    | 25.02 | 0.8493 | 25.31 | 0.8368 | 28.20 | 0.9254 | 23.56 | 0.8181 |
| DeblurGANv2 [8]       | 0.113862s    | 28.00 | 0.9051 | 28.68 | 0.8871 | 32.55 | 0.9484 | 26.61 | 0.8899 |
| DMPHN(1-2-4-8) [10]  | 0.048710s    | 30.45 | 0.9416 | 27.51 | 0.8687 | 30.96 | 0.9481 | 27.79 | 0.9150 |
| Stack(4)+DMPHN [10]  | 1.993762s    | 31.39 | 0.9521 | 28.23 | 0.8773 | 31.99 | 0.9481 | 29.10 | 0.9331 |
| SIUN [30]             | 2.257751s    | 30.22 | 0.9404 | 29.84 | 0.9169 | 32.67 | 0.9616 | 28.42 | 0.9228 |
| DeblurDCTGAN          | 0.008548s    | 30.75 | 0.9467 | 30.45 | 0.9245 | 34.15 | 0.9644 | 29.20 | 0.9366 |

using DCT loss rather than the pixel-wise loss in the spatial domain.

C. LOSSES

1) Pre-training of the Generator

First, the generator of GAN is trained in advance to prevent adversarial training from failing to learn as used in SRGAN [4]. The pre-training loss is Mean Squared Error defined as

\[ L_{MSE} = \| x - G(y) \|^2 \]  \hspace{1cm} (4)

2) Training of The DCTGAN

To restore the details of the image, we use Absolute Difference between the output of the generator and the ground truth image. This is written by

\[ L_{L1} = \| x - G(y) \| \]  \hspace{1cm} (5)

Finally, the loss function for training in DeblurDCTGAN is defined as

\[ L_{total} = \alpha L_{L1} + \beta L_{cont} + \gamma L_{adv} + \delta L_{DCT} \]  \hspace{1cm} (6)
FIGURE 5. Visual comparison on GoPro Dataset (first two column), DVD Dataset (third column), NFS Dataset (fourth column) and HIDE Dataset (last columns). From top to bottom are blurry input, deblurred results of SRN, DeblurGAN, DeblurGANv2, Stack(4)-DMPHN, SIUN and our results.
Therefore, we analyze four datasets (GoPro, DVD, NFS and HIDE) in detail. We compare deblurred results on the real image by our model trained with different datasets. Fig. 3 shows the results deblurred by four methods which is trained by GoPro, DVD, NFS and HIDE training datasets. We use the same conditions to train each of our models. Fig. 3 indicates that the result using DVD dataset is the worst. In addition, numerical results of using different training datasets are shown in Tab. 1. Each column represents the dataset used for training and each row represents the dataset used for testing. Tab. 1 shows that the results of DCTGAN is dependent on the training dataset.

2) Training kernel estimation network

In order to analyse the size of the motion kernel, we use a kernel estimation neural network proposed by Lu et al. [35]. We train the network shown in Fig. 6. This network can estimate the size of the motion kernel by using a training dataset with synthetically blurred images from the MSCOCO [36] dataset and its motion kernel size. Our kernel estimation network is different from Lu et al. in the output of the network. Our network does not estimate the orientation of the motion kernel and only estimates the size of the motion kernel in the image.

3) Estimate size of motion kernel in training dataset

We estimated the size of motion kernel in GoPro, DVD, NFS and HIDE training datasets. Fig. 7 shows the histograms of the motion kernel sizes in each training dataset. In the histogram of the DVD training dataset, the number of large kernel sizes is particularly low and the distribution of small kernel size region is larger than the others.

4) Creating a new training dataset

The reason why using different training datasets results in varying results is that each dataset has a different distribution of kernel sizes. In particular, the blur left in Fig. 3(a) is due to that the DVD training dataset includes many small kernel size images and the distribution is biased towards small kernel sizes. Therefore, a customized dataset is created by removing images such that the distribution of the histogram becomes a normal distribution and remaining many images with large blurs. Fig. 7 (e) shows the histogram of motion kernel size in the customized dataset. The peak number of kernel sizes is larger than previous datasets and the number of images with small kernel size is less. The proposed method uses this customized dataset for training.

D. TRAINING DATASETS

We use 2103 pairs of the GoPro Datasets [14], 8849 pairs of the DVD Datasets [15], 8500 pairs of the NFS Datasets [16] and 6397 pairs of the HIDE Datasets [17] to train the generator because we find that the results on the real image dataset is greatly affected by the training dataset.

1) Effect of using different training datasets

The results of DCTGAN are affected by the training dataset. Therefore, we analyze four datasets (GoPro, DVD, NFS and HIDE dataset) in detail. We compare deblurred results on the real image by our model trained with different datasets. Fig. 3 shows the results deblurred by four methods which is trained by GoPro, DVD, NFS and HIDE training datasets. We use the same conditions to train each of our models. Fig. 3 indicates that the result using DVD dataset is the worst. In addition, numerical results of using different training datasets are shown in Tab. 1. Each column represents the dataset used for training and each row represents the dataset used for testing. Tab. 1 shows that the results of DCTGAN is dependent on the training dataset.

2) Training kernel estimation network

In order to analyse the size of the motion kernel, we use a kernel estimation neural network proposed by Lu et al. [35]. We train the network shown in Fig. 6. This network can estimate the size of the motion kernel by using a training dataset with synthetically blurred images from the MSCOCO [36] dataset and its motion kernel size. Our kernel estimation network is different from Lu et al. in the output of the network. Our network does not estimate the orientation of the motion kernel and only estimates the size of the motion kernel in the image.

3) Estimate size of motion kernel in training dataset

We estimated the size of motion kernel in GoPro, DVD, NFS and HIDE training datasets. Fig. 7 shows the histograms of the motion kernel sizes in each training dataset. In the histogram of the DVD training dataset, the number of large kernel sizes is particularly low and the distribution of small kernel size region is larger than the others.

4) Creating a new training dataset

The reason why using different training datasets results in varying results is that each dataset has a different distribution of kernel sizes. In particular, the blur left in Fig. 3(a) is due to that the DVD training dataset includes many small kernel size images and the distribution is biased towards small kernel sizes. Therefore, a customized dataset is created by removing images such that the distribution of the histogram becomes a normal distribution and remaining many images with large blurs. Fig. 7 (e) shows the histogram of motion kernel size in the customized dataset. The peak number of kernel sizes is larger than previous datasets and the number of images with small kernel size is less. The proposed method uses this customized dataset for training.
FIGURE 8. Subjective comparison on the Real image dataset. From top to bottom are blurry input, deblurred results of SRN, DeblurGANv2, Stack(4)-DMHPN, SIUN and our results.
total number of epochs is 400, which is executed after pre-training the generator for 10 epochs. It takes about 4 days to train the network on a single GeForce GTX 1080 Ti.

IV. EXPERIMENT RESULTS

(Experimented code with pre-trained weights, datasets and results are available at https://github.com/Hiroki- Tomosada/DCTGAN-master)

We compare the proposed method with other methods, including SRN [9], DeblurGAN [7], DeblurGANv2 [8], DMPHN [10] and SIUN [30] in this section. In addition, an ablation study of the loss function in DeblurDCTGAN is also done to validate the DCT generator loss, adversarial loss and content loss. We use PSNR and SSIM for the numerical comparison with Python, which implements standard calculations. Also, we use 1111 test image pairs of GoPro dataset, 1500 test images of DVD dataset, 1500 test images of NFS dataset, 2025 test images of HIDE dataset and Real image dataset [18] for the experiment. Real image dataset is used for the subjective results to verify the effectiveness of DeblurDCTGAN.

A. NUMERICAL RESULTS

Tab.3 shows the average numerical results of GoPro, DVD, NFS and HIDE validation datasets. In this Table, the proposed method achieves the highest performances on both PSNR and SSIM compared with other conventional methods except for the GoPro dataset. The reason why the results of PSNR and SSIM on the GoPro dataset is lower than that of Stack(4)-DMPHN is that DMPHN is trained with the GoPro training dataset. Furthermore, the computational time of the proposed method is extremely faster than the conventional methods in these datasets(The execution time does not include Disc I/O of images). Therefore, the proposed method is superior to other conventional methods in terms of numerical results.

B. SUBJECTIVE RESULTS

In this section, we consider the results and explain the effectiveness of each loss. Fig.4 shows the effect of the DCT generator loss and Adversarial loss. Fig.5 and 8 show the subjective results of GoPro, DVD, NFS, HIDE datasets and Real dataset, respectively.

1) Results of GoPro, DVD, NFS and HIDE datasets

In this section, we compare DCTGAN with other methods. Fig.5 shows subjective results on four validation datasets. The first two columns in the figure are GoPro validation images and the others columns show DVD, NFS and HIDE validation images in order. In GoPro images, the proposed method is better able to reduce the ringing artifacts and restore the texture in the output image except for Stack(4)-DMPHN. This can be confirmed with the car’s license plate in the images. For the third and fourth columns, the conventional methods cannot remove the strong blur on the tree and bicycle. However, the proposed method can remove the strong blur and restore the shape of the tree and bicycle. For the image in the fourth column, our method can remove the strong blur.

2) Results of Real image dataset

Fig.8 shows the results of Real image dataset for visual comparison to conventional methods. As for the images in the first and last columns, each deblurred images generated by conventional methods include ringing artifacts and block noise or fail to remove the blur. In contrast, our method suppresses the ringing artifacts and block noise on the characters in the image. The second results of SRN, DMPHN and SIUN lose these texture and the images in the fourth column of SRN and DMPHN look unnatural. In contrast, the restored image from our DeblurDCTGAN reduces ringing artifacts and blur.

C. ABLATION STUDY

1) Effect of each loss

In order to validate DCT generator loss, adversarial loss and content loss, we trained DeblurDCTGAN without each of the losses using the GoPro dataset under the same conditions and evaluate the deblurred images objectively and subjectively. Tab.2 shows the ablation study in the proposed DeblurDCTGAN. We can see from this table that the proposed method with all the losses achieves the highest PSNR. In particular, the effect of DCT generator loss is larger under any circumstances. In the first and third images of Fig.4, the proposed method reduces the ringing artifacts of the face in Fig.4b and the car’s license plate in Fig.4a compared to Fig.4c. Also, the second row in Fig.4 shows that Fig.4c removes large blur more accurately than the others. By using all of our losses, we can effectively train our model for image deblurring.

2) Effect of the customized dataset

As shown in Tab.1, using a customized dataset for training can raise the PSNR and SSIM. It can be said that the customized training dataset not only enables DCTGAN to deblur generic images, but also makes it easier to deblur images with large motion kernels.

V. CONCLUSION

In this paper, we proposed DeblurDCTGAN by introducing a GAN. The proposed method can retain the restored details and reduce block noise and ringing artifacts by using the DCT generator loss. In particular, the DCT generator loss is expected to suppress the extra frequency component created by the generator. Also, the proposed method can remove large motion blurs in any validation dataset. Both objective and subjective results show that the proposed method has high deblurring performance and suppresses excessive patterns using the customized dataset. Furthermore, the proposed method can dramatically reduce the computational time compared to the conventional method.
TAKAHIRO KUDO received the B.E. and M.E. degrees in electrical engineering from Keio University, Yokohama, Japan, in 2018 and 2020, where he is currently pursuing the M.E. degree, under the supervision of Prof. M. Ikehara. His research interests include image deblurring, demoiréing and image processing using deep learning.

TAKANORI FUJISAWA received the B.E., M.E. and PhD degrees in electrical engineering from Keio University in 2014, 2016, and 2019 respectively. His research interests are in the field of applications of sparse coding to image super resolution, image inpainting, and automatic music transcription.

MASAAKI IKEHARA received the B.E., M.E. and PhD degrees in electrical engineering from Keio University in 1984, 1986, and 1989, respectively. He is currently a Full Professor with the Department of Electronics and Electrical Engineering, Keio University. His research interests are in the areas of multi-rate signal processing, wavelet image coding, and filter design problems.

***