Deblurring by Realistic Blurring

Kaihao Zhang[1] Wenhan Luo[2] Yiran Zhong[1,4] Lin Ma[2] Bjrn Stenger[3] Wei Liu[2] Hongdong Li[1,4]
[1] Australian National University [2] Tencent AI Lab [3] Rakuten Institute of Technology [4] ACRV

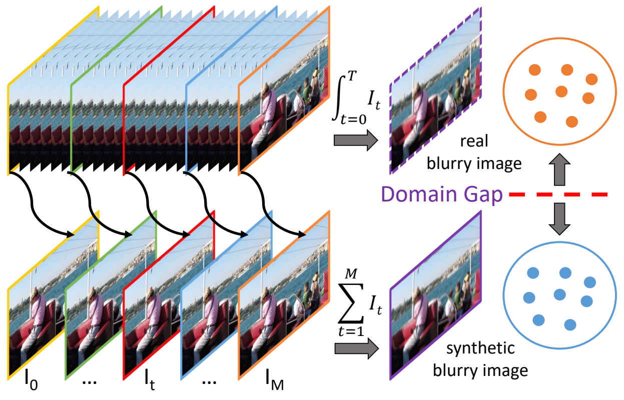
Abstract

Existing deep learning methods for image deblurring typically train models using pairs of sharp images and their blurred counterparts. However, synthetically blurring images do not necessarily model the genuine blurring process in real-world scenarios with sufficient accuracy. To address this problem, we propose a new method which combines two GAN models, i.e., a learning-to-Blur GAN (BGAN) and learning-to-DeBlur GAN (DBGAN), in order to learn a better model for image deblurring by primarily learning how to blur images. The first model, BGAN, learns how to blur sharp images with unpaired sharp and blurry image sets, and then guides the second model, DBGAN, to learn how to correctly deblur such images. In order to reduce the discrepancy between real blur and synthesized blur, a relativistic blur loss is leveraged. As an additional contribution, this paper also introduces a Real-World Blurred Image (RWBI) dataset including diverse blurry images. Our experiments show that the proposed method achieves consistently superior quantitative performance as well as higher perceptual quality on both the newly proposed dataset and the public GOPRO dataset.

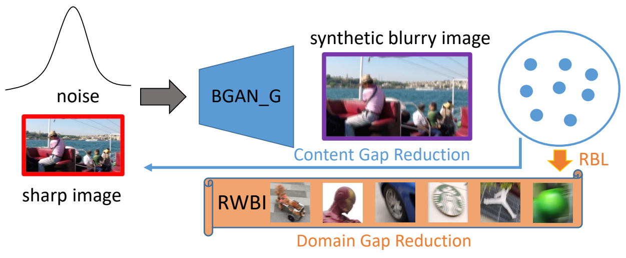
1. Introduction

Image deblurring is a classic problem in low-level computer vision, and it remains an active topic in the vision research community. Given a blurred image, which is corrupted by some unknown blur kernel or a spatially variant kernel, the task of (blind) image deblurring is to recover the sharp version of the original image, by reducing or removing the undesirable blur in the blurred image. Traditional deblurring methods handle this problem via estimating a blur kernel, through which a sharp version of the blurred input image can be recovered. Often, special characteristics of the blur kernel are assumed, and natural image priors are exploited in the deblurring process [5, 7, 22, 40, 41]. However, estimating the optimal blur kernel is a difficult task and can therefore impair the overall performance.

Recently, deep learning methods, particularly convolutional neural networks (CNNs), have been applied to tackle this task and obtained a remarkable success, *e.g.*, [21, 33, 36, 42]. Existing deep learning methods focus on training deblurring models using *paired* blurry and sharp images. For example, Nah *et al.* [21] propose a multi-scale loss function to implement a coarse-to-fine processing pipeline. Tao *et al.* [36] and Gao *et al.* [6] improve the work by using shared network weights among different scales, achieving state-of-the-art performance.



(a)



(b)

Figure 1. (a) The differences between real and synthetic blurry images; (b) an illustration of learning to blur. Sharp images and random noises are fed into the BGAN_G model to generate realistic blurry images via the RBL loss and the RWBI dataset.

However, many common effects are not adequately captured by the current deep learning models in the following sense. First, since in real-world scenarios, an image is captured during a time window (*i.e.*, the exposure duration), the blurred image is in fact the integration of multi-frame instant and sharp snapshots [10]. This can be formulated as

$$I_B = g\left(\frac{1}{T}\int_{t=0}^{T} I_{S(t)}\mathrm{d}t\right), \tag{1}$$

where I_S is an instant sharp frame and I_B is the blurry image. T is the exposure time period and $g(\cdot)$ is the Camera Response Function (CRF). In contrast, in conventional deblurring methods, blurry images used in the training set are often artificially synthesized by approximating the integration step with a simple averaging operation, as shown in Eq. (2), where M is the number of frames:

$$I_B \simeq g\left(\frac{1}{M}\sum_{t=1}^{M} I_{S[t]}\right). \tag{2}$$

Prior methods use M sharp frames $I_{S[t]}$ to replace the continuous sequence $I_{S(t)}$ and generate paired training data, avoiding the complexity of obtaining pairs of real blurry and sharp images. However, there is a clear gap between real blurry images and those artificially blurred images. Fig. 1(a) shows the generation of real and synthetic blurry images.

Second, in real situations there are multi-fold factors (not limited to a single linear integration or summation) which can cause image blurs, for instance, camera shake, fast object motion, and small aperture with a wide depth of field. Many of these factors are very difficult to model precisely. To design a better deblurring algorithm, all these factors should be taken into consideration. If the real blurred images are different from the samples in the training set, the trained model may not perform well on the testing data. This observation inspires us to develop a new deblurring method which does not assume any particular blur type; rather such a method will be able to learn a blurring process in order to achieve better deblurring quality.

Specifically, in this paper we propose a method which contains a leaning-to-Blur GAN (BGAN) module and a learning-to-DeBlur GAN (DBGAN) module. BGAN and DBGAN are two complementary processes, in the sense that BGAN learns to mimic properties of real-world blurs by generating photo-realistic blurry images. This module is trained using unpaired sharp and blurry images, thus relaxing the requirement of needing paired data. Recently, Shaham *et al.* propose SinGAN [27] to produce different images based on random noises, which inspires us to generate various blurry images given different noises. During the generation, sharp images are also fed into BGAN to make the generated blurry images bear the same content as the input images. The DBGAN module learns to recover sharp images from blurry images with real sharp and generated blurry images. We further employ a relativistic blur loss, which helps predict the probability that a real blurry image is relatively more realistic than a synthesized one. Finally, a Real-World Blurry Image (RWBI) dataset is created to help train the BGAN model and evaluate the performance of our proposed image deblurring model. Fig. 1(b) shows the process of learning realistic blur.

The contributions of this paper are three-fold: (1) We develop a new image deblurring framework which contains the process of image blurring and image deblurring. In contrast to previous deep learning methods which solely focus on image deblurring, our framework also considers image blurring, which generates realistic blurry images to help enhance the performance and robustness of image deblurring. (2) In order to train the BGAN model and generate blurry images like those in the real world, a relativistic blur loss is introduced. We also contribute a real-world blurry dataset RWBI, which can be used for training an image blurring module and for evaluating deblurring models. (3) Experimental results show that the proposed method achieves not only the state-of-the-art quantitative performance on the public GOPRO benchmark, but also consistently superior perceptual quality on real-world blurry images.

2. Related Works

Our work in this paper is closely related to image blurring and image deblurring, which are briefly introduced as follows, respectively.

2.1. Image Blurring

Blur artifacts are caused by various factors. The blurring process can be mathematically formulated as [8, 38],

$$I_B = K * I_S + N, \tag{3}$$

where I_B and I_S are blurry and sharp images, respectively. K is the unknown (blind) or known (non-blind) blur kernel and N is additive noise. For images with spatially varying blurs there are no camera response function (CRF) estimation techniques [35]. Alternatively, the CRF can be approximated as the average of known CRFs as follows:

$$g(I_{S[i]}) = I_{S'[i]}{}^{\frac{1}{\gamma}}, \tag{4}$$

where γ is a parameter. The latent realistic sharp images $I_{S[i]}$ can be obtained based on the observed sharp images $I_{S'[i]}$. The blurry images can then be generated based on Eq. (2). Eq. (2) and Eq. (3) are the two main methods to generate image pairs for training. However, neither of them is able to synthesize realistic blurry images like Eq. (1).

2.2. Image Deblurring

Early works use image priors, including total variation [3], a heavy-tailed gradient prior [28], or a hyper-Laplacian prior [15], which are typically applied to images in a coarse-to-fine manner. Recently, deep learning methods

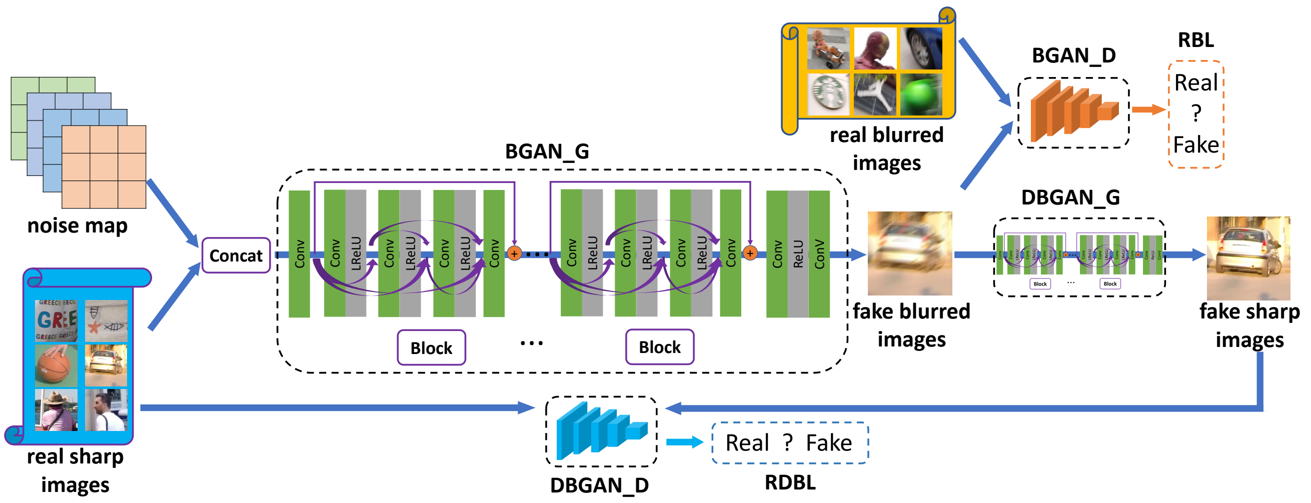


Figure 2. **The proposed framework and training process.** This framework contains two main modules, a BGAN and a DBGAN. D and G denote discriminator and generator networks, respectively. The BGAN takes sharp images as input and outputs realistic blurry images, which are then fed into the DBGAN in order to learn to deblur. During the inference stage, only the DBGAN is applied.

have achieved a great success in the areas of object recognition [9, 44, 43, 19, 18] and image reconstruction including video deblurring [45], video dehazing [25], and other GAN-based generation tasks [29, 24, 39, 37]. For image deblurring, Sun *et al.* [34] propose a CNN-based model to estimate a kernel and remove non-uniform motion blur. Chakrabarti [2] uses a network to compute estimations of sharp images that are blurred by an unknown motion kernel. Nah *et al.* [21] propose a multi-scale loss function to apply a coarse-to-fine strategy and an adversarial loss. Kupyn *et al.* propose DeblurGAN [16] and DeblurGAN-v2 [17] to remove blur kernels based on adversarial learning. Further, RNN-based methods have been proposed for image deblurring. Zhang *et al.* [42] propose a spatially variant neural network, which includes three CNNs and one RNN. Tao *et al.* [36] propose an SRN-DeblurNet, which includes one LSTM and CNNs for multi-scale image deblurring. Shen *et al.* [30] introduce a human-aware deblurring method to remove blur from foreground humans and background. Gao *et al.* [6] propose a nested skip connection structure which achieves state-of-the-art performance.

All these above neural network based methods focus on solely recovering sharp images from blurry images (*i.e.*, image deblurring), rather than better modeling the blurring process itself. Pan *et al.* [23] try to generate blurry images in their algorithm based on deblurred results, and then calculate the difference between the generated blurry images and "GT" blurry images to update models. Therefore, these methods actually propose a new loss function, rather than data augmentation. The idea of data augmentation has been widely applied in different fields [31], like face verification [20] and SR [1]. For deblurring, one of the most relevant works is from the field of video deblurring [4]. However, it generates blurred images based videos and it does not consider to generate realistic blurry images based on real blurred ones. More recently, a SinGAN [27] model is proposed to learn how to generate different related images from one input image based on random noises. Inspired by this method, a GAN-based model is proposed to generate various blurry images based on different noises.

3. Deblurring by Blurring

3.1. Overall Architecture

Our framework contains two primary modules. Similar to prior image deblurring works, our framework includes a learning-to-DeBlur GAN (DBGAN) module, which is trained on paired sharp and blurry images to recover sharp images from blurry images. The paired sharp-blurry images are obtained from the BGAN module. The BGAN is trained on unpaired data, where sharp images come from a public dataset, while the blurry images come from a new real-world blurry dataset. Fig. 2 shows the overall architecture of the proposed framework.

We further enhance the standard GAN model with a relativistic blur loss. In traditional GAN-based models for image deblurring, the discriminator D estimates the probability that the input data is real, and the generator G is trained to increase the probability that the generated data looks real. The developed relativistic blur loss estimates the probability that the given real-world blurry images are more realistic than the generated blurry images.

In the training stage, sharp images are input into the BGAN generator and its output is fed into the DBGAN to learn how to deblur. The generators in the DBGAN and BGAN modules generate corresponding images, and the discriminators conduct discrimination to create more realistic synthetic images. During the inference stage, only the DBGAN generator network is required for the image deblurring task.

3.2. BGAN: Learning to Blur

The BGAN module is the primary difference from other neural network based methods for image deblurring. Similar to other GAN based models, the BGAN consists of a generator network and a discriminator network. In this section, we first discuss its architecture and loss functions.

BGAN Generator. The input to the BGAN generator is a sharp image from a public dataset. Given the numerous possible factors that can cause undesired blurring artifacts, we concatenate the input image with a noise map to model the different conditions. To obtain the noise map, we sample a noise vector of length 4 from a normal distribution and duplicate it 128×128 times in the spatial dimension to obtain a $4 \times 128 \times 128$ noise map as in [46]. In this way, we can generate various blurry images based on one sharp image. The network architecture consists of one convolutional layer, 9 residual blocks (ResBlocks) [9] and extra two convolutional layers. Each ResBlock consists of 5 convolutional layers ($64 \times 3 \times 3$) and 4 ReLU activations. There is also a skip connection in each ResBlock, connecting the input and output features (refer to Fig. 2). The output of our BGAN generator is a blurry image of the same size as the sharp input image.

BGAN Discriminator. The input to the BGAN discriminator is the output of the BGAN generator. Its architecture is the same as the VGG19 network [32], and its output is the probability of the blurry image being classified as real.

BGAN Loss. The generator and discriminator of the BGAN are trained with a perceptual loss and an adversarial loss. Specifically, the perceptual loss is calculated based on the synthesized blurry images from the proposed BGAN and images taken from a public dataset. In this way, they can have similar contents. The adversarial loss is calculated between the synthesized and real blurry images. The real blurry images are taken from our newly created dataset.

3.3. DBGAN: Learning to Deblur

The BGAN module aims to mimic the real-world blurry images and cover as many blur cases as possible. Its goal is to drive the DBGAN module to be more effective in recovering sharp images from blurry images. In the following, we present the architecture and loss of the DBGAN.

DBGAN Generator. The input to the DBGAN generator is a blurry image. Many approaches have been proposed for this task [2, 21, 34, 36]. When we design the DBGAN generator, we adopt their advantages. Specifically, we remove the batch normalization layers, which have been shown to increase the computational complexity and decrease the performance on different tasks [21]. Secondly, we use additive residual layers in each block, which combine multi-level residual networks and dense connections [11]. The BGAN consists of one convolutional layer, 16 residual blocks (ResBlocks) [9] and two more convolutional layers. The kernel size in ResBlocks is $63 \times 3 \times 3$. The details can be referred to Fig. 2. The output of the DBGAN generator is the desired sharp image.

DBGAN Discriminator. Similar to the BGAN discriminator, the DBGAN also adopts the VGG19 network [32] as its discriminator. The output of this model is the probability of the given sharp images looking realistic.

DBGAN Loss. Like the BGAN module, the proposed DBGAN model is trained using a perceptual loss and an adversarial loss. We also use an L_1 loss to update the DBGAN. All the three types of loss functions are calculated based on the generated and real sharp images, so the DBGAN is trained on paired images.

3.4. Relativistic Blur Loss

In this section, we describe a Relativistic Blur Loss (RBL) and other loss functions which are used to train our framework.

Perceptual Loss. In contrast to previous image deblurring methods [21, 36], the proposed framework applies a perceptual loss $\mathcal{L}_{perceptual}$ to update models. Note that Johnson *et al.* [13] use a similar loss. However, in contrast to their work, we calculate the perceptual loss based on features before rather than after the ReLU activation layer.

Content Loss. The Mean Squared Error (MSE) is widely used as a loss function for image restoration methods. Based on the MSE, the content loss between ground-truth and generated images is calculated.

Relativistic Blur Loss. In order to drive the BGAN generator to produce blurry images similar to the real-world images, we develop a relativistic blur loss based on [14] to update the model. The BGAN generator parameters are updated in order to fool the BGAN discriminator. The adversarial loss D is formulated as:

$$D(I_{blurry}^{real}) = \sigma(C(I_{blurry}^{real})) \rightarrow 1,$$

$$D(I_{blurry}^{fake}) = D(G(I_{sharp}^{real})) = \sigma(C(G(I_{sharp}^{real}))) \rightarrow 0\,, \tag{5}$$

where $D(\cdot)$ is the probability that the input is a real image. $C(\cdot)$ is the feature representation before activation and $\sigma(\cdot)$ is the sigmoid function. The generator G is trained to increase the probability that synthesized images are real. Real and synthesized images are labeled as 0 or 1 by D, respectively. As Fig. 3 (a) shows, the effect of G is to transfer real sharp images to blurry images and "push" these generated images (label=0) closer to real blurry images (label=1). However, during the training stage, only the second part of Eq. (5), *i.e.*, $D(I_{blurry}^{fake}) = D(G(I_{sharp}^{real})) \rightarrow 0$, updates

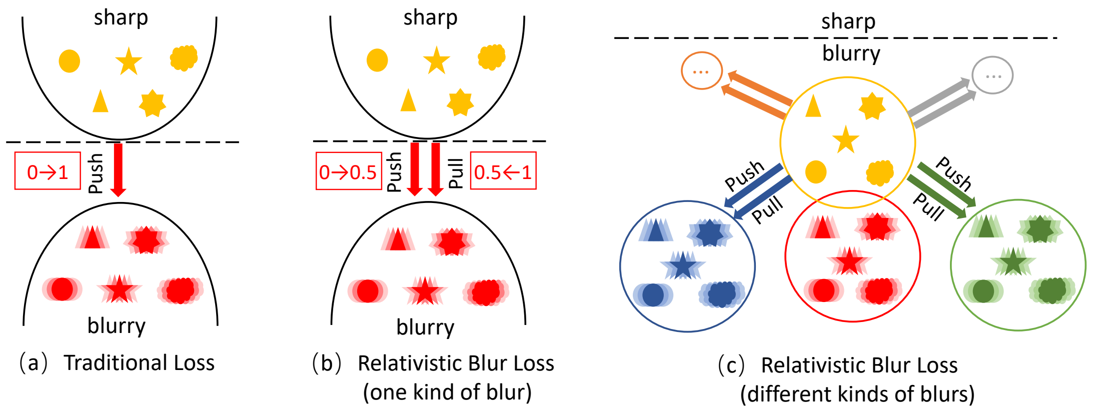

sharp
0→1
Push
blurry
(a) Traditional Loss
sharp
0→0.5
Push
Pull
0.5←1
blurry
(b) Relativistic Blur Loss (one kind of blur)
sharp
blurry
Push
Pull
Push
Pull
(c) Relativistic Blur Loss (different kinds of blurs)

Figure 3. **An illustration of the Relativistic Blur Loss (RBL).** Real and synthesized images are labeled as 1 and 0, respectively. (a) A traditional loss function is used to update the generator to create blurry images (label=0) which are similar to real ones (label=1). (b) The RBL not only increases the probability that generated images look real ($0 \to 0.5$, which is labeled as "Push"), but also simultaneously decreases the output probability that real images are real ($1 \to 0.5$, which is labeled as "Pull"). (c) In order to increase the variations of blurry images, different blurry images are used to model the different types of blurs in the real world.

the parameters of generator G, while the first part is used to update the discriminator D model rather than the generator G [21]. In fact, a powerful generator G should also decrease the probability that real blurry images are real. This is because a realistic synthesized image labeled as fake is similar to real one, and will thus fool the D model to learn to distinguish real and fake in the training stage. Based on this idea, we add $D(I^{real}_{blurry})$ into the process of learning G in BGAN. Specially, a Relativistic Blur Loss (RBL) is developed to help calculate whether a real blurry image is more realistic than the synthesized blurry image. The formulation of Eq. (5) is modified to

$$\begin{aligned} \sigma(C(I^{real}_{blurry}) - E(C(G(I^{input})))) \to 1\,, \\ \sigma(C(I^{fake}_{blurry}) - E(C(I^{real}_{blurry}))) \to 0\,, \end{aligned} \tag{6}$$

where $E(\cdot)$ denotes the averaging operation over images in one batch. Fig. 3 (b) shows the aim of RBL. Although the goal is still to generate realistic blurry images which are similar to real-world ones, the optimization objective is different. RBL aims to update G to generate synthetic images which are near 0.5, and meanwhile to fool the D model, making it difficult to distinguish real images from fake ones. In this way, the probability of real blurry images predicted by D is also near to 0.5. We term the effects as "push" and "pull", respectively, which can complement each other to update the generator G. As Fig. 3 shows, the sharp and blurry images can be regarded as two different domains. In order to rapidly generate blurry images and utilize prior research results of generating blurry images, we first train our BGAN model with artificially blurry images as Fig. 3(b) shows. We then add other types of blurry images to increase the variations of the produced blurry images based on Eq. (6) to cover different conditions in the real world, which is shown in Fig. 3(c).

Based on Eq. (6) and Fig. 3, our RBL, which is used in the BGAN generator, can be represented as

$$\begin{aligned} \mathcal{L}_{RBL} = -[&\log(\sigma(C(I^{real}_{blurry}) - E(C(G(I^{input}))))) \\ &+ \log(1 - (\sigma(C(G(I^{input})) - E(C(I^{real}_{blurry}))))]. \end{aligned} \tag{7}$$

Based on the RBL, we apply a Relativistic Deblur loss (RDBL) in the DBGAN generator as

$$\begin{aligned} \mathcal{L}_{RDBL} = -[&\log(\sigma(C(I^{real}_{sharp}) - E(C(G(I^{input}))))) \\ &+ \log(1 - (\sigma(G(I^{input})) - E(C(I^{real}_{sharp})))))]. \end{aligned} \tag{8}$$

Balance of Different Loss Functions. During the training stage, the loss functions for DBGAN and BGAN are combinations of different terms using a weighted fusion,

$$\mathcal{L}_{BGAN} = \mathcal{L}_{perceptual} + \beta \cdot \mathcal{L}_{RBL}, \tag{9}$$

$$\mathcal{L}_{DBGAN} = \mathcal{L}_{perceptual} + \alpha \cdot \mathcal{L}_{content} + \beta \cdot \mathcal{L}_{RDBL}\,. \tag{10}$$

In order to balance the different kinds of losses, we use two hyper-parameters α and β to yield the final loss $\mathcal{L}$ for BGAN and DBGAN.

4. Experiments

4.1. Datasets

GOPRO Dataset. We evaluate the performance of our model on the public GOPRO dataset [21], which contains $3,214$ image pairs. The training and testing sets include $2,103$ and $1,111$ pairs, respectively. Existing methods convolve sharp images with a blur kernel [2, 26, 34] to synthesized blurry images. These synthetic blurry images

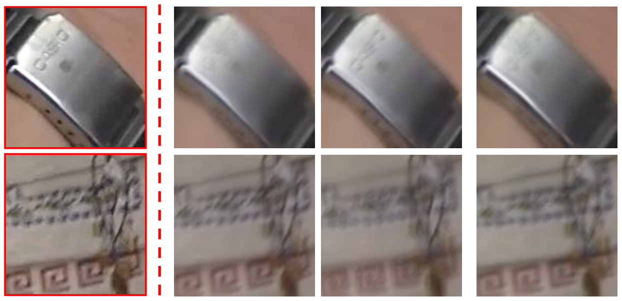


Figure 4. **Synthesized blurry images.** Examples of different blurry images created by the proposed BGAN. The first column shows input sharp images, and the next three columns are the produced blurred images used to train the DBGAN(+).

are different from real ones captured by camera. In order to model more realistic blurry conditions, in the GOPRO dataset, sharp images with a high-speed camera and synthesize blurry images were collected by averaging these sharp images from videos.

RWBI Dataset. In order to train our BGAN model and evaluate the performance of deblurring models, we collect a Real-World Blurry Image dataset. The blurry images are captured with different hand-held devices, including an iPhone XS, a Samsung S9 Plus, a Huawei P30 Pro and a GoPro Hero 5 Black. Multiple devices are used to reduce the bias towards one specific device which may capture blurry images with unique characteristics. The dataset contains 22 different sequences of $3,112$ diverse blurry images.

We compare the performance of the proposed method with the state-of-the-art methods on the public GOPRO dataset quantitatively and qualitatively. As there is no ground truth of the developed RWBI dataset, we only conduct a qualitative comparison.

4.2. Implementation Details

When training BGAN and DBGAN, we use a Gaussian distribution with zero mean and a standard deviation of 0.01 to initialize the weights. In each iteration, we update all the weights after learning a mini-batch of size 4. To augment the training set, we crop a 128×128 patch at any location of an image. To further increase the number of training samples, we also randomly flip frames. We use a learning rate annealing scheme, starting with a value of 10^{-4} and reducing it to 10^{-6} after the training loss gets converged. The hyper-parameters α and β are set as 0.005 and 0.01, respectively.

4.3. Ablation Study

In this section, we conduct experiments to investigate the effectiveness of different components of our model. The proposed model has three variants:

(1) **DBGAN** is the model for learning to deblur. Its input is a blurry image and the output is a deblurred image. Similar to previous GAN-based deblurring methods [21, 16], this model contains generator and discriminator networks. Thus its loss function is a combination of $\mathcal{L}_{percetpual}$, $\mathcal{L}_{content}$ and $\mathcal{L}_{RDBL}$ with weights α and β. The final loss function is shown in Eq. (10).

(2) **DBGAN(-)** has the same architecture as DBGAN. Differently, we replace the $\mathcal{L}_{RDBL}$ with a traditional adversarial loss as [21]. Namely, the training process does not contain the relativistic loss functions. It is trained based on $\mathcal{L}_{percetpual}$, $\mathcal{L}_{content}$ and the traditional adversarial loss.

(3) **DBGAN(+)** is our full method. It has a similar architecture to DBGAN with the main difference of additionally employing the BGAN module during the fine-tuning stage. Specially, we firstly train a DBGAN model as above, and then blurry images generated by the BGAN model are randomly added into the training samples to enhance the learning performance of DBGAN. Fig. 4 shows the examples of different blurry images produced by the proposed BGAN.

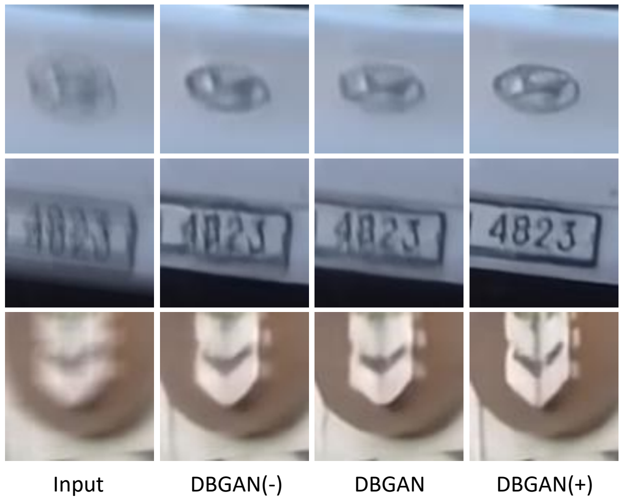


Figure 5. **Qualitative ablation results.** Examples of deblurred images generated by the proposed framework with different model structures. The first column shows input blurred images, and the next three columns are the deblurred images produced by DBGAN(-), DBGAN and DBGAN(+), respectively.

Fig. 5 shows results of the qualitative comparison. The proposed DBGAN outperforms the DBGAN(-), which shows the effectiveness of the relativistic loss function for image deblurring. With the learning-to-blur module, DBGAN(+) achieves a further improvement over DBGAN, suggesting the benefits of learning to deblur by learning to blur.

4.4. Comparison with Existing Methods

To verify the effectiveness of our model, we compare its performance with several state-of-the-art approaches on the GOPRO dataset quantitatively and qualitatively. [12] by

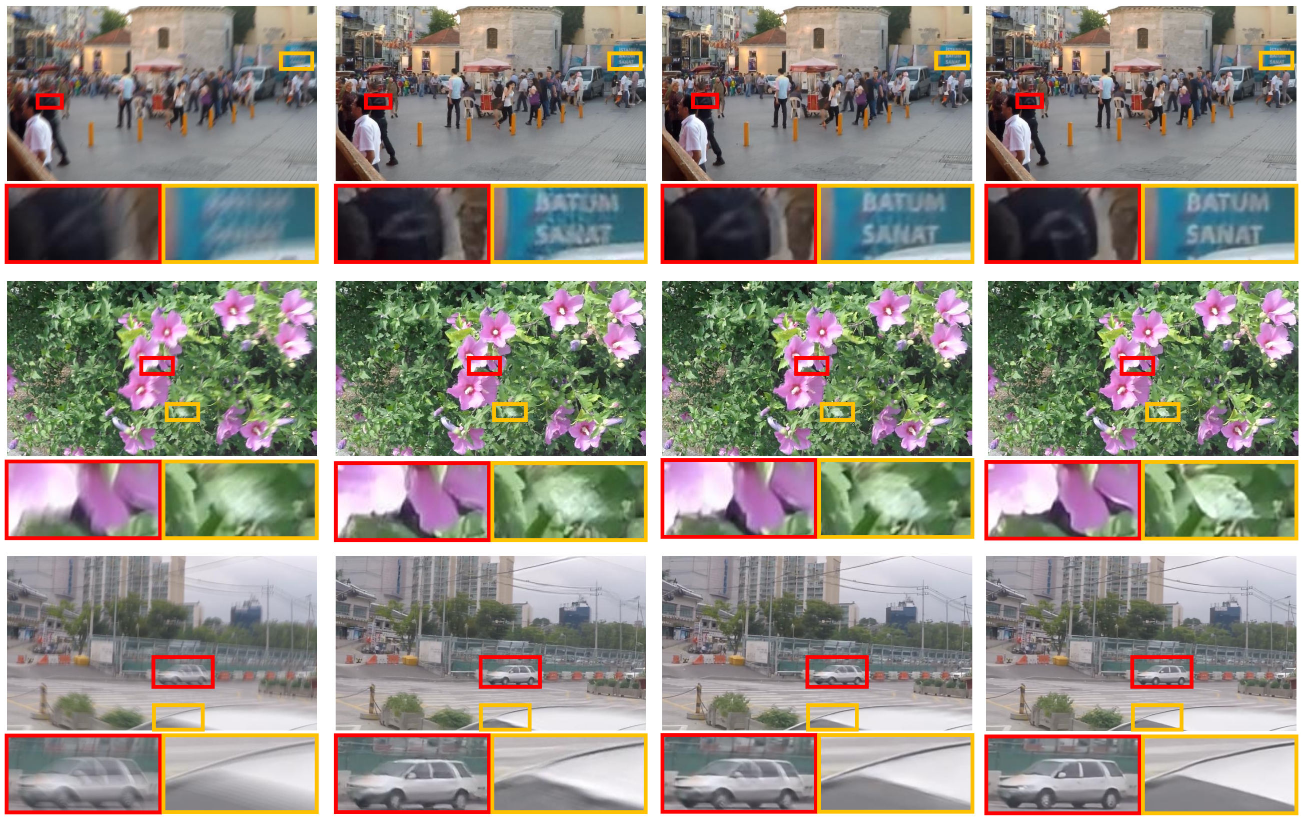


Figure 6. **Comparison with state-of-the-art deblurring methods.** From left to right: blurry images, results of Nah *et al.* [21], Tao *et al.* [36] and the proposed DBGAN(+) method. The improvement is clearly visible in the magnified patches.

Kim *et al.* is a traditional method to handle complex dynamic blurring images. For deep learning methods, Sun *et al.* [34] use a CNN network to estimate blur kernels and apply traditional deconvolution methods to synthesize sharp images. Nah *et al.* [21] propose a multi-scale function to model the coarse-to-fine approach. Similar to [21], Tao *et al.* [36] propose a multi-scale network via sharing network weights between different scales to recover sharp images. In addition, Shen *et al.* [30] introduce a human-aware deblurring method and Gao *et al.* [6] propose a nested skip connection structure and achieve state-of-the-art performance. Table 1 shows the results of the quantitative comparison. DBGAN outperforms most of previous methods, while DBGAN(+) achieves the state-of-the-art performance due to the framework of learning to deblur by learning to blur. For fair comparison, all values refer to the performance achieved by single model trained on the GOPRO dataset. Qualitative comparisons with some state-of-the-art methods are shown in Fig. 6, demonstrating that our method consistently achieves better visual quality results. Fig. 7 compare the proposed method with Gao *et al.** [6]. $*$ means this model is trained with extra pairs of images.

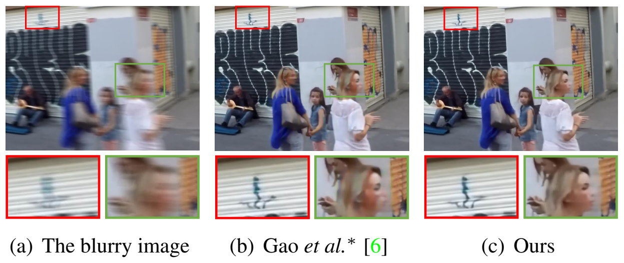


Figure 7. **Comparison with [6], which is trained with extra pairs of images.**

4.5. Performance in Real-World Scenarios

To validate the effectiveness of our method, we compare the performance of our approach with several state-of-the-art methods on the RWBI dataset of real-world blurry images. Fig. 8 shows qualitative results of different models. The blurry images in the first column are from the RWBI dataset, and the images in the following columns are the results of Nah *et al.* [21], Tao *et al.* [36] and the proposed DBGAN(+). Fig. 8 shows that our method achieves better performance on real-world blurry images.

Table 1. *Performance comparison on the* GOPRO_Large *dataset.*

Method	Kim *et al.*	Sun *et al.*	Nah *et al.*	Tao *et al.*	Shen *et al.*	Gao *et al.*	**DBGAN**	**DBGAN(+)**
PSNR	23.64	24.64	29.08	30.10	30.26	30.92	**30.43**	**31.10**
SSIM	0.8239	0.8429	0.9135	0.9323	0.940	0.9421	**0.9372**	**0.9424**

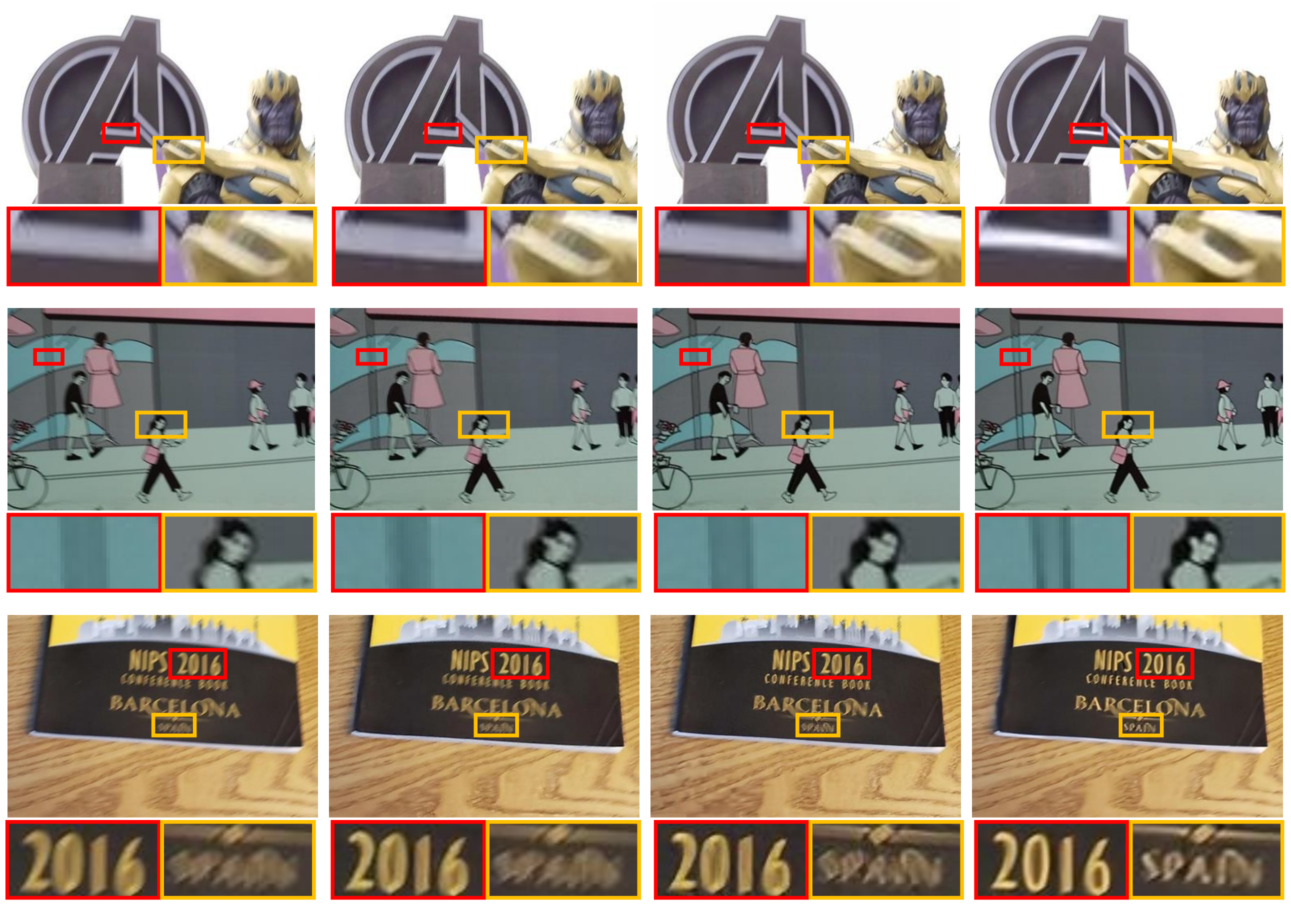


Figure 8. **Performance comparison on real-world blurry images.** From left to right: blurry images, results of Nah *et al.* [21], Tao *et al.* [36] and the proposed DBGAN(+) method. The improvement is clearly visible in the magnified patches.

5. Conclusion

This paper has presented a new framework which firstly learns how to transfer sharp images to realistic blurry images via a learning-to-blur GAN (BGAN) module. This framework trains a learning-to-deblur GAN (DBGAN) module to learn how to recover a sharp image from a blurry image. In contrast to prior work which solely focuses on learning to deblur, our method learns to realistically synthesize blurring effects using unpaired sharp and blurry images. In order to generate more realistic blurred images, a relativistic blur loss is employed to help the BGAN module reduce the gap between synthesized blur and real blur. In addition, a RWBI dataset is built to help train and test deblurring models. The Experimental results have demonstrated that our method not only yields results of consistently superior perceptual quality, but also outperforms state-of-the-art methods quantitatively.

Acknowledgment

This work is funded in part by the ARC Centre of Excellence for Robotics Vision (CE140100016), ARC-Discovery (DP 190102261) and ARC-LIEF (190100080) grants, as well as a research grant from Baidu on autonomous driving. The authors gratefully acknowledge the GPUs donated by NVIDIA Corporation. We thank all anonymous reviewers and ACs for their constructive comments.

References

[1] Adrian Bulat, Jing Yang, and Georgios Tzimiropoulos. To learn image super-resolution, use a gan to learn how to do image degradation first. In *ECCV*, 2018. 3

[2] Ayan Chakrabarti. A neural approach to blind motion deblurring. In *ECCV*, 2016. 3, 4, 5

[3] Tony F Chan and Chiu-Kwong Wong. Total variation blind deconvolution. *TIP*, 1998. 2

[4] Huaijin Chen, Jinwei Gu, Orazio Gallo, Ming-Yu Liu, Ashok Veeraraghavan, and Jan Kautz. Reblur2deblur: Deblurring videos via self-supervised learning. In *ICCP*, 2018. 3

[5] Sunghyun Cho and Seungyong Lee. Fast motion deblurring. *TOG*, 2009. 1

[6] Hongyun Gao, Xin Tao, Xiaoyong Shen, and Jiaya Jia. Dynamic scene deblurring with parameter selective sharing and nested skip connections. In *CVPR*, 2019. 1, 3, 7

[7] Amit Goldstein and Raanan Fattal. Blur-kernel estimation from spectral irregularities. In *ECCV*, 2012. 1

[8] Ankit Gupta, Neel Joshi, C Lawrence Zitnick, Michael Cohen, and Brian Curless. Single image deblurring using motion density functions. In *ECCV*, 2010. 2

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3, 4

[10] Michael Hirsch, Christian J Schuler, Stefan Harmeling, and Bernhard Schölkopf. Fast removal of non-uniform camera shake. In *ICCV*, 2011. 2

[11] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 4

[12] Tae Hyun Kim, Byeongjoo Ahn, and Kyoung Mu Lee. Dynamic scene deblurring. In *ICCV*, 2013. 6

[13] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016. 4

[14] Alexia Jolicoeur-Martineau. The relativistic discriminator: a key element missing from standard gan. *ICLR*, 2019. 4

[15] Dilip Krishnan and Rob Fergus. Fast image deconvolution using hyper-laplacian priors. In *NeurIPS*, 2009. 2

[16] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiří Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *CVPR*, 2018. 3, 6

[17] Orest Kupyn, Tetiana Martyniuk, Junru Wu, and Zhangyang Wang. Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. In *ICCV*, 2019. 3

[18] Dongxu Li, Cristian Rodriguez, Xin Yu, and Hongdong Li. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. In *WACV*, 2020. 3

[19] Dongxu Li, Xin Yu, Chenchen Xu, Lars Petersson, and Hongdong Li. Transferring cross-domain knowledge for video sign language recognition. *arXiv preprint arXiv:2003.03703*, 2020. 3

[20] Yu Liu, Fangyin Wei, Jing Shao, Lu Sheng, Junjie Yan, and Xiaogang Wang. Exploring disentangled feature representation beyond face identification. In *CVPR*, 2018. 3

[21] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *CVPR*, 2017. 1, 3, 4, 5, 6, 7, 8

[22] Jinshan Pan, Zhe Hu, Zhixun Su, and Ming-Hsuan Yang. Deblurring text images via l0-regularized intensity and gradient prior. In *CVPR*, 2014. 1

[23] Jinshan Pan, Yang Liu, Jiangxin Dong, Jiawei Zhang, Jimmy Ren, Jinhui Tang, Yu-Wing Tai, and Ming-Hsuan Yang. Physics-based generative adversarial models for image restoration and beyond. *TPAMI*, 2018. 3

[24] Wenqi Ren, Lin Ma, Jiawei Zhang, Jinshan Pan, Xiaochun Cao, Wei Liu, and Ming-Hsuan Yang. Gated fusion network for single image dehazing. In *CVPR*, 2018. 3

[25] Wenqi Ren, Jingang Zhang, Xiangyu Xu, Lin Ma, Xiaochun Cao, Gaofeng Meng, and Wei Liu. Deep video dehazing with semantic segmentation. *TIP*, 2018. 3

[26] Christian J Schuler, Michael Hirsch, Stefan Harmeling, and Bernhard Schölkopf. Learning to deblur. *TPAMI*, 2016. 5

[27] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. In *ICCV*, 2019. 2, 3

[28] Qi Shan, Jiaya Jia, and Aseem Agarwala. High-quality motion deblurring from a single image. *TOG*, 2008. 2

[29] Ziyi Shen, Wei-Sheng Lai, Tingfa Xu, Jan Kautz, and Ming-Hsuan Yang. Deep semantic face deblurring. In *CVPR*, 2018. 3

[30] Ziyi Shen, Wenguan Wang, Xiankai Lu, Jianbing Shen, Haibin Ling, Tingfa Xu, and Ling Shao. Human-aware motion deblurring. In *ICCV*, 2019. 3, 7

[31] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 2019. 3

[32] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015. 4

[33] Shuochen Su, Mauricio Delbracio, Jue Wang, Guillermo Sapiro, Wolfgang Heidrich, and Oliver Wang. Deep video deblurring for hand-held cameras. In *CVPR*, 2017. 1

[34] Jian Sun, Wenfei Cao, Zongben Xu, and Jean Ponce. Learning a convolutional neural network for non-uniform motion blur removal. In *CVPR*, 2015. 3, 4, 5, 7

[35] Yu-Wing Tai, Xiaogang Chen, Sunyeong Kim, Seon Joo Kim, Feng Li, Jie Yang, Jingyi Yu, Yasuyuki Matsushita, and Michael S Brown. Nonlinear camera response functions and image deblurring: Theoretical analysis and practice. *TPAMI*, 2013. 2

[36] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. In *CVPR*, 2018. 1, 3, 4, 7, 8

[37] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *ECCV*, 2018. 3

[38] Oliver Whyte, Josef Sivic, Andrew Zisserman, and Jean Ponce. Non-uniform deblurring for shaken images. *IJCV*, 2012. 2

[39] Wei Xiong, Jiahui Yu, Zhe Lin, Jimei Yang, Xin Lu, Connelly Barnes, and Jiebo Luo. Foreground-aware image inpainting. In *CVPR*, 2019. 3

[40] Li Xu and Jiaya Jia. Two-phase kernel estimation for robust motion deblurring. In *ECCV*, 2010. 1

[41] Li Xu, Shicheng Zheng, and Jiaya Jia. Unnatural l0 sparse representation for natural image deblurring. In *CVPR*, 2013. 1

[42] Jiawei Zhang, Jinshan Pan, Jimmy Ren, Yibing Song, Linchao Bao, Rynson WH Lau, and Ming-Hsuan Yang. Dynamic scene deblurring using spatially variant recurrent neural networks. In *CVPR*, 2018. 1, 3

[43] Kaihao Zhang, Wenhan Luo, Lin Ma, and Hongdong Li. Cousin network guided sketch recognition via latent attribute warehouse. In *AAAI*, 2019. 3

[44] Kaihao Zhang, Wenhan Luo, Lin Ma, Wei Liu, and Hongdong Li. Learning joint gait representation via quintuplet loss minimization. In *CVPR*, 2019. 3

[45] Kaihao Zhang, Wenhan Luo, Yiran Zhong, Lin Ma, Wei Liu, and Hongdong Li. Adversarial spatio-temporal learning for video deblurring. *TIP*, 2018. 3

[46] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *NeurIPS*, 2017. 4