A Novel Double-Tail Generative Adversarial Network for Fast Photo Animation

Gang Liu[1*] and Xin Chen[2]

[1] School of Computer, Hubei University of Technology, Wuhan 430068, China

lg0061408@126.com

[2] AI Center, Wuhan TianYu Information Industry CO., LTD., Wuhan 430223, China

asher_chan@foxmail.com

Abstract. Photo animation is to transform photos of real-world scenes into anime style images, which is a challenging task in AIGC (AI Generated Content). Although previous methods have achieved promising results, they often introduce noticeable artifacts or distortions. In this paper, we propose a novel double-tail generative adversarial network (DTGAN) for fast photo animation. DTGAN is the third version of the AnimeGAN series. Therefore, DTGAN is also called AnimeGANv3. The generator of DTGAN has two output tails, a support tail for outputting coarse-grained anime style images and a main tail for refining coarse-grained anime style images. In DTGAN, we propose a novel learnable normalization technique, termed as linearly adaptive denormalization (LADE), to prevent artifacts in the generated images. In order to improve the visual quality of the generated anime style images, two novel loss functions suitable for photo animation are proposed: 1) the region smoothing loss function, which is used to weaken the texture details of the generated images to achieve anime effects with abstract details; 2) the fine-grained revision loss function, which is used to eliminate artifacts and noise in the generated anime style image while preserving clear edges. Furthermore, the generator of DTGAN is a lightweight generator framework with only 1.02 million parameters in the inference phase. The proposed DTGAN can be easily end-to-end trained with unpaired training data. Extensive experiments have been conducted to qualitatively and quantitatively demonstrate that our method can produce high-quality anime style images from real-world photos and perform better than the state-of-the-art models.

Keywords: AIGC, generative adversarial networks, photo animation, linearly adaptive denormalization, double-tail

1 Introduction

Animation can be seen everywhere in our daily life, which has become a very popular artistic form and has been widely applied in social media platforms, games, film and television productions, etc. Actually, the current anime creation process is usually labor intensive and requires professional drawing skills. For a professional anime artist, it often takes several hours to complete a page of high-quality work. Meanwhile, different anime artists usually establish a set of personalized production style in their creating career, which is difficult for others to imitate and reproduce. The creation sources of many well-known works of anime artists are derived from the experience and inspiration of real life. They tend to apply the abstract drawing expression of realistic characters and scenes to anime works, which also means that it is a common creative technique to convert real-world photos into anime materials. However, for those who do not have professional skills, it is still a very difficult task to quickly create their own high-quality anime works. Therefore, an effective approach to automatically convert real-world photos into attractive anime works with predefined styles is much desired. This can be described as a task of anime style transfer, which has gained a lot of attention from researchers in the fields of machine learning and computer vision.

Photo animation is the transformation of real-world photos into the anime style images and is essentially the task of anime style transfer. Anime style transfer is a branch of image-to-image translation [1-3]. In recent years, many image-to-image translation methods based on deep learning [4] have been proposed, such as neural style transfer (NST) methods [5-7] and generative adversarial networks (GANs) [8-9]. Currently, due to the excellent image synthesis ability of GANs, GANs have gradually become the commonly used style transfer methods and achieved some results [10-14]. Although these methods have achieved good performances in artistic style transfer, they suffer from obvious drawbacks in anime style transfer. Different from artistic style transfer [15-16], which transfers the texture and color of paintings to the target images, anime style transfer requires that the generated anime images are visually close to the hand-crafted anime works. Some anime style transfer methods [17-21] based on GANs can generate the results close to the works of anime artists, but the images generated by

* Corresponding Author

these methods often have some obvious problems, such as high-frequency noise, artifacts, excessive blur and semantic structure mismatch.

To overcome the above problems, we propose a novel double-tail generative adversarial network (DTGAN), which can synthesize high-quality anime images from real-world photos. SinceDTGANis the third version of the AnimeGAN series, another name of DTGAN is AnimeGANv3. The generator of DTGAN has two output tails: the support tail and the main tail. The support tail of DTGAN outputs the coarse-grained anime style images with specific high-frequency noise and artifacts. The main tail of DTGAN denoises and removes artifacts from the output of the support tail to generate the high-quality anime images. We present a novel learnable normalization technique, called linearly adaptive denormalization (LADE), to solve the artifacts in the generated anime images. LADE can obtain the global data distribution of all channels and use this distribution to guide instance normalization (IN) [22]. LADE effectively avoids the artifacts caused by IN in the generated images.

Furthermore, to achieve better anime visuals and prevent over-stylization, we also propose two new loss functions: the region smoothing loss function and the fine-grained revision loss function. The region smoothing loss function is used to weaken the complex texture details in the generated anime images. The fine-grained revision loss function is used to eliminate the artifacts and noise in the generated images while preserving clear edges.

To further improve the visual quality of the generated anime images, we also improve the grayscale style loss and the color reconstruction loss proposed in AnimeGAN [19]. The improved grayscale style loss uses the grayscale generated image and the grayscale anime image as inputs, which effectively avoids the interference of color of the anime image on anime texture learning. The improved color reconstruction loss use the Lab color space, which is closer to real human vision, instead of the original YUV color space.

Unlike the previous methods that used the residual module [23] to increase the depth of the generator, we use a lightweight attention module to greatly reduce the size of the generator, enabling faster photo animation. In fact, DTGAN has a more lightweight generator with only 1.02 million parameters in the inference phase. We train DTGAN with unpaired data in an unsupervised manner. In order to accelerate the convergence of the network and improve the stability of training, we perform an initialization training on the generator of DTGAN. Extensive experiments have been conducted to qualitatively and quantitatively demonstrate that DTGAN can quickly generate higher-quality anime style images and outperforms other state-of-the-art methods.

The main contributions of the proposed work can be summarized as follows:

(1) We propose a novel lightweight GAN-based dedicated generative model for fast photo animation. The proposed DTGAN has a unique double-tailed structure, and it can be easily end-to-end trained with unpaired training data. Our method is able to generate high-quality stylized anime images from real-world photos and performs better than the current state-of-the-art models. When the anime images from individual artists are used for training, our model can reproduce their anime style.

(2) We present a novel learnable normalization technique, called Linear Adaptive Instance Denormalization (LADE). It is used in the generator and the discriminator to generate anime works closer to the real style of anime artists.

(3) We propose two novel loss functions: the region smoothing loss function and the fine-grained revision loss function. The region smoothing loss function uses the region-smoothed references based on the super-pixel segmentation algorithm to reduce complex texture details from real photos. The fine-grained revision loss function uses a fine-grained revision module composed of two different smoothing operations based on the frequency domain to eliminate visual artifacts and noise of the generated anime images while preserving clear edges.

(4) We further improve the grayscale style loss and color reconstruction loss proposed in the literature [19]. The improved grayscale style loss will focus on the style transfer of the anime line texture without considering the color style, and the improved color reconstruction loss uses L1 loss in the Lab color space to promote the generated anime images to preserve the brightness and color of the source photos.

2 Related Work

For the task of anime style transfer, the main problem is that the paired data is unavailable or typically difficult to collect. Nevertheless, some recent works using unpaired data have shown their extensive potential in anime style transfer. These style transfer methods based on unsupervised learning can be divided into two main directions: neural style transfer (NST), and GAN-based cross-domain translation.

2.1 Neural Style Transfer

The goal of Neural Style Transfer is to synthesized an image with the style of one image and content of another. Inspired by the progress of convolutional neural networks (CNNs) [6], many approaches have been developed to synthesize novel images with different styles. Gatys et al. [5-6] proposed the pioneering NST works, which

utilized CNN's ability to extract abstract features to find the optimal match between content and style for artistic style transfer. Since the normalization methods and loss functions have a significant impact on the quality of the generated images, some scholars had proposed many novel normalization methods [22, 24-26] and loss functions [15, 27]. The literature [22] replaced batch normalization (BN) [28] with instance normalization (IN) to remove instance-specific contrast information from the content image to improve the quality of stylization. Huang et al. [24] proposed the Adaptive Instance Normalization (AdaIN), which adjusted the mean and variance of the content image features to match those of the style image features to learn style transfer instead of using the style capture ability of Gram matrices [16]. Johnson et al. [27] used perceptual loss based on similarity measure in high-level feature space constructed from pre-trained classification networks to accelerate stylization. Li et al. [16] proposed a CNN-based deep generative feed-forward network for diverse multi-texture synthesis and multi-style transfer, which enables efficient synthesis of multiple textures within one single network based on its diversity loss. Kolkin et al. [15] proposed an optimization-based style transfer method by Relaxed Optimal Transport and Self-Similarity (STROTSS) to achieve high-quality stylization. Although these methods have worked well in improving the visual quality and generating speed of typical artistic stylization, they cannot perform well in producing animation works with specific drawing styles. And their application is often limited to irregular abstract textures and colors due to the manner that the colors and textures are simply stylized on content images. When they are used for anime style transfer, they tend to obtain visually unsatisfactory effects.

2.2 GAN-based Cross-Domain Translation

Generative Adversarial Networks (GANs) have achieved impressive performance in various image-to-image translation tasks. A GAN consists of a generator and a discriminator. In training, the generator aims to force the distribution of the generated images to be similar to that of the real images in the training data while the discriminator tries to distinguish the generated images from real images. In recent years, a wide variety of GAN-based works have been proposed and applied to cross-domain translation, whose goal is to learn a mapping from a source domain to a significantly different target domain. Pix2Pix [1] proposed a unified framework for various image-to-image translation tasks based on conditional GANs and L1 loss. However, it requires paired data to supervise the training. For the task of anime style transfer, it is expensive and not easy to collect a large number of paired animation data. To overcome this fundamental limitation, many unpaired image-to-image translation approaches have been proposed. CycleGAN [3] proposed a cycle consistency loss and utilized a cyclic network structure composed of two pairs of generators and discriminators to perform image translation with unpaired training data. However, due to the strict pixel-level constraint of the cycle consistency loss, it cannot perform shape changes, remove large objects, or ignore irrelevant texture. Chen et al. [17] proposed a dedicated GAN-based approach called CartoonGAN that effectively learns the mapping from real-world photos to cartoon images using unpaired image sets for training. AnimeGAN that is a new lightweight GAN for fast animation style transfer was presented in the literature [19], it was improved from CartoonGAN and proposed three novel loss functions that are grayscale style loss, grayscale adversarial loss and color reconstruction loss to make the generated images have better animation visual effects. AnimeGANv2 [21] proposed a more lightweight generator network based on AnimeGAN, and replaced instance normalization with layer normalization to avoid high-frequency artifacts of the generated images. White-box [20] presented a GAN-based image cartoonization framework to generate cartoonized images from real-world photos, which was optimized with the guide of three extracted representations. It controls the style of the output by adjusting the weight of each representation in the loss function. However, the generator of White-box does not use the normalization layer, the cartoon effect it generates is unstable. In addition, it uses the post-processing method of guided filter [29], which increases the time consumption of the inference process and also leads to the visual problem that the edge of the generated cartoon image appears feathering.

Different from previous works, we propose a novel generator network, a novel normalization method and two novel loss functions suitable for photo animation in this paper. Experiments demonstrate that our method can provide higher quality anime stylization than previous works.

3 Proposed Method

3.1 Linearly Adaptive Denormalization

In the literatures [21, 30, 31], It has been reported that IN or AdaIN [24] is the main cause of artifacts exhibited in the generated images. Hence, inspired by AdaIN and IN, LADE is proposed to replace IN for avoiding artifacts. The illustration of LADE is shown as Fig.1.

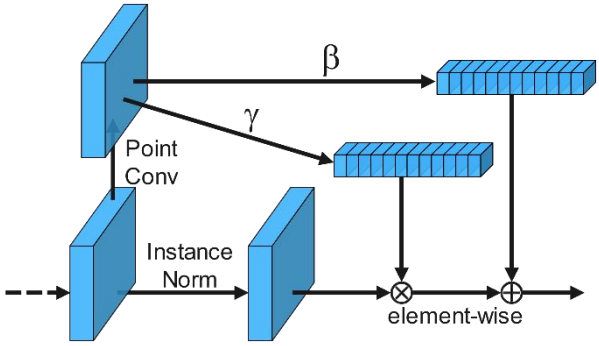

Fig. 1. In LADE, the features are convolved to produce the modulation parameters γ and β. γ and β are vectors. The produced γ and β are multiplied and added to the normalized activation element-wise

Formally, LADE can be defined as: given the feature $x \in R^{N*C*H*W}$, C is the number of channels, H and W are the height and width of the feature map, N is the number of samples in a batch, $n \in N$, $c \in C$, the proposed LADE can be denoted as:

$$LADE(x) = \gamma_{n,c}\frac{x-\mu_{n,c}(x)}{\sigma_{n,c}(x)+\varepsilon} + \beta_{n,c}\,. \tag{1}$$

Where ε is a tiny constant to avoid division by zero. The $\mu_{n,c}(x)$ and $\sigma_{n,c}(x)$ are the mean and standard deviation of the cth channel of the nth feature. The $\mu_{n,c}(x)$ and $\sigma_{n,c}(x)$ are calculated as follows:

$$\mu_{n,c}(x) = \frac{1}{HW}\sum_{h=1}^{H}\sum_{w=1}^{W} x_{n,c,h,w}\,. \tag{2}$$

$$\sigma_{n,c}(x) = \sqrt{\frac{1}{HW}\sum_{h=1}^{H}\sum_{w=1}^{W}\left(x_{n,c,h,w}-\mu_{n,c}(x)\right)^2+\varepsilon}\,. \tag{3}$$

The $\gamma_{n,c}$ and $\beta_{n,c}$ are the scale and bias of the cth channel of the nth feature. They are obtained by performing a pointwise convolution on the feature x and can be expressed as:

$$\beta_{n,c} = \frac{1}{HW}\sum_{h=1}^{H}\sum_{w=1}^{W} P\left(x_{n,c,h,w}\right)\,. \tag{4}$$

$$\gamma_{n,c} = \sqrt{\frac{1}{HW}\sum_{h=1}^{H}\sum_{w=1}^{W}\left(P\left(x_{n,c,h,w}\right)-\beta_{n,c}\right)^2+\varepsilon}\,. \tag{5}$$

where $P(x)$ represents the point convolution applied to x. After $P(\cdot)$ is applied to x, a new tensor $P(x)$ is obtained, and then the mean and standard deviation of each channel of $P(x)$ are calculated. Finally, the mean and the standard deviation of these channels constitute $\beta_{n,c}$ and $\gamma_{n,c}$, respectively.

LADE uses the pointwise convolution to linearly weight the features, and then obtains the global distribution information of the features. It is worth noting that γ and β are statistics of the features. This enables LADE to better learn the data distribution of the features and avoid artifacts.

3.2 Framework Overview

We propose an end-to-end generative framework, namely the double-tail generative adversarial network (DTGAN), to generate the high-quality anime images from real-world photos. The pipeline of DTGAN is shown in Fig.2.

In Fig.2, the green and blue cuboids represent the convolution modules which consist of a convolutional layer, a LADE layer and the LReLU activation layer [32]. EA is the external attention module [33] and VGG19 is the pre-trained VGG19 network [34]. DTGAN mainly consists of two discriminators and a generator with two output tails. It is worth noting that the structure of the two output tails is identical in the generator. The support tail has two outputs $G_{s0}(p)$ and $G_{s1}(p)$, $G_{s0}(p)$ is the unsmoothed image and $G_{s1}(p)$ is the smoothed. Since the non-parametric differentiable guided filter [29] can perform edge-preserving filtering while preserving the global semantic structure in the image, we use the guided filter to smooth $G_{s0}(p)$ to obtain $G_{s1}(p)$.

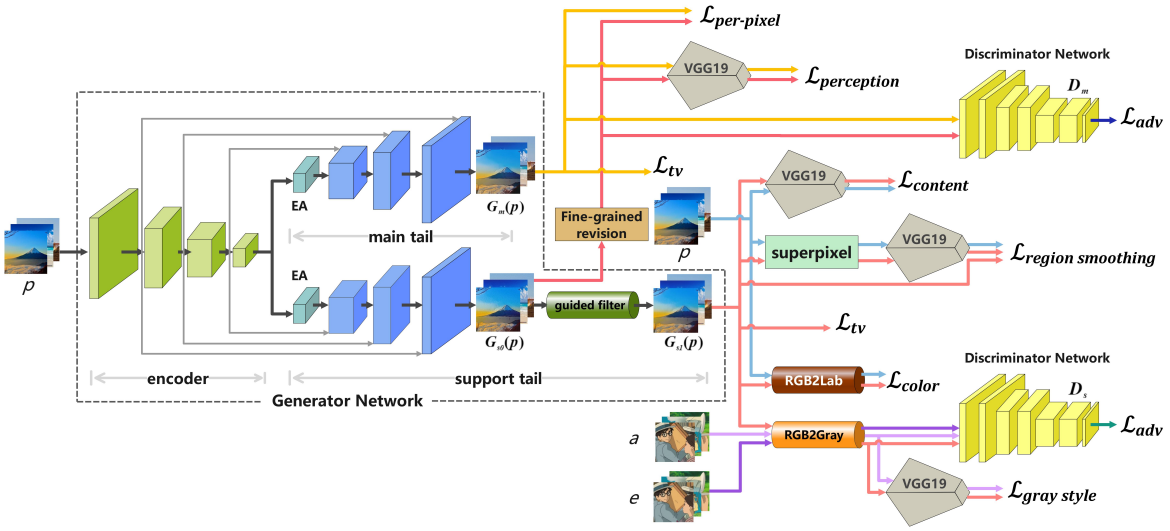

Fig. 2. The pipeline of DTGAN. The p represents real-world photo, a represents the anime image and e represents the anime image with blurred edges. The skip connections between the encoder and the two tails are connected using element-wise addition

From Fig.3, we can observe that $G_{s1}(p)$ has no clear edges and is rather blurry, and $G_{s0}(p)$ exhibits obvious high-frequency noise and artifacts. Although both $G_{s0}(p)$ and $G_{s1}(p)$ have the same anime styles, neither of them have satisfactory visual quality. The fine-grained revision module is proposed for fine-grained denoising and removal of visual artifacts on $G_{s0}(p)$. The high-quality anime style images output by the module are used as the ground truth and the task of the main tail of the generator is mainly to build the mapping from the input image to the ground truth. Fig.3 intuitively shows that the $G_m(p)$ generated by the main tail has the high-quality anime style while preserving clear edges. In DTGAN, the purpose of the support tail is to generate preliminary anime stylized images and revise them. Then, the revised images are used as auxiliary labels of the main tail to assist the learning process of the main tail. The purpose of the main tail is to generate the final anime stylized image with the aid of the supporting tail. The main tail is essentially used to make the final revision to the results of the support tail. In this way, an end-to-end anime stylized image generation method is achieved. DTGAN essentially only has the main tail, and the support tail is just an accessory used by the main tail in training. After training, the support tail is discarded. In the inference phase, DTGAN only contains the main tail. Therefore, the number of the network parameters used for inference is extremely small and can be directly deployed on mobile devices.

Different from existing methods [17-21], the generator of DTGAN uses the more lightweight external attention modules [33] instead of the deep residual modules [23] to achieve faster photo animation. The two linear layers in the external attention module adopt the weight-sharing mode to further reduce the number of parameters. Complementary to the generator network, two patch-level discriminators with the identical structure are used to distinguish whether the input local (49×49) image patches are from the real target manifold or from the synthesized image.

The process of transforming real-world photos into anime images is formulated as the mapping function in this paper. Let P, A and E indicate the photo domain, the anime domain and the blur anime domain respectively, with no pairing between them. Given a photo $p \in P$, DTGAN learns a mapping $\xi: P \to A$ that can transfer p to an anime sample $G(p) = \xi(p)$, while preserving the content of p and giving $G(p)$ the anime style of A. As shown in Fig.2, the generator takes p as input and then synthesizes three anime images ($G_{s0}(p)$, $G_{s1}(p)$ and $G_m(p)$). The $a \in A$ is the anime image in the anime dataset and $e \in E$ is the image obtained by blurring the edges of the anime images in the anime dataset. In the discriminator D_s, e is used as the negative sample. It is worth noting that $G_{s1}(p)$, a, and e are transformed to the grayscale images and fed to the discriminator D_s. The goal is to prompt the network to learn only line textures rather than the color styles in the anime domain. The D_s is introduced to distinguish whether the input images are from the synthesized anime images, the anime dataset or the blur anime dataset, and guides the support tail to learn the clear contours and the fine textures. For the main tail, the discriminator D_m is introduced to distinguish the real samples that are from the fine-grained revision module and the fake samples that are from the main tail, which is to enforce the main tail to synthesize the anime images with vivid edges and high visual quality.

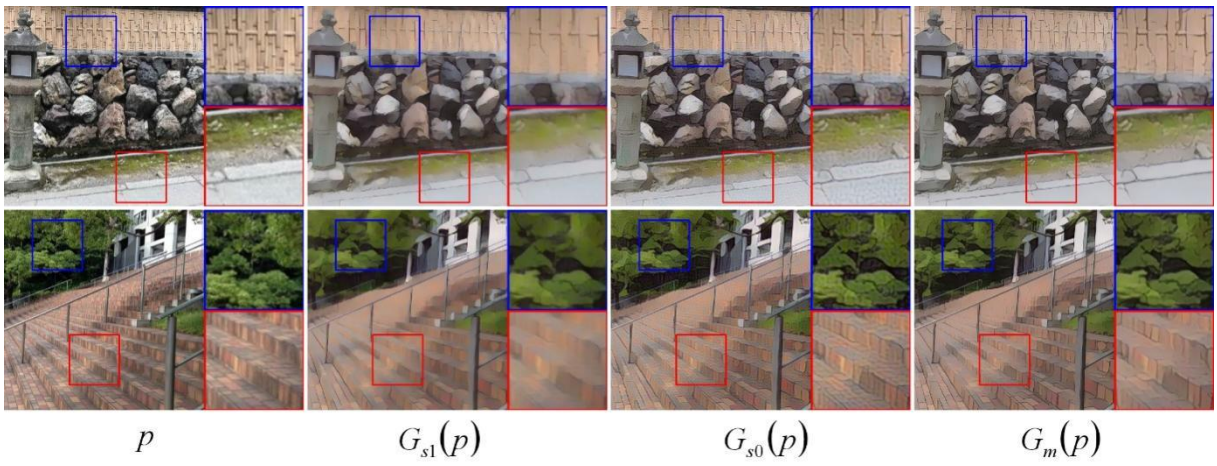

Fig. 3. Anime images with different visual qualities generated by two output tails. Zoom in for details

3.3 Loss Function

Our model has two learning objectives during training, namely image-to-image translation for the two output tails. In our work, instead of using the vanilla GAN objective, we use the least squares GAN [35] objective for stable training. The main tail and the support tail have their own different optimization goals and do not interfere with each other.

The losses of the support tail include the grayscale style loss, the content loss, the color reconstruction loss, the region smoothing loss, the adversarial loss and the total variation loss [27]. The loss function of the support tail can be simply expressed as:

$$L_{G_s} = \lambda_1 L_{con} + L_{gra} + L_{rs} + L_{s-adv} + \lambda_2 L_{col} + \lambda_3 L_{tv} \ . \tag{6}$$

where L_{s-adv} is the adversarial loss, L_{con} is the content loss, L_{gra} represents the grayscale style loss, L_{rs} represents the region smoothing loss, L_{col} is the color reconstruction loss and L_{tv} is the total variation loss. λ_1, λ_2 and λ_3 are the weights to balance six given loss functions. In our experiments, the generator generates good results when $\lambda_1 = 0.5$, $\lambda_2 = 20$ and $\lambda_3 = 0.001$.

Content loss. L_{con} is used to ensure that the generated images and the input photos are semantically identical. In L_{con}, a pre-trained VGG19 network is used as the perceptual network to extract high-level semantic features of the input images and the generated images. The content loss is defined as:

$$L_{con} = E_{p\sim P}\left[\frac{1}{C}\left\|VGG_n(p) - VGG_n\left(G_{s1}(p)\right)\right\|_1\right]. \tag{7}$$

where VGG_n refers to the feature maps of the nth layer in VGG19 and the nth layer is the "conv4-4" layer, C represents the number of channels of perceptual features.

Grayscale style loss. We adopt the grayscale style loss based on Gram matrix [16] to enhance the anime style of the generated images. L_{gra} is defined as:

$$\begin{aligned} L_{gra} = E_{p\sim P, a\sim A}\Bigg[\sum_n \phi_n \frac{1}{C}\Bigg\| & Gram\left(\Delta\left(VGG_n\left(f_{rgb2gray}(a)\right)\right)\right) - \\ & Gram\left(\Delta\left(VGG_n\left(f_{rgb2gray}\left(G_{s1}(p)\right)\right)\right)\right)\Bigg\|_1\Bigg]. \end{aligned} \tag{8}$$

where $Gram(\cdot)$ represents the Gram matrix of the features. $f_{rgb2gray}$ represents the derivable operation of RGB color images to grayscale images. ϕ_n represents the weight of the grayscale style loss of the nth layer in VGG19. In L_{gra}, the features of three layers ("conv2-2", "conv3-3" and "conv4-4") are used to calculate the total grayscale style loss to achieve stable stylization effect. In our work, $\phi_{conv2-2}$, $\phi_{conv3-3}$ and $\phi_{conv4-4}$ are 0.1, 5.0 and 25.0 respectively, achieving a balance between style and content. Inspired by the texture loss used in [16], we use the re-centered Gram matrices to avoid the artifacts and the color mixing problems in the synthesized results. The Δ indicates the zero-centered operation in the channel dimension.

Region smoothing loss. In order to effectively capture the global structural information and obtain a more stable anime abstraction effect, we use the region-smoothed input photos and the region-smoothed generated images as the smoothing reference for the generated results. L_{rs} is formulated as:

$$\mathrm{L_{rs}} = 0.2\mathrm{E_{p\sim P}}\left[\frac{1}{\mathrm{C}}\left\|\mathrm{VGG_n}\left(\mathrm{f_{sp}(p)}\right) - \mathrm{VGG_n}\left(\mathrm{G_{s1}(p)}\right)\right\|_1 + \frac{1}{\mathrm{C}}\left\|\mathrm{VGG_n}\left(\mathrm{f_{sp}}\left(\mathrm{G_{s1}(p)}\right)\right) - \mathrm{VGG_n}\left(\mathrm{G_{s1}(p)}\right)\right\|_1\right]. \tag{9}$$

where n refers to the "conv4-4" layer of VGG19 and f_{sp} represents the superpixel segmentation operation based on the felzenszwalb algorithm [36]. Superpixel segmentation is used as region smoothing in DTGAN. Using the region-smoothed photos as the references for the generated images enables the network to learn the abstract content and the simple textures, thereby reducing the complex texture details in the generated anime images. L_{rs} can effectively prevent over-stylization.

Color reconstruction loss. L_{col} based on the Lab color space is used to ensure that the generated anime images can effectively preserve the brightness and color of the input photos. Formally, L_{col} can be defined as:

$$L_{col} = E_{p\sim P}\left[\left\|L(p) - L\left(G_{s1}(p)\right)\right\|_1 + \left\|a(p) - a\left(G_{s1}(p)\right)\right\|_1 + \left\|b(p) - b\left(G_{s1}(p)\right)\right\|_1\right]. \quad (10)$$

where $L(p)$, $a(p)$, $b(p)$ represent the three channels of the image p in the Lab format, respectively.

Adversarial loss and total variation loss. L_{s-adv} and L_{tv} are expressed as follows:

$$L_{s-adv} = E_{p\sim P}\left[\left(1 - D_s\left(f_{rgb2gray}\left(G_{s1}(p)\right)\right)\right)^2\right]. \quad (11)$$

$$L_{tv} = E_{p\sim P}\left[\frac{1}{2HWC}\sum_{i,j,k}\left[\left\|G_{s1}(p)_{i,j+1,k} - G_{s1}(p)_{i,j,k}\right\|_2 + \left\|G_{s1}(p)_{i+1,j,k} - G_{s1}(p)_{i,j,k}\right\|_2\right]\right]. \quad (12)$$

The loss function of the discriminator D_s can be formulated as:

$$L_{D_s} = L_{gra-adv} + L_{edge-adv}\ . \quad (13)$$

where $L_{gra-adv}$ is the grayscale style adversarial loss and $L_{edge-adv}$ is the edge-promoting adversarial loss [17]. These loss functions are expressed as follows:

$$L_{gra-adv} = 0.5E_{a\sim A}\left[\left(1 - D_s\left(f_{rgb2gray}(a)\right)\right)^2\right] + E_{p\sim P}\left[\left(D_s\left(f_{rgb2gray}\left(G_{s1}(p)\right)\right)\right)^2\right]. \quad (14)$$

$$L_{edge-adv} = E_{e\sim E}\left[\left(D_s\left(f_{rgb2gray}(e)\right)\right)^2\right]. \quad (15)$$

$L_{gra-adv}$ utilizes the grayscale anime images to force the network to focus on discriminating the anime textures and avoid color interference. $L_{edge-adv}$ is used to preserve the clear edges of the generated images.

The loss function of the main tail is mainly the fine-grained revision loss function, which is composed of the per-pixel loss and the perceptual loss. The loss function of the main tail can be defined as:

$$L_{G_m} = \eta_1 L_{per-pixel} + \eta_2 L_{perception} + \eta_3 L_{m-adv} + \eta_4 L_{tv}\ . \quad (16)$$

where $L_{per-pixel}$ is the per-pixel loss, $L_{perception}$ is the perceptual loss and L_{m-adv} is the adversarial loss. L_{tv} indicates the total variation regularization on the generated image $G_m(p)$. η_1 to η_4 are used as the weights to balance different losses. When η_1 = 50, η_2 = 0.5, η_3 = 0.02 and η_4 = 0.001, we find that the generated images have good visual quality.

It is worth noting that the proposed fine-grained revision module is composed of two image smoothing algorithms based on frequency domain, namely NL-means [37] and L0 Smoothing [38]. Fig.4 shows some results generated by the fine-grained revision module. As can be seen from Fig.4, NL-means can effectively remove the high-frequency noise in $G_{s0}(p)$, and L0 Smoothing can effectively eliminate the visual artifacts in $G_{s0}(p)$. when these two smoothing algorithms are used simultaneously, the high-quality anime images with clear edges can be obtained. Hence, we take the high-quality revised images as the ground truth for the fine-grained revision loss functions.

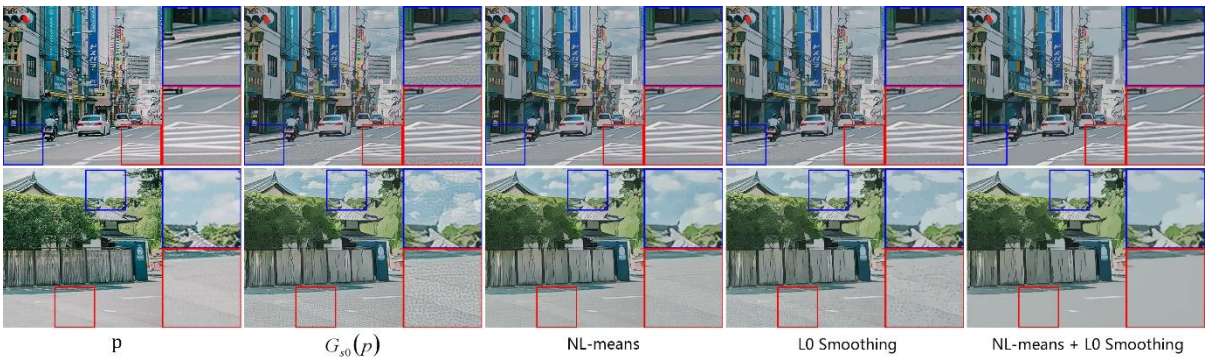

Fig. 4. Effect of the fine-grained revision module. Zoom in for details

Let f_{fgr} denote the fine-grained revision module, and each loss function in L_{G_m} can be expressed as:

$$L_{per-pixel} = E_{p\sim P}\left[\left\|G_m(p) - f_{fgr}(G_{s0}(p))\right\|_1\right]. \tag{17}$$

$$L_{perception} = E_{p\sim P}\left[\frac{1}{C}\left\|VGG_n(G_m(p)) - VGG_n\left(f_{fgr}(G_{s0}(p))\right)\right\|_1\right]. \tag{18}$$

$$L_{m-adv} = E_{p\sim P}\left[\left(1 - D_m(G_m(p))\right)^2\right]. \tag{19}$$

where n refers to the "conv4-4" layer of VGG19. The ground truth used in L_{G_m} is $f_{fgr}(G_{s0}(p))$.

Then, the loss of the discriminator D_m can be expressed as follows:

$$L_{D_m} = E_{p\sim P}\left[\left(1 - D_m\left(f_{fgr}(G_{s0}(p))\right)\right)^2\right] + E_{p\sim P}\left[\left(D_m(G_m(p))\right)^2\right]. \tag{20}$$

In summary, the total loss of the generator contains the losses of two output tails. It can be expressed as:

$$L_G = L_{G_s} + L_{G_m}. \tag{21}$$

Similarly, the total loss of the discriminator is the sum of the losses of the two discriminators. It can be defined as:

$$L_D = L_{D_s} + L_{D_m}. \tag{22}$$

4 Experiments

4.1 Training Details

We use TensorFlow [39] to implement our model and all experiments are performed on a computer with a single NVIDIA 2080Ti GPU. We apply the spectral normalization [40] to all the layers in the two discriminators and employ the Adam solver [41] with a batch size of 8. The learning rates for the generator and discriminator are set to 0.0001 and 0.0002, respectively. The two tails of the generator are first pre-trained for 5 epochs using only the content loss with a learning rate of 0.0002. Training is stopped after 100 epochs or on convergency. During inference, we only run the main tail of the generator.

4.2 Dataset and Evaluation Metrics

Dataset. In our work, real-world photos and anime images are used as the unpaired training data, and the test data only includes real-world photos. We employ 6,648 real-world landscape photos from the photo data of CycleGAN [3] and 1,792 anime images from the Hayao Style data of AnimeGAN for training. For the validation set, we collect 1,591 real-world photos from the DIV2K dataset [42] and Internet. During training, all images are scaled to 256 × 256.

Evaluation metrics. To quantitatively evaluate the performance of photo animation, we use the Fréchet Inception Distance (FID) [43] and the Kernel Inception Distance (KID) [44] which have been popularly used to evaluate the quality of synthetic images in image translation tasks. The FID score evaluates the distribution

discrepancy between the synthesized results and source images in the high-dimensional feature space. A lower FID score indicates that the distribution of generated images is more similar to that of real anime works. The KID score computes the squared Maximum Mean Discrepancy between the feature representations of real and generated images. The lower KID indicates that the more shared visual similarities between real and generated images. For qualitative evaluation, we present results with details of five related state-of-the-art methods and original images, as well as qualitative analysis.

4.3 Quantitative and Qualitative Comparisons

We compare our approach with five state-of-the-art style transfer methods that have achieved some significant results in recent year. The SOTA methods include: AdaIN, CartoonGAN [17], AnimeGAN, AnimeGANv2 [21] and White-box [20]. From Table 1, our method achieves the lowest scores on both FID to anime image distribution and KID to anime image distribution, which proves DTGAN generates results most similar to anime images. AnimeGANv2 outperforms all methods on FID to photo distribution and KID to photo distribution, which indicates that AnimeGANv2 preserve more the content of photos than other methods. However, it is not as good as our approach in anime style. Our model can process a 1920 × 1080 image on GPU within only 115.50 ms, which is the fastest inference speed among all methods. Moreover, DTGAN has the smallest model size among all methods.

Table 1. Quantitative comparisons of all methods. KID ± *std.* for different methods. Lower is better for FID and KID. LR means 256 × 256 resolution. HR means 1920 × 1080 resolution. "ms" means millisecond

Methods	AdaIN	CartoonGAN	AnimeGAN	AnimeGANv2	White-box	Ours
FID to Photo	149.55	66.51	41.66	**34.55**	42.31	40.77
KID to Photo	9.43±0.69	1.65±0.45	0.44±0.27	**0.28±0.18**	0.45±0.21	0.42±0.25
FID to Anime	204.45	89.94	80.36	78.10	85.13	**75.95**
KID to Anime	0.16±0.008	0.026±0.002	0.017±0.002	0.016±0.002	0.023±0.002	**0.015±0.002**
LR (ms)	10.83	15.23	18.75	12.53	**6.97**	14.92
HR (ms)	193.87	471.11	557.11	326.30	149.92	**115.50**
Parameter(M)	7.01	11.12	3.95	2.14	1.46	**1.02**

The qualitative results of all methods are shown in Fig.5. The results generated by our method effectively preserve the color and brightness of the input photos while exhibiting clear edges. AdaIN generates the over-stylized results and the images generated by CartoonGAN have significant visual artifacts. Although AnimeGAN can preserve more details of the input photos, AnimeGAN weakens the abstract anime style. As an improved version of AnimeGAN, AnimeGANv2 enhances the anime style of the generated images, but the generated images suffer from blurred details and edge feathering. White-box causes over-smoothed color and fails to generate clear edges. Compared with all SOTA methods, our method produces the most satisfactory results with clear edges, authentic colors, actual brightness, abstract details and less noise. In summary, our method can generate high-quality anime images from real-world photos and outperform all SOTA methods. More examples of the results comparing our model with other models are included in the appendix.

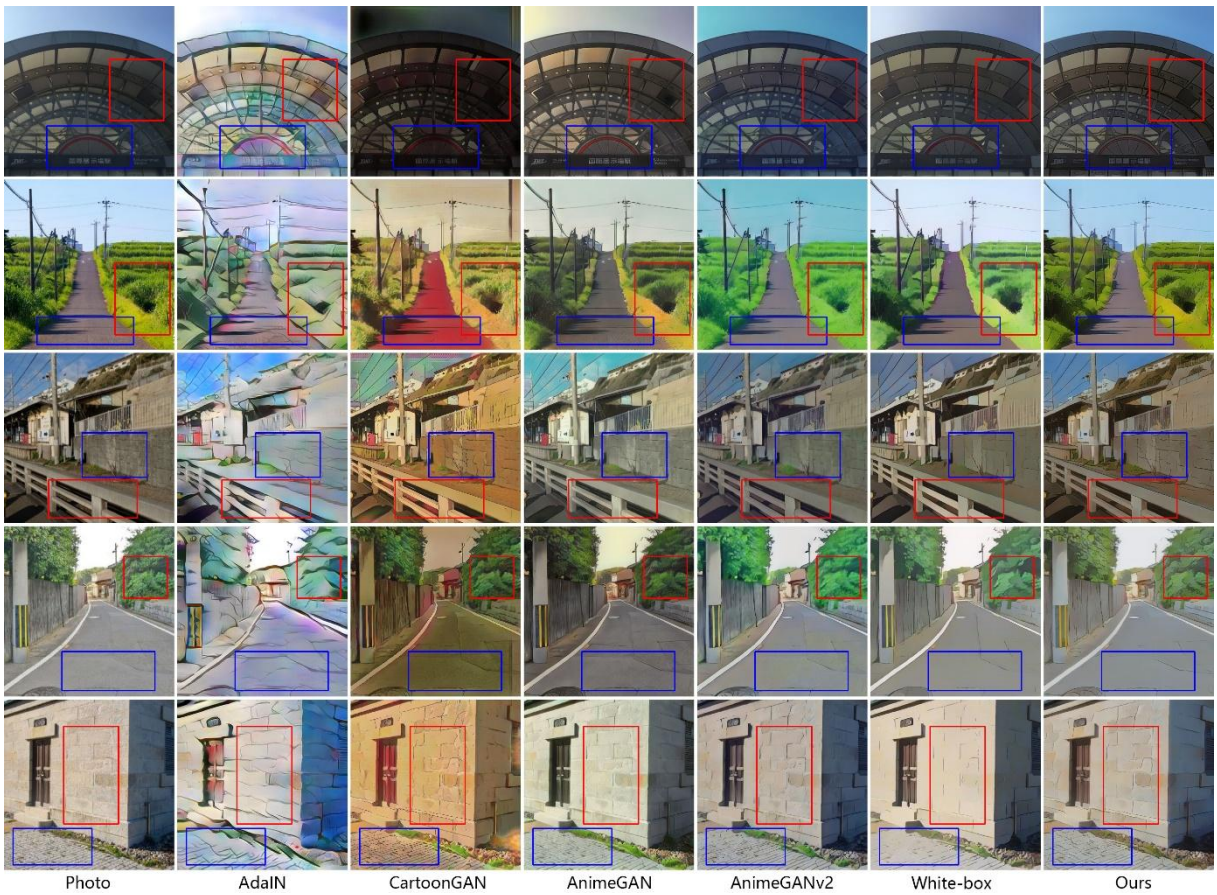

Fig. 5. Qualitative comparisons of different methods. Zoom in for details

4.4 Ablation Studies

We conducted an ablation study to confirm that LADE has better stylization performance compared with other 4 widely used normalization methods. These normalization methods include: batch normalization (BN) [28], layer normalization (LN) [45], instance normalization (IN) and group normalization (GN) [46]. The results are presented in Table 2. DTGAN with LADE achieves the lowest scores on both FID and KID, which proves that LADE can effectively help the model preserve the content of the input photos and obtain the results closer to the real anime images. From Table 2, the number of parameters of LADE is only 0.1M more than IN and LN, and 0.09M more than BN and GN. Therefore, the number of parameters of LADE is not the decisive factor for its performance gain.

Table 2. Quantitative comparisons of different normalization methods. KID ± *std.* for different methods. Lower is better for FID and KID

Methods	BN	IN	LN	GN	Ours
FID to Photo	56.14	59.10	42.04	51.81	**40.77**
KID to Photo	1.45 ± 0.37	1.27 ± 0.41	0.57 ± 0.22	0.84 ± 0.31	**0.42 ± 0.25**
FID to Anime	120.66	99.00	99.92	88.35	**75.95**
KID to Anime	0.062 ± 0.004	0.032 ± 0.002	0.037 ± 0.002	0.023 ± 0.002	**0.015 ± 0.002**
Parameter(M)	0.93	0.92	0.93	0.92	1.02

The qualitative results of all methods are shown in Fig.6. What can be clearly seen in this figure is the visual quality of the results generated by LADE is significantly better than other methods. The images produced by BN and LN have a large number of cracks. The images generated by IN and GN suffer from obvious visual artifacts. This demonstrate that BN, LN, IN and GN do not solve anime style transfer well. To conclude, the proposed LADE has superior anime stylization performance compared with other 4 commonly used normalization methods.

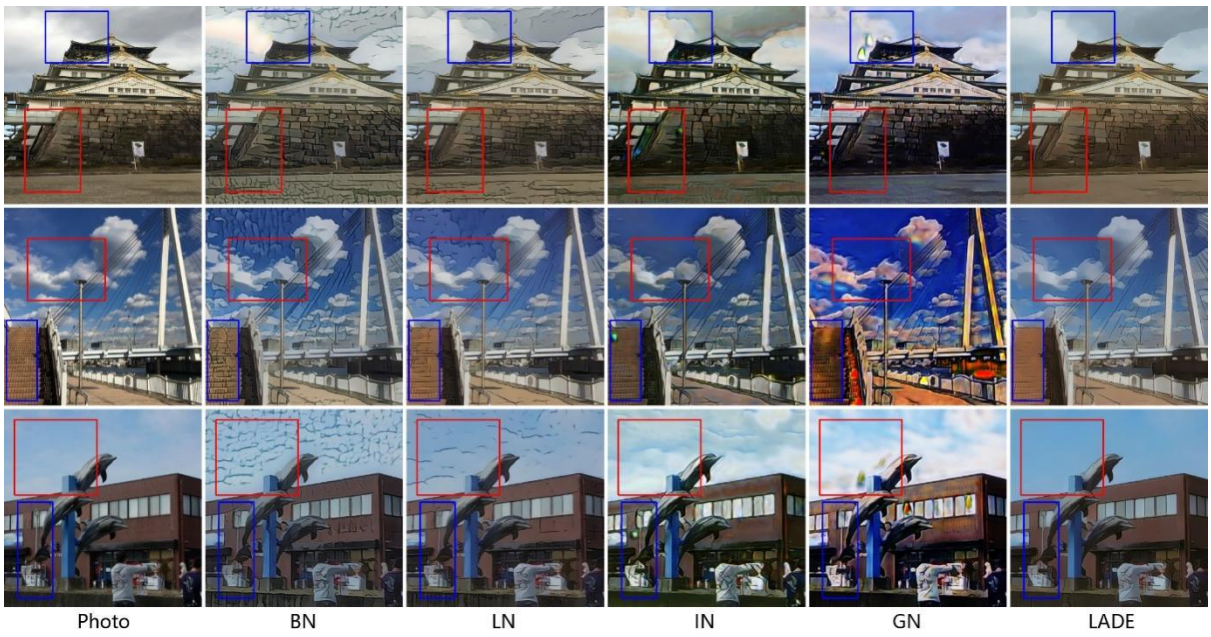

Fig. 6. Qualitative comparisons of different normalization methods. Zoom in for details

In DTGAN, we propose region smoothing loss function and fine-grained revision loss function, and improve grayscale style loss and color reconstruction loss. In order to evaluate the effects of the proposed loss functions, we conducted the ablation experiments to confirm the benefits of using them in DTGAN. According to the Table 3, (a) represents the grayscale style loss using RGB color anime images as the input (This loss is also called the color style loss.), (b) represents DTGAN without the fine-grained revision loss function, (c) represents DTGAN without the region smoothing loss function and (d) represents the color reconstruction loss using the YUV color space. From Table 3, it can be seen that our method achieves the lowest scores on both FID to photo distribution and KID to photo distribution. Simultaneously, our method also achieves the lowest scores on both FID to anime image distribution and KID to anime image distribution. Overall, it is demonstrated that the proposed loss functions can effectively help our model to produce higher quality anime images.

Table 3. Quantitative comparisons of different loss functions. KID±*std.* for different methods

Methods	(a)	(b)	(c)	(d)	Ours
FID to Photo	76.87	49.46	47.26	41.61	**40.77**
KID to Photo	2.12 ± 0.48	0.84 ± 0.32	0.79 ± 0.34	0.55 ± 0.28	**0.42 ± 0.25**
FID to Anime	96.62	110.63	103.78	91.89	**75.95**
KID to Anime	0.033 ± 0.003	0.042 ± 0.004	0.037 ± 0.003	0.027 ± 0.002	**0.015 ± 0.002**

The qualitative results of all methods are shown in Fig.7. The proposed loss functions and the improved loss functions have obvious advantages. Compared with (a), our method can avoid the interference of color style from anime data and preserve the real color of the input photos by using grayscale anime images as true samples in adversarial loss of the support tail. Compared with (b), the fine-grained revision loss function including per-pixel loss and perceptual loss are able to help DTGAN to avoid visual artifacts in the generated images. Compared with (c), the region smoothing loss can help DTGAN to weaken high-frequency style texture details and make the generated images have more abstract and smooth effects. In contrast to (d) which generates blurred results, our method can generate clearer anime images by using color reconstruction loss based on Lab color space. The experiments show that the proposed loss functions and the improved loss functions have significant effects on the performance of the network and effectively improve the visual quality of the generated images.

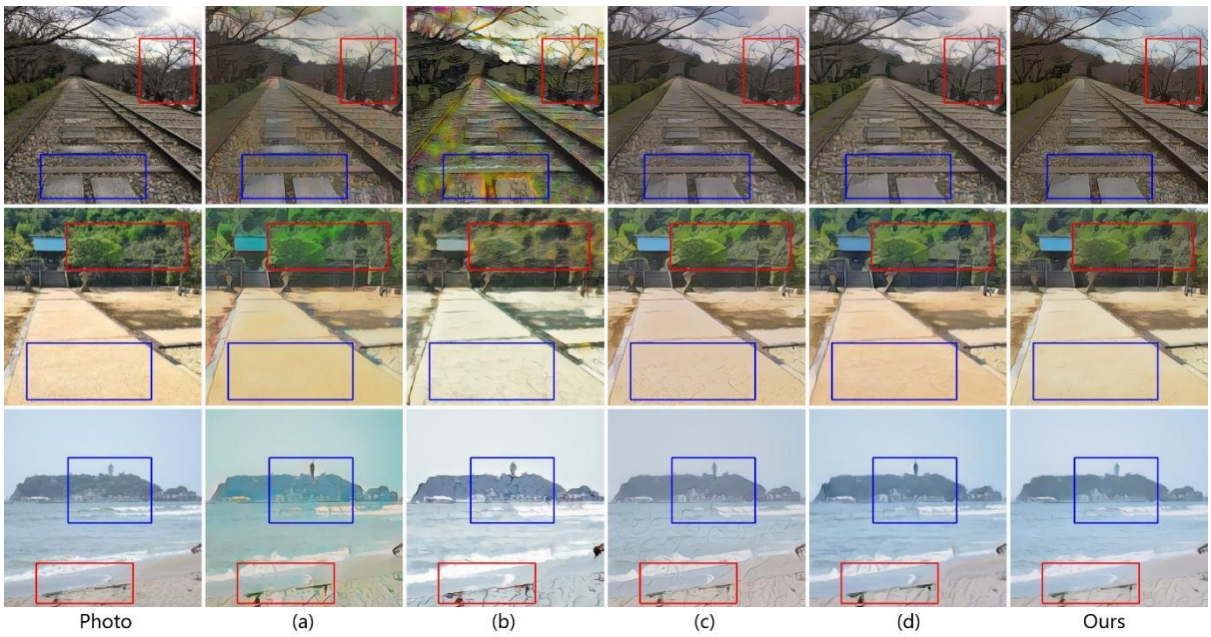

Fig. 7. Qualitative comparisons of different loss functions. (a) to (d) show the results with color style loss, without fine-grained revision loss, without region smoothing loss and with YUV color space in color reconstruction loss

5 Conclusions

In this paper, we propose DTGAN for fast generating high-quality anime images from real-world photos. The main contributions are as follows: 1) a novel lightweight GAN framework with double-tailed structure, in which the support tail is used to generate the coarse-grained anime images and the main tail is used to generate the fine-grained anime images; 2) a novel Linear Adaptive Instance Denormalization (LADE) for generating anime images closer to the real style of anime artists; 3) the region smoothing loss function for achieving abstract anime effects and the fine-grained revision loss function for improving the visual quality of the generated anime images; 4) the improved grayscale style loss function and the improved color reconstruction loss function. Experiments show that DTGAN can transform real-world photos into high-quality anime images and significantly outperforms some state-of-the-art methods. In the ablation experiments, we also analyzed the effects of LADE and two novel loss functions on the quality of the generated images. We find that LADE and two novel loss functions can effectively improve the quality of generated images.

References

[1] Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: Proc. 30th IEEE Conference on Computer Vision and Pattern Recognition, (CVPR'2017), Honolulu, HI, United states (2017) 5967-5976.

[2] Huang, X., Liu, M.Y., Belongie, S., Kautz, J.: Multimodal unsupervised image-to-image translation. In: Proc. 15th European Conference on Computer Vision, (ECCV'2018), Munich, Germany (2018) 179-196.

[3] Zhu, J.Y.., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proc. 16th IEEE International Conference on Computer Vision, (ICCV'2017), Venice, Italy (2017) 2242-2251.

[4] Schmidhuber, J.: Deep learning in neural networks: An overview. Neural Networks 61 (2015) 85-117.

[5] Gatys, L.A., Ecker, A.S., Bethge, M.: A neural algorithm of artistic style. CoRR abs/1508.06576, 2015.

[6] Gatys, L.A., Ecker, A.S., Bethge, M.: Image style transfer using convolutional neural networks. In: Proc. 29th IEEE Conference on Computer Vision and Pattern Recognition, (CVPR'2016), Las Vegas, NV, United states (2016) 2414-2423.

[7] Li, Y., Fang, C., Yang, J., Wang, Z., Lu, X., Yang, M.H.: Universal style transfer via feature transforms. In: Proc. 31st Annual Conference on Neural Information Processing Systems, (NIPS'2017), Long Beach, CA, United states (2017) 386-396.

[8] Wang, K., Gou, C., Duan, Y., Lin, Y., Zheng, X., Wang, F.Y.: Generative adversarial networks: Introduction and outlook. IEEE/CAA Journal of Automatica Sinica. 4 (2017) 588-598.

[9] Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Proc. 28th Annual Conference on Neural Information Processing Systems, (NIPS'2014), Montreal, QC, Canada (2014) 2672-2680.

[10] Kim, J., Kim, M., Kang, H., Lee, K.: U-GAT-IT: unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation. CoRR abs/1907.10830 (2019).

[11] Choi, Y., Choi, M., Kim, M., Ha, J.W., Kim, S., Choo, J.: Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In: Proc. 31st Meeting of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, (CVPR'2018), Salt Lake City, UT, United states (2018) 8789-8797.

[12] Yi, Z., Zhang, H., Tan, P., Gong, M.: Dualgan: Unsupervised dual learning for image-to-image translation. In: Proc. 16th IEEE International Conference on Computer Vision, (ICCV'2017), Venice, Italy (2017) 2868-2876.

[13] Ma, S., Fu, J., Chen, C.W., Mei, T.: Da-gan: Instance-level image translation by deep attention generative adversarial networks. In: Proc. 31st Meeting of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, (CVPR'2018), Salt Lake City, UT, United states (2018) 5657-5666.

[14] Li, B., Zhu, Y., Wang, Y., Lin, C., Ghanem, B., Shen, L.: Anigan: Style-guided generative adversarial networks for unsupervised anime face generation. CoRR abs/2102.12593 (2021).

[15] Kolkin, N., Salavon, J., Shakhnarovich, G.: Style transfer by relaxed optimal transport and self-similarity. In: Proc. 32nd IEEE/CVF Conference on Computer Vision and Pattern Recognition, (CVPR'2019), Long Beach, CA, United states (2019) 10043-10052.

[16] Li, Y., Fang, C., Yang, J., Wang, Z., Lu, X., Yang, M.H.: Diversified texture synthesis with feed-forward networks. In: Proc. 30th IEEE Conference on Computer Vision and Pattern Recognition, (CVPR'2017), Honolulu, HI, United states (2017) 266-274.

[17] Chen, Y., Lai, Y.K., Liu, Y.J.: Cartoongan: Generative adversarial networks for photo cartoonization. In: Proc. 31st Meeting of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, (CVPR'2018), Salt Lake City, UT, United states (2018) 9465-9474.

[18] Wu, R., Gu, X., Tao, X., Shen, X., Tai, Y., Jia, J.: Landmark assisted cyclegan for cartoon face generation. CoRR abs/1907.01424 (2019).

[19] Chen, J., Liu, G., Chen, X.: Animegan: A novel lightweight gan for photo animation. In: Proc. 11th International Symposium on Intelligence Computation and Applications, (ISICA'2019), Guangzhou, China (2019) 242-256.

[20] Wang, X., Yu, J.: Learning to cartoonize using white-box cartoon representations. In: Proc. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, (CVPR'2020), Virtual, Online, United states (2020) 8087-8096.

[21] Chen, X., Liu, G.: Animeganv2. <https://tachibanayoshino.github.io/AnimeGANv2>, 2020.

[22] Ulyanov, D., Vedaldi, A., Lempitsky, V.S.: Instance normalization: The missing ingredient for fast stylization. CoRR abs/1607.08022 (2016).

[23] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proc. 29th IEEE Conference on Computer Vision and Pattern Recognition, (CVPR'2016), Las Vegas, NV, United states (2016) 770-778.

[24] Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: Proc. 5th International Conference on Learning Representations, (ICLR'2017), Toulon, France (2017).

[25] Nam, H., Kim, H.E.: Batch-instance normalization for adaptively style-invariant neural networks. In: Proc. 32nd Conference on Neural Information Processing Systems, (NeurIPS'2018), Montreal, QC, Canada (2018) 2558-2567.

[26] Jing, Y., Liu, X., Ding, Y., Wang, X., Ding, E., Song, M., Wen, S.: Dynamic instance normalization for arbitrary style transfer. In: Proc. 34th AAAI Conference on Artificial Intelligence, (AAAI'2020), New York, NY, United states (2020) 4369-4376.

[27] Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: Proc. 14th European Conference on Computer Vision, (ECCV'2016), Amsterdam, Netherlands (2016) 694-711.

[28] Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. CoRR abs/1502.03167 (2015).

[29] Wu, H., Zheng, S., Zhang, J., Huang, K.: Fast end-to-end trainable guided filter. In: Proceedings of 31st Meeting of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, (CVPR'2018), Salt Lake City, UT, United states, IEEE Computer Society (2018) 1838-1847.

[30] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: Proceedings of 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, (CVPR-2020), Virtual, Online, United states (2020) 8107-8116.

[31] Jung, D., Yang, S., Choi, J., Kim, C.: Arbitrary style transfer using graph instance normalization. In: Proceedings of 2020 IEEE International Conference on Image Processing, (ICIP'2020), Virtual, Abu Dhabi, United arab emirates (2020) 1596-1600.

[32] Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: ICML Workshop on Deep Learning for Audio, Speech and Language Processing. Volume 30. (2013).

[33] Guo, M., Liu, Z., Mu, T., Hu, S.: Beyond self-attention: External attention using two linear layers for visual tasks. In: Proc. IEEE Transactions on Pattern Analysis & Machine Intelligence 01 (2022): 1-13.

[34] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556 (2014).

[35] Mao, X., Li, Q., Xie, H., Lau, R.Y., Wang, Z., Smolley, S.P.: Least squares generative adversarial networks. In: Proceedings of 16th IEEE International Conference on Computer Vision, (ICCV'2017), Venice, Italy, IEEE Computer Society (2017) 2813-2821.

[36] Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. International Journal of Computer Vision 59 (2004) 167-181.

[37] Buades, A., Coll, B., Morel, J.M.: A non-local algorithm for image denoising. In: Proceedings of 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR'2005), San Diego, CA, United states, IEEE Computer Society (2005) 60-65.

[38] Xu, L., Lu, C., Xu, Y., Jia, J.: Image smoothing via l0 gradient minimization. In: Proceedings of the 2011 SIGGRAPH Asia Conference, (SA'11), Hong Kong, China, Association for Computing Machinery (2011).

[39] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D.G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., Zheng, X.: Tensorflow: A system for large-scale machine learning. In: Proceedings of 12th USENIX Symposium on Operating Systems Design and Implementation, (OSDI'2016), Savannah, GA, United states, USENIX Association (2016) 265-283.

[40] Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y.: Spectral normalization for generative adversarial networks. In: Proceedings of 6th International Conference on Learning Representations, (ICLR'2018), Vancouver, BC, Canada, International Conference on Learning Representations, ICLR (2018).

[41] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Proceedings of the 3rd International Conference for Learning Representations, (ICLR'2015), San Diego, CA, United states, Springer Verlag (2015) 1-15.

[42] Agustsson, E., Timofte, R.: Ntire 2017 challenge on single image super-resolution: Dataset and study. In: Proceedings of 30th IEEE Conference on Computer Vision and Pattern Recognition Workshops, (CVPRW'2017), Honolulu, HI, United states, IEEE Computer Society (2017) 1122-1131.

[43] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: Proceedings of 31st Annual Conference on Neural Information Processing Systems, (NIPS'2017), Long Beach, CA, United states, Neural information processing systems foundation (2017).

[44] Binkowski, M., Sutherland, D.J., Arbel, M., Gretton, A.: Demystifying mmd gans. In: Proceedings of 6th International Conference on Learning Representations, (ICLR'2018), Vancouver, BC, Canada, International Conference on Learning Representations, ICLR (2018).

[45] Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization (2016).

[46] Wu, Y., He, K.: Group normalization. CoRR abs/1803.08494 (2018).

6 Appendix

6.1 Network Architecture

Fig.8 shows the architecture of generator network and discriminator network in the proposed DTGAN, in which "k" is the kernel size, "n" is the number of feature maps and "s" is the stride in each convolutional layer, "Conv" represents two-dimensional convolution and "Conv1d" represents one-dimensional convolution, "BN" indicates the batch normalization layer. The generator begins with a flat convolution stage followed by three down-convolution blocks that use convolution layers with stride 2 for down-sample to spatially compress and encode the images. Useful local signals extracted in this stage are downstream connected by the manner of elementwise sum. The encoded features are then fed separately to the two output tails, in which bilinear interpolation layers are used for upsampling to avoid checkerboard artifacts. In the two output tails, the final convolutional layers with 7×7 convolution kernels does not use the normalization layer and is followed by the tanh nonlinear activation function.

PatchGAN [1] is adapted in the two discriminator networks. Each pixel in the output feature map correspond to a patch in the input image, and the patch size equals to 49×49. The discriminators are used to judge whether the patch belongs to anime images or generated images. The discriminator network consists of a flat convolution layer, 3 down-convolution blocks and a point convolution. The two discriminators in DTGAN have exactly the same network architecture. Spectral normalization [40] is applied to all the convolutional layers of the discriminators to enforce Lipschitz constrain on the network and stabilize training.

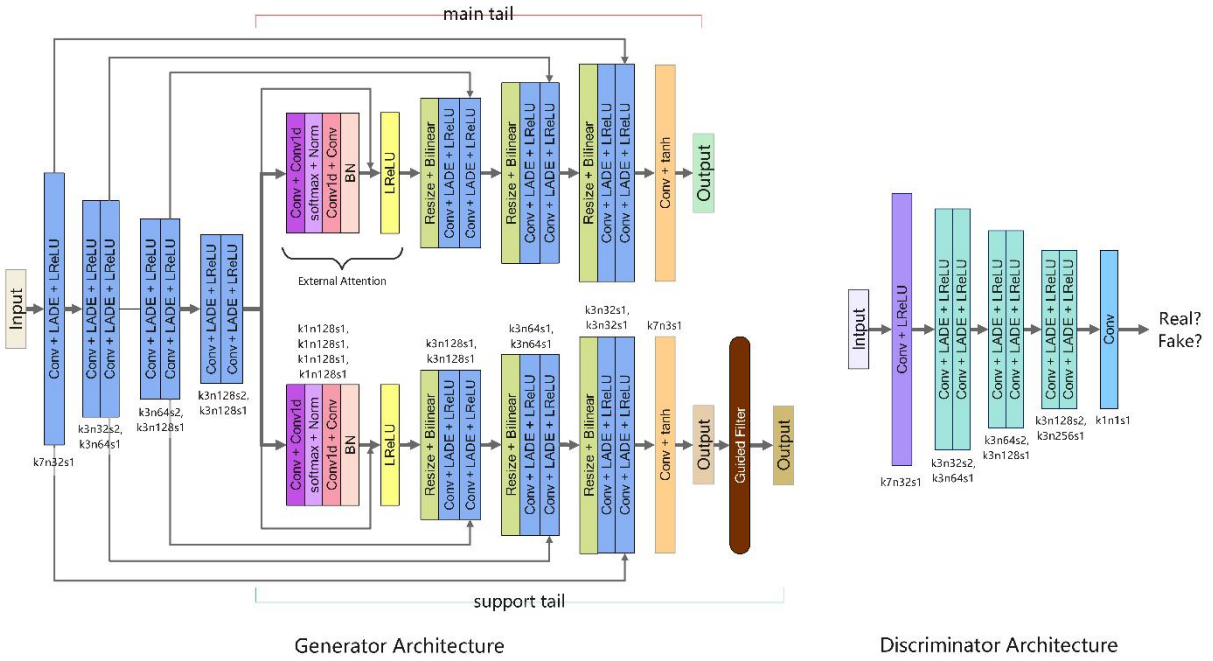

Fig. 8. The architecture of generator network and discriminator network

6.2 Additional experiments

We have introduced two additional evaluating criteria of the image visual quality to compare the proposed methods with other existing methods on the DIV2K dataset [42]. The two evaluation metrics used in our experiments are peak signal to noise ratio (PSNR) and structural similarity (SSIM). Table 4 reports PSNR and SSIM of the different methods. The best results are shown in boldface. Our method outperforms the other methods on two metrics and can generate anime images with higher visual quality.

Table 4. Quantitative comparison results of different methods

Metrics	AdaIN	CartoonGAN	AnimeGAN	AnimeGANv2	White-box	Ours
PSNR	11.555	16.727	19.243	19.500	22.598	**24.160**
SSIM	0.343	0.714	0.774	0.775	0.797	**0.806**

Due to the limitation of space, only a few qualitative comparison results are presented in Section 4.3 of the paper. Here we apply our method on the DIV2K dataset to produce high-resolution anime results and compare with state-of-the-art methods. The qualitative comparison results are shown in Fig.9 and Fig.10. It is observed that the results generated by our method can effectively preserve the color and brightness of the input photos while exhibiting clear edges. AdaIN generates the over-stylized results and CartoonGAN generates the images with significant visual artifacts. AnimeGAN preserves more details of the input photos and weakens the abstract anime style. AnimeGANv2 produces the images with blurred details and edge feathering. White-box causes over-smoothed color and fails to generate clear edges. In comparison, our method generates satisfactory results. To conclude, our method can produce high-quality anime images from real-world photos and outperforms the previous methods.

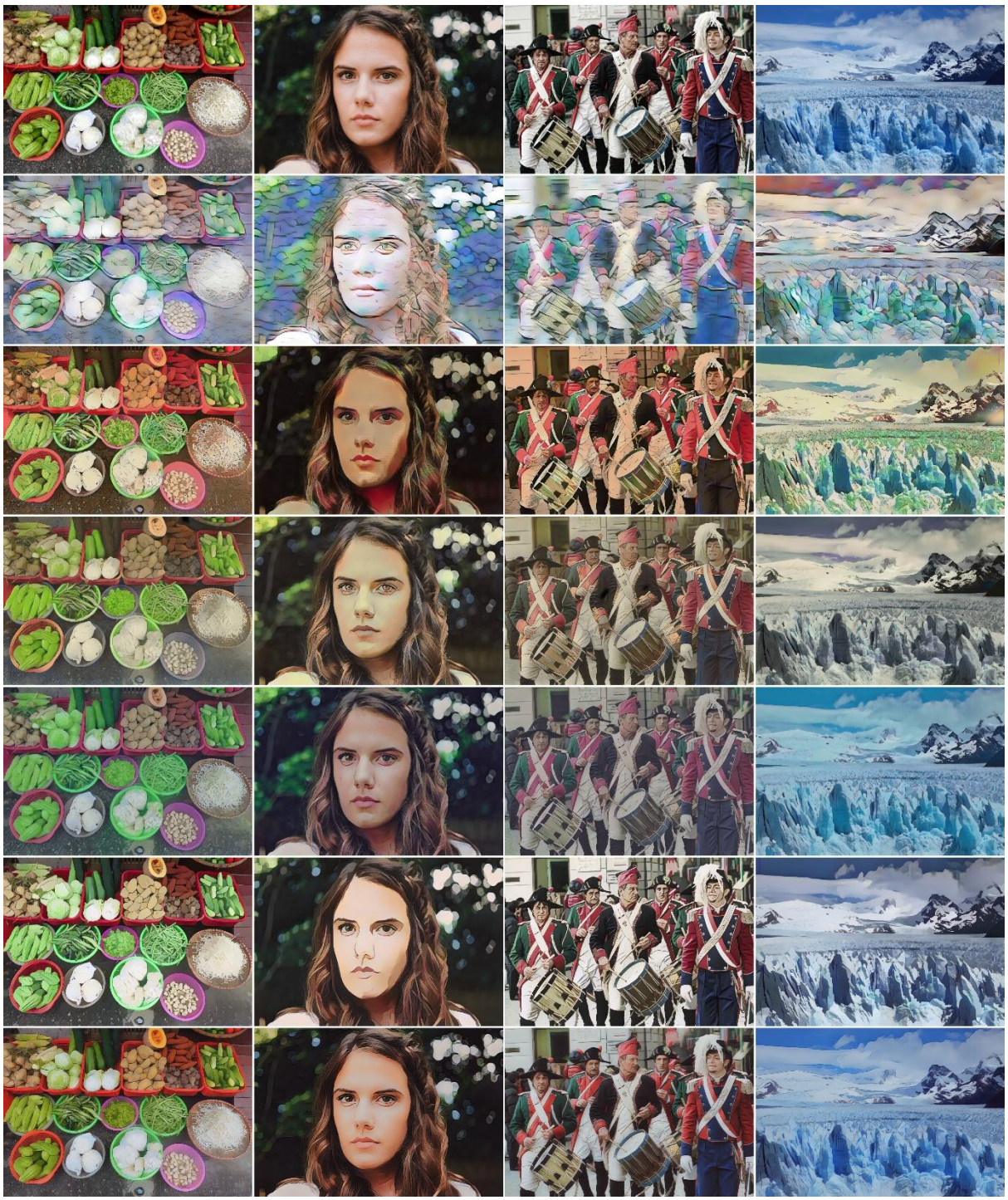

Fig. 9. Qualitative comparison of different methods. The first row to the last row are the input photos, the results of AdaIN, the results of CartoonGAN, the results of AnimeGAN, the results of AnimeGANv2, the results of White-box and the results of DTGAN respectively

Fig. 10. Qualitative comparison of different methods. The first row to the last row are the input photos, the results of AdaIN, the results of CartoonGAN, the results of AnimeGAN, the results of AnimeGANv2, the results of White-box and the results of DTGAN respectively

To further investigate our improvements, we add PSNR and SSIM as new evaluation metrics in the ablation experiments. The comparison results of different normalization methods are presented in Table 5. The best results are shown in boldface. Compared with the other four normalization methods, the proposed LADE has achieved the best results on all metrics. It further indicates that LADE is superior to other 4 normalization methods.

Table 5. Quantitative comparison results of different normalization methods

Metrics	BN	LN	IN	GN	LADE
PSNR	22.642	23.669	18.371	17.081	**24.160**
SSIM	0.679	0.775	0.710	0.664	**0.806**

The comparison results of different loss functions are presented in Table 3. In Table 3, (a) represents the grayscale style loss using RGB color anime images as the input (This loss is also called the color style loss.), (b) represents DTGAN without the fine-grained revision loss function, (c) represents DTGAN without the region smoothing loss function and (d) represents the color reconstruction loss using the YUV color space. Our proposed loss functions and the improved loss functions have achieved the best results on all metrics, which proves that these loss functions have significant effects on the performance of the network and effectively improve the visual quality of the generated images.

Table 6. Quantitative comparison results of different loss functions

Metrics	(a)	(b)	(c)	(d)	ours
PSNR	21.558	19.265	22.693	23.868	**24.160**
SSIM	0.795	0.635	0.744	0.786	**0.806**

6.3 High Definition Results

To further demonstrate the performance of our method, we apply our method to high-resolution photos and display the generated anime results. After training our network with a dataset of a specific anime artist style, the proposed method can generate the anime images of the corresponding style. In the experiment, we train our network using the Hayao Miyazaki style dataset and the Makoto Shinkai style dataset to generate the anime images of the specified style. The two datasets are from AnimeGAN and AnimeGANv2 respectively. The generated anime images in the style of Hayao Miyazaki are shown in Fig.11 to Fig.13, and the generated anime images in the style of Makoto Shinkai are shown in Fig.14 to Fig.16. For different scenes, the anime images generated by our method can effectively preserve the shapes and contours of various objects in the scenes, and have good animation effects. For the characters in the scenes, the animated characters generated by our method better preserve the characteristics of the real characters. Overall, the presented results demonstrate that our method can generate high-quality anime images, and can be applied in different real-world scenes.

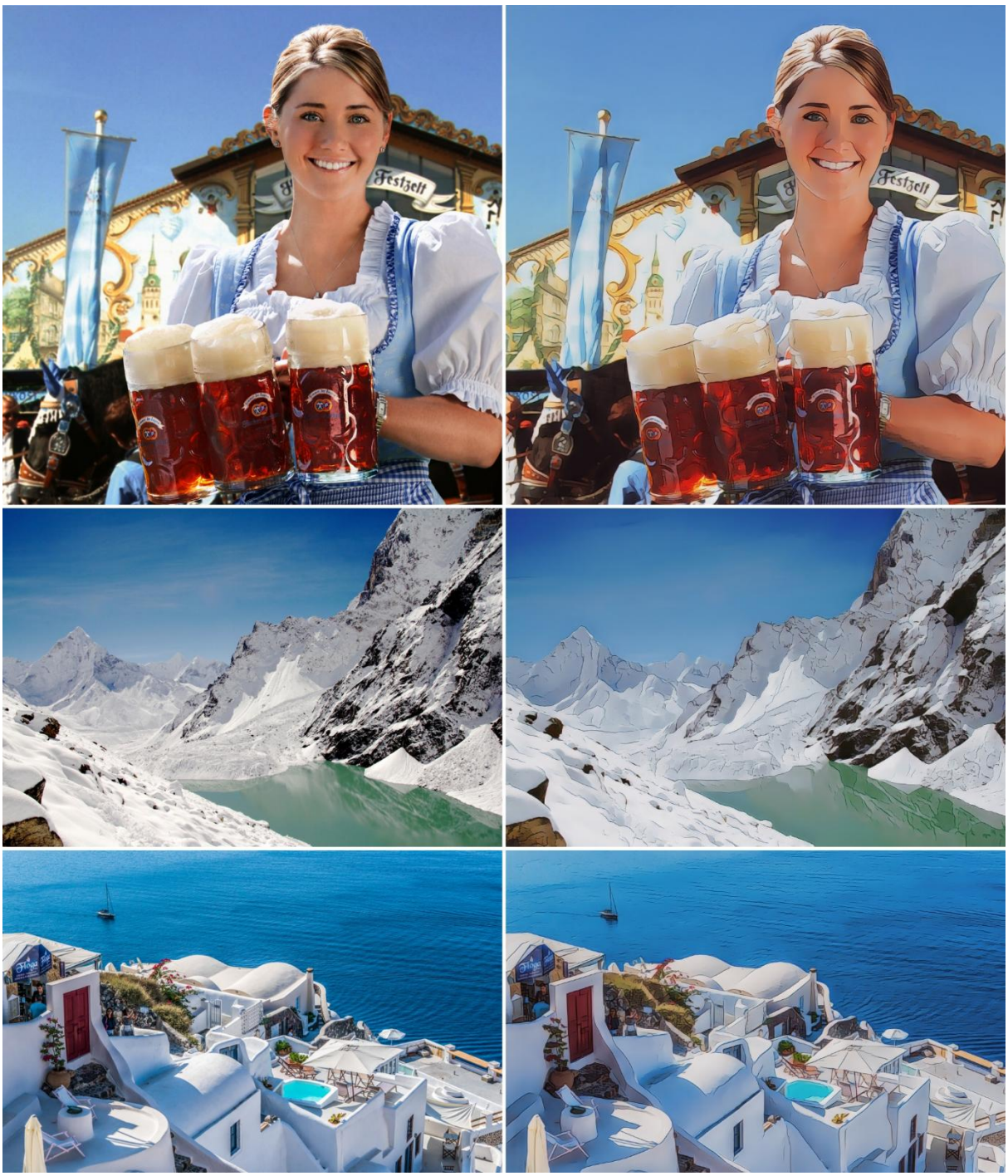

Fig. 11. The input photos (left) and the generated anime images (right) in the style of Hayao Miyazaki

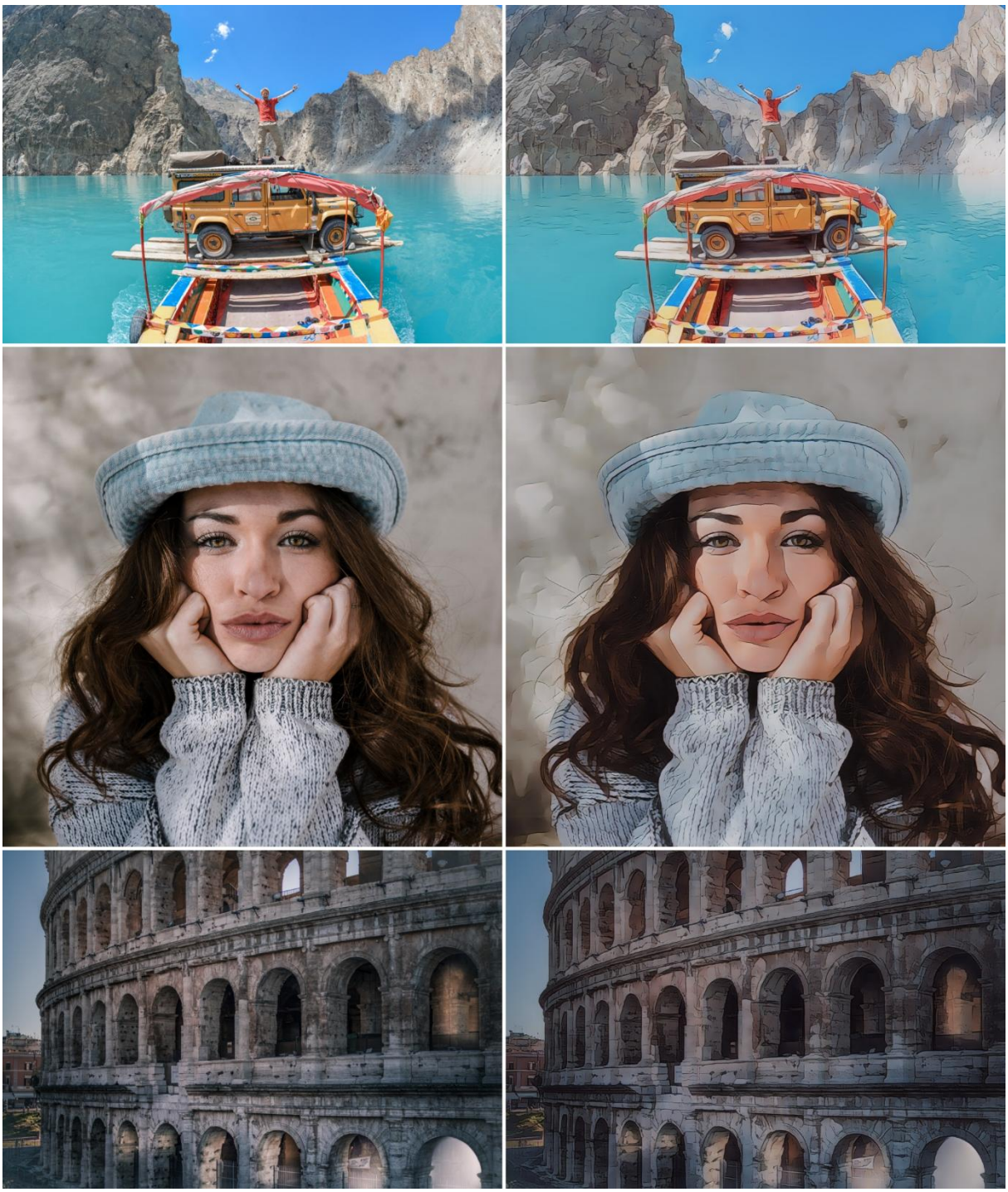

Fig. 12. The input photos (left) and the generated anime images (right) in the style of Hayao Miyazaki

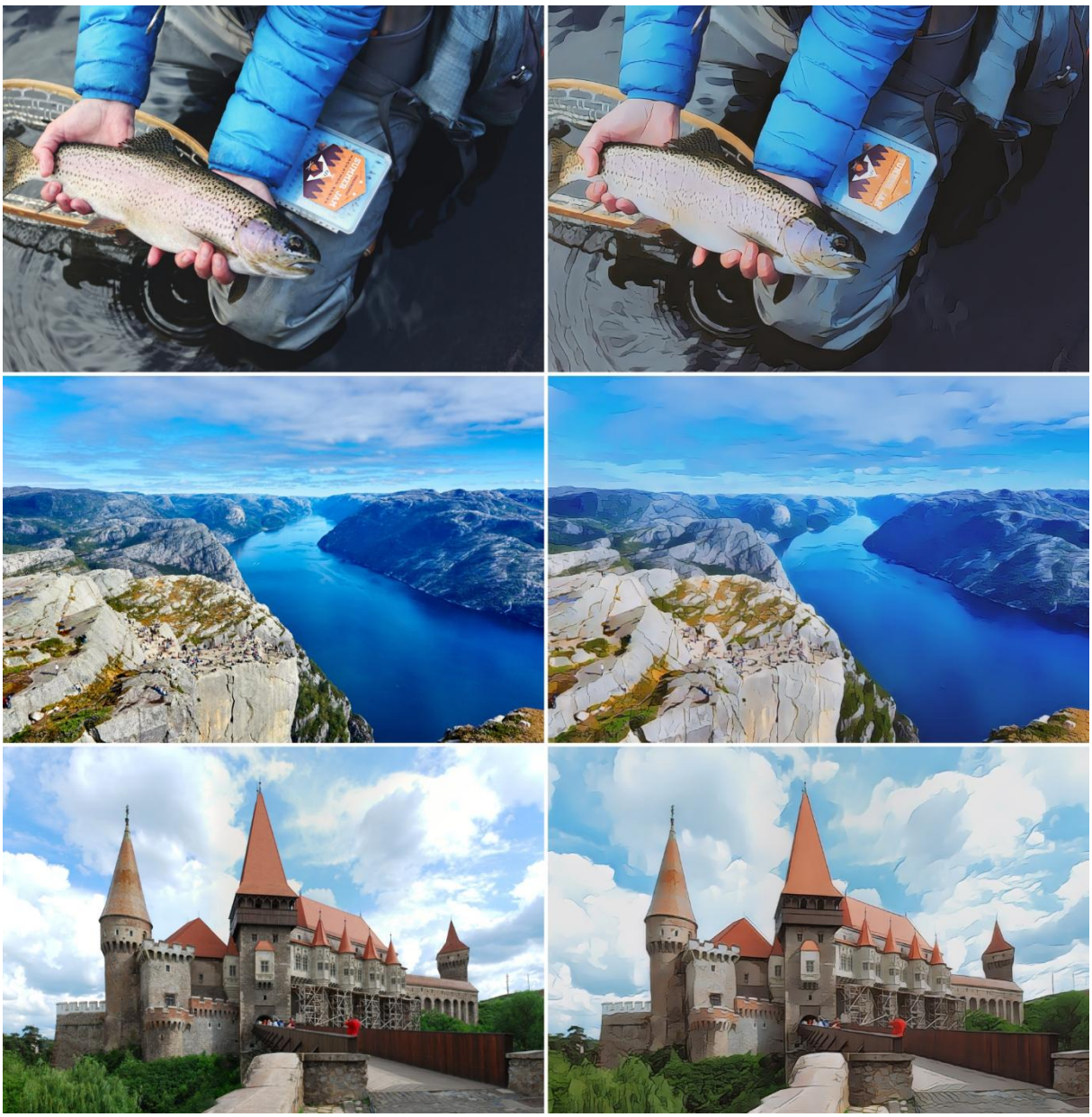

Fig. 13. The input photos (left) and the generated anime images (right) in the style of Hayao Miyazaki

Fig. 14. The input photos (left) and the generated anime images (right) in the style of Makoto Shinkai

Fig. 15. The input photos (left) and the generated anime images (right) in the style of Makoto Shinkai

Fig. 16. The input photos (left) and the generated anime images (right) in the style of Makoto Shinkai

6.4 Inference Speed Comparison

In Section 4.3 of the paper, the inference speed of several state-of-the-art (SOTA) methods and DTGAN are compared. In this section, we give the details of the inference speed comparison. In the experiment, we extracted 300 frames from a 1080p video as the test data, all of which are 1920 × 1080. The environment is the same for all methods. As shown in Fig.17, the vertical axis represents FPS (frames processed per second), and the horizontal axis represents the number of the test data. It can be seen that DTGAN has faster FPS and more stable translation process than other methods.

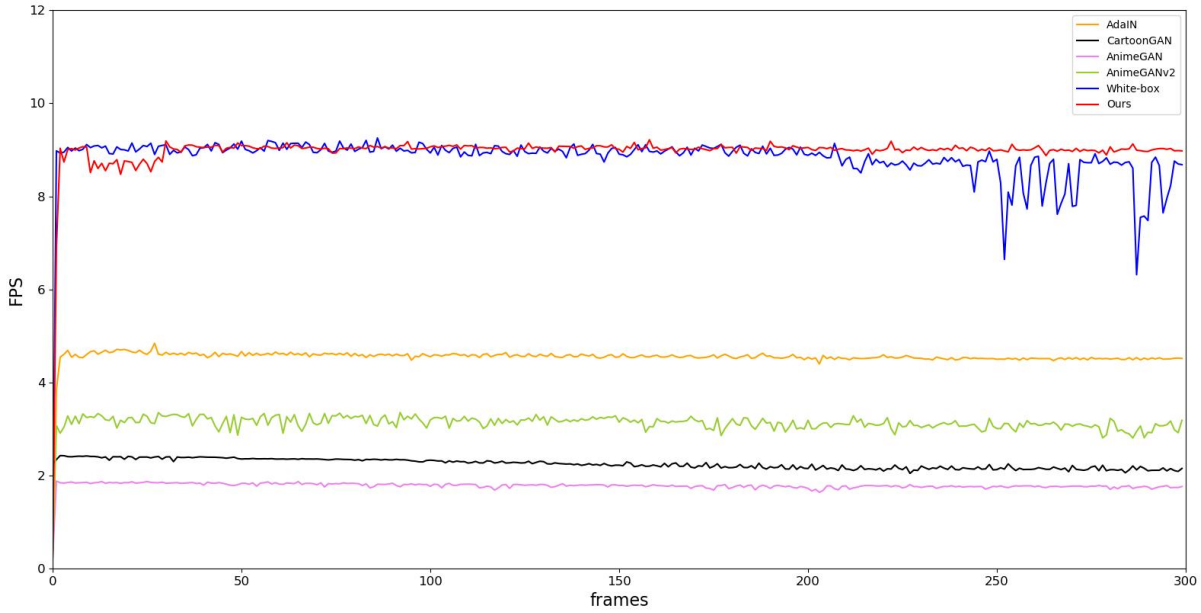

Fig. 17. Inference speed comparison between our method and the SOTA methods on 1080p resolution