Redistributing and Re-Stylizing Features for Training a Fast Photorealistic Stylizer

Chunpeng Wu*, Bin Ni†, and Hai Li*
*Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708, USA
†QUANTIL Inc., Santa Clara, CA 95054, USA
{chunpeng.wu, hai.li}@duke.edu, nibin@quantil.com

***Abstract*—Style transfer studies can be categorized into two types—artistic and photorealistic. The high-speed transfer has been well-studied for artistic styles but remains challenging for photorealistic styles. To guarantee semantic accuracy and style faithfulness, prior photorealistic style transfer techniques often rely on intensive feature matching, hierarchical stylization, and complex auxiliary smoothing. Such high design complexity severely limits the space of transfer speed improvement. In this paper, we propose to accelerate the transfer through a single-level stylization without complex auxiliary smoothing. We design a two-stage "stylization and re-stylization" training pipeline to enhance style faithfulness. The stylization/re-stylization stage consists of two core steps: feature aggregation and redistribution. A new type of layers, *Feature Aggregation (FA)* layers, is proposed to gradually aggregate multi-scale style features into content features at each spatial location. A *Spatially coherent Content-style Preserving (SCP)* loss at feature map level is then used to preserve semantic accuracy. The SCP loss provides effective guidance on redistributing the aggregated features between locations to enforce spatial coherence of style-sensitive content semantic. Experimental results show that compared to previous competitive methods, our method reduces at least 72% run time while achieving better image synthesis quality based on both subjective and objective evaluation metrics. Ablation studies validate the major contribution of our proposed SCP loss and re-stylization to the quality of our synthesized images.**

***Index Terms*—Photorealistic style transfer, re-stylization, feature aggregation, feature redistribution, spatially coherent content-style preserving loss, semantic accuracy, and style faithfulness.**

I. Introduction

Style transfer that denotes the process of modifying the visual characteristics of a content image by referencing a given style image has been widely applied to artwork creation, data augmentation [1], and image/video editing and rendering [2]. The style transfer methods integrate multiple-domain tasks, such as image/feature matching [3] and image/texture synthesis [4].

Style transfer has been studied since the mid-1990s. The related works can be categorized into *artistic* or *photorealistic* style transfer. Comparably, the studies on artistic style transfer have made significantly bigger progress, from the perspective of universality and speed [5]–[9]. These methods are greatly encouraged by style transfer techniques [10], [11] based on convolutional neural network (CNN) and perceptual loss [11]. Moreover, real-time and universal artistic stylizers [7], [12] are proposed recently. Compared to artistic style transfer, the fast photorealistic transfer is a lot more challenging. Prior methods often rely on the intensive feature matching [13], complex initialization [14], hierarchical stylization [15], and complex auxiliary smoothing [13], [15], [16]. The high complexity ensures the semantic accuracy and style faithfulness, while costing high computing resources and processing time.

Theoretically, both artistic and photorealistic style transfer need to match statistics of a style image with that of a content image while keeping semantic accuracy. However, photorealistic style transfer has a more harsh requirement for semantic preservation. For example, AdaIN [7] gets high-quality artistic transfer results in real-time by adopting a simple "scaling+shifting" strategy but results in distorted photorealistic transfer in our initial experiments. A possible reason is that besides the simple "scaling+shifting," higher-order statistics is necessary for more effective semantic preservation. DeepAnalogy [13] uses the patch matching for high-quality artistic transfer. It needs extra time-consuming smoothing (seconds to tens of seconds) for photorealistic transfer. Similarly, DeepPhoto [14] adopts a time-consuming artistic transfer as initialization and extra smoothing for photorealistic transfer. Recent photorealistic style transfer methods, PhotoWCT [15] and FlexSolver [16], propose or adopt time-consuming smoothing strategies to remove intolerable distortions.

In this work, we aim at a learning-based fast photorealistic style transfer technique without hierarchical stylization and complex auxiliary smoothing. We propose a two-stage "stylization and re-stylization" training pipeline to enhance the style faithfulness. The stylization/re-stylization stage consists of two core steps: feature *aggregation* and feature *redistribution*. We design a new type of layers, *Feature Aggregation (FA)* layers, to gradually aggregate multi-scale style features into content features at each spatial location. The FA layers don't contain learnable affine parameters to guarantee transfer universality. The aggregated features are then redistributed between spatial locations on the feature maps. To optimize the redistribution step, we propose a feature-map level *Spatially coherent Content-style Preserving (SCP) loss*, which enforces the spatial coherence of style-sensitive content semantic. Our SCP loss is designed for preserving the semantic accuracy at the feature map level. Our major contributions are:

This work was in part supported by National Science Foundation (NSF) under grant CNS-1822085 and Department of Energy (DOE) under grant SC0017030.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

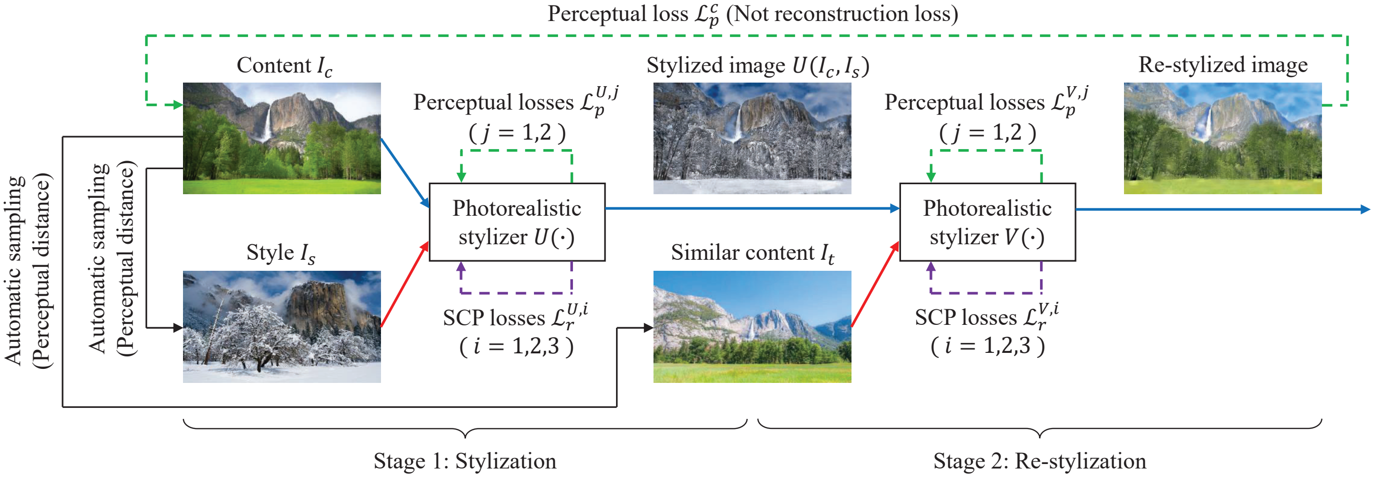
Perceptual loss $\mathcal{L}_p^c$ (Not reconstruction loss)
Content I_c
Stylized image $U(I_c, I_s)$
Re-stylized image
Perceptual losses $\mathcal{L}_p^{U,j}$ ($j = 1,2$)
Perceptual losses $\mathcal{L}_p^{V,j}$ ($j = 1,2$)
Photorealistic stylizer $U(\cdot)$
Photorealistic stylizer $V(\cdot)$
Automatic sampling (Perceptual distance)
Automatic sampling (Perceptual distance)
Style I_s
Similar content I_t
SCP losses $\mathcal{L}_r^{U,i}$ ($i = 1,2,3$)
SCP losses $\mathcal{L}_r^{V,i}$ ($i = 1,2,3$)
Stage 1: Stylization
Stage 2: Re-stylization

Fig. 1: Our proposed training pipeline. All content, style, and similar content images are automatically sampled from the same dataset. The left photorealistic stylizer $U(\cdot)$ is used for inference after training.

- We propose a fast photorealistic style transfer method without hierarchical stylization and complex auxiliary smoothing. To achieve the high speed, We integrate core processing steps into an end-to-end CNN by designing FA layers.
- We propose two training strategies, the SCP loss at feature map level and re-stylization stage, to enhance image synthesis quality. The SCP loss is to preserve semantic accuracy, while the re-stylization helps enhance style faithfulness. Our ablation studies validate the major contribution of our SCP loss and re-stylization to the quality of our synthesized images.
- Experimental results show that compared to previous competitive methods [14], [15], our method reduces at least 72% run time while achieving better image synthesis quality based on both subjective and objective evaluation metrics.

II. Related Work

Artistic and photorealistic style transfers share most techniques. Early studies on style transfer focus on non-photorealistic rendering [17] and texture synthesis [18]. As low-level visual statistics only are used, the capability of capturing and preserving image semantic is limited. So these methods are specifically designed for particular artistic styles. Gatys *et al.* [10] make a breakthrough by gradually matching VGG [19] features extracted from a content image and a style image, respectively. Nowadays style transfer methods can be categorized into image-optimization based [10], [13], [14], [20]–[25] and model-optimization based [5]–[7], [11], [15], [26]–[29]. Model-optimization based methods replace the online optimization process of image-optimization based approaches with a CNN offline training process [30], so the runtime can be dramatically reduced. Various loss functions have been adopted including Gram loss [31], perceptual loss [11], maximum mean discrepancy loss [22], adversarial loss [32], histogram loss [24], hierarchical loss [33], Laplacian loss [23], Markov random field loss [20]. For photorealistic style transfer [14]–[16], typical auxiliary processing is time-consuming smoothing. In this work, we propose a model-optimization based method for fast transfer. Our major difference, compared to previous photorealistic style transfer methods, is that we do not rely on traditional complex auxiliary processing but propose new training strategies, the SCP losses and re-stylization, to enhance image synthesis quality.

Evaluation of image synthesis quality consists of objective and subjective metrics. Typical objective metrics are Inception score [34], FID score [35], SSIM [36], SWD distance [37], and PSNR, and often adopted in image generation and image-to-image translation methods [37]–[44]. Typical subjective metrics are Photorealism [14], Style faithfulness [14], and User preference [6], [15], and often adopted in artistic/photorealistic style transfer methods [6], [14], [15]. The subjective metrics based evaluation is also named as *user studies*. In this work, we use both subjective user studies introduced in [14] and objective metric FID score to evaluate image synthesis quality, instead of subjective user studies only in the previous photo-realistic style transfer methods [14], [15].

III. Proposed Method

Figure 1 shows our training pipeline. An image dataset, $D = \{\boldsymbol{I}_i | i = 1, 2, ..., M\}$, is used to sample content, style, and similar content images. We adopt an automatic image sampling method for content, style and similar content images, which will be described in Section III-F.

The training pipeline consists of two stages: *Stylization* and *Re-stylization*. Assume that currently sampled content, style, and similar content images are denoted as $\boldsymbol{I}_c$, $\boldsymbol{I}_s$, and $\boldsymbol{I}_t$, respectively. First, the content and style images, $\boldsymbol{I}_c$ and $\boldsymbol{I}_s$, are fed into the left photorealistic stylizer $U(\cdot)$ in Figure 1, which generates a stylized image $U(\boldsymbol{I}_c, \boldsymbol{I}_s)$. Together with the similar content image $\boldsymbol{I}_t$, $U(\boldsymbol{I}_c, \boldsymbol{I}_s)$ is then fed into the right photorealistic stylizer $V(\cdot)$ to generate a re-stylized image $V(U(\boldsymbol{I}_c, \boldsymbol{I}_s), \boldsymbol{I}_t)$. Our method uses the similar content image as the "style image" in the re-stylization stage. So compared to

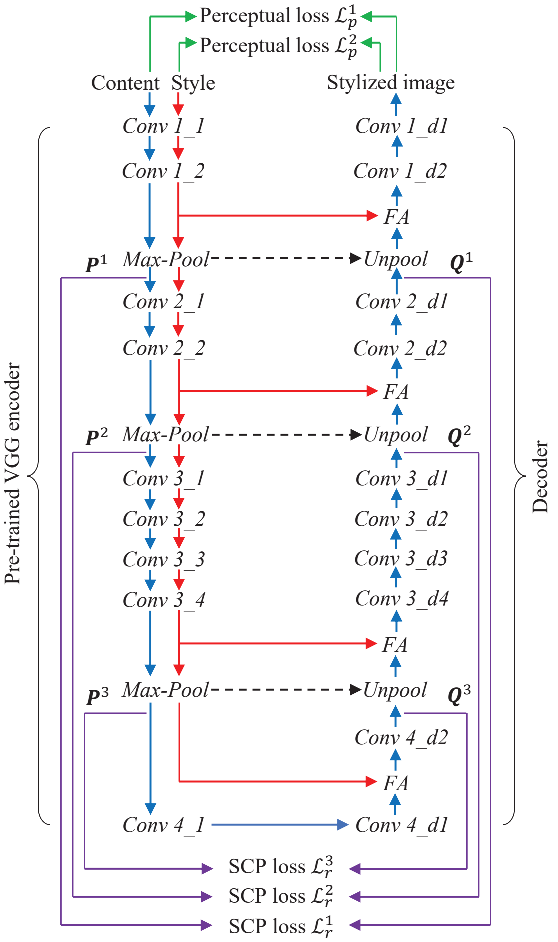

Fig. 2: The architecture of our photorealistic stylizer (best viewed in color). For the right stylizer $V(\cdot)$ shown in Figure 1, *Content*, *Style*, and *Stylized* images here can be replaced by *Stylized image*, *Similar content*, and *Re-stylized* images, respectively. The encoder is fixed during training.

the stylized image, the re-stylized image visually shares more similarity with the content image.

Very important, we aim to train the re-stylized image to be perceptually similar, not pixel-level consistent, to the content image. As shown in Figure 1, two types of losses are used for updating the two stylizers during training: our proposed *SCP* losses and the traditional perceptual losses. The SCP losses, $\mathcal{L}_r^{U,i}$ $(i = 1, 2, 3)$ and $\mathcal{L}_r^{V,i}$ $(i = 1, 2, 3)$, are evaluated between encoder and decoder layers within each stylizer as shown in Figure 2, where i denotes ith pooling layer. The perceptual losses, including $\mathcal{L}_p^{U,j}$ $(j = 1, 2)$, $\mathcal{L}_p^{V,j}$ $(j = 1, 2)$, and $\mathcal{L}_p^c$, are evaluated between input and output images of a stylizer (or two stylizers)[1].

A. Architecture of our photorealistic stylizer

Figure 2 depicts the architecture of our proposed photorealistic stylizer. In most of previous artistic/photorealistic style transfer methods [7], [11], [14], [15], a symmetric encoder-decoder is adopted, while the encoder is comprised of pre-trained VGG layers. We also adopt such symmetric encoder-decoder design, except that our decoder has an extra convolutional layer, *Conv_4_d2*, and a new type of layers, *Feature Aggregation (FA)* layers. The convolutional layer *Conv_4_d2* is to enable feature redistribution introduced in Section III-C. The layer *Conv_4_d2* is fixed to have the same number of output feature maps and size of each convolution kernel as the layer *Conv_4_d1*. The FA layers are to aggregate style features (horizontally red arrow lines) into the decoder. *Unpool* (unpooling) layers are adopted in our decoder to restore spatial information using pooling masks provided by the corresponding *Max-Pool* (max-pooling) layers in the encoder (horizontally black dashed lines with an arrowhead).

B. Feature aggregation in a FA layer

A FA layer has two input: output features $\boldsymbol{F}$ of the previous layer (a vertically blue arrow line in Figure 2) and style features $\boldsymbol{G}$ extracted from the encoder (a horizontally red arrow line in Figure 2). The features $\boldsymbol{F}$ and $\boldsymbol{G}$ have the same number of feature maps and same size of each feature map. The features $\boldsymbol{F}$ is denoted to have N maps and the size of each feature map is $H \times W$. Let a vector $\boldsymbol{x}_m$ be all activations at spatial location m in $\boldsymbol{F}$ where $|\boldsymbol{x}_m| = N$ and $m = 1, 2, ..., H \times W$. Similarly, let a vector $\boldsymbol{y}_m$ be all activations at spatial location m in $\boldsymbol{G}$ where $|\boldsymbol{y}_m| = N$ and $m = 1, 2, ..., H \times W$. The mean vector of all vectors $\{\boldsymbol{y}_m | m = 1, 2, ..., H \times W\}$ is denoted as $\boldsymbol{\mu}_G$. Therefore, the FA layer's output $\boldsymbol{z}_m$ at spatial location m ($|\boldsymbol{z}_m| = N$ and $m = 1, 2, ..., H \times W$) is computed as:

$$\boldsymbol{z}_m = \boldsymbol{x}_m + \frac{1}{H \times W} \sum_{j=1}^{H \times W} (\boldsymbol{x}_m - (\boldsymbol{y}_j - \boldsymbol{\mu}_G)). \tag{1}$$

The physical meaning of Equation (1) is that the averaged difference between the features $\boldsymbol{x}_m$ and the features $\{\boldsymbol{y}_m | m = 1, 2, ..., H \times W\}$ with mean removal is aggregated into $\boldsymbol{x}_m$. Our FA layers have no learnable affine parameters to guarantee transfer universality.

C. Feature redistribution at feature map level

At the end of aggregation, convolutional layers following FA layers (Figure 2) use the aggregated content and style features for image synthesis. We propose a feature-map level SCP loss to provide an effective guidance on redistributing the aggregated information to preserve semantic accuracy.

Our proposed SCP loss is shown in Figure 3. According to our proposed photorealistic stylizer architecture, we evaluate the SCP losses at three pooling layers, $\mathcal{L}_r^1$, $\mathcal{L}_r^2$, and $\mathcal{L}_r^3$, as highlighted in purple arrow lines in Figure 2. For each scale i $(i = 1, 2, 3)$, the SCP loss $\mathcal{L}_r^i$ is evaluated between the output feature maps $\boldsymbol{P}^i$ of the *Max-Pool* layer in the encoder and the input feature maps $\boldsymbol{Q}^i$ of the corresponding *Unpool* layer in the decoder. $\boldsymbol{P}^i$ is content features, while $\boldsymbol{Q}^i$ is aggregated features. Please note that the feature maps in $\boldsymbol{P}^i$ and $\boldsymbol{Q}^i$ have the same number and size. We use $\boldsymbol{W}_r^i$ to represent kernels of

[1] Since the two stylizers $U(\cdot)$ and $V(\cdot)$ have the same architecture, we use $\mathcal{L}_r^i$ to represent $\mathcal{L}_r^{U,i}$ and $\mathcal{L}_r^{V,i}$ in the following sections for convenience whenever possible. Similarly, $\mathcal{L}_p^j$ is used to represent $\mathcal{L}_p^{U,j}$ and $\mathcal{L}_p^{V,j}$ without explicit explanation.

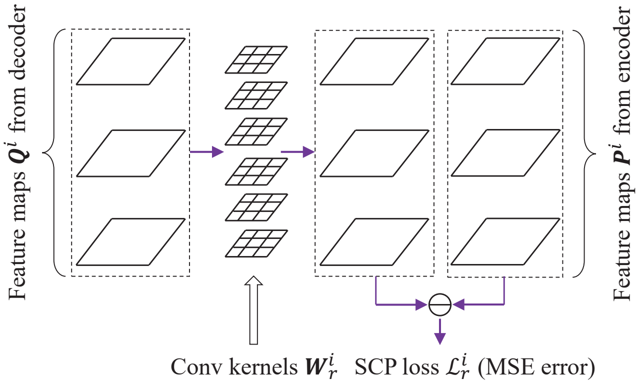
Feature maps $\boldsymbol{Q}^i$ from decoder
Conv kernels $\boldsymbol{W}_r^i$
SCP loss $\mathcal{L}_r^i$ (MSE error)
Feature maps $\boldsymbol{P}^i$ from encoder

Fig. 3: Implementation details of our SCP loss $\mathcal{L}_r^i$ $(i = 1, 2, 3)$.

the convolutional layer in Figure 3. The SCP loss $\mathcal{L}_r^i$ is thus defined as:

$$\mathcal{L}_r^i = ||\boldsymbol{W}_r^i * \boldsymbol{Q}^i - \boldsymbol{P}^i||_2^2. \tag{2}$$

The convolutional layer is to preserve both content and style features during the reconstruction process in Equation (2). If the convolutional layer is not used, our proposed SCP loss in Equation 2 becomes a traditional reconstruction loss: $\mathcal{L}_r^i = ||\boldsymbol{Q}^i - \boldsymbol{P}^i||_2^2$. If so, all style features in $\boldsymbol{Q}^i$ will be discarded during the reconstruction process, since $\boldsymbol{P}^i$ is content features extracted from the encoder shown in Figure 2 (the encoder is fixed during training). Therefore, our proposed SCP loss based on a convolutional layer enforces spatial coherence of style-sensitive content semantic in the reconstruction process. On the contrary, reconstruction in [15] does not simultaneously train content and style together, and reconstruction in [14] do not directly match spatial coherence of content and style but indirectly does it via output map. Based on this difference, our method alleviates spatial arrangement issue and can achieve better image synthesis quality compared to [14], [15], as validated in Section IV-B.

Compared to the traditional perceptual loss [11], our proposed SCP loss has two differences. First, our SCP loss is designed for preserving semantic accuracy at the feature map level, while the perceptual loss is for feature distribution matching at image level. Therefore, our SCP loss has a different optimization goal, and can capture finer-grained (*i.e.*, feature map level) features. Our ablation studies in Section IV-C will validate that our SCP loss is significantly more capable of enhancing quality of our synthesized images, compared to the perceptual loss. Second, our SCP loss integrates both content and style features into one loss function, while the perceptual loss compares a stylized image to a content or style image only. Therefore, our SCP loss can capture joint feature (content-style) changes which are possibly missed by the perceptual loss.

D. Feature matching at image level

The traditional perceptual loss is used in our method to match the feature distribution between input and output images of a stylizer (or two stylizers), as shown in Figure 1. The perceptual loss and our proposed SCP loss have different optimization goals, as stated in Section III-C. Let ϕ_1, ϕ_2, ϕ_3, and ϕ_4 denote pre-trained VGG layers *ReLU 1_1*, *ReLU 2_1*, *ReLU 3_1*, and *ReLU 4_1*. The perceptual loss $\mathcal{L}_p^{U,1}$ evaluated between the content image $\boldsymbol{I}_c$ and the stylized image $U(\boldsymbol{I}_c, \boldsymbol{I}_s)$ is:

$$\mathcal{L}_p^{U,1} = \sum_{i=1}^{4} ||\phi_i(\boldsymbol{I}_c) - \phi_i(U(\boldsymbol{I}_c, \boldsymbol{I}_s))||_2^2. \tag{3}$$

The perceptual loss $\mathcal{L}_p^{U,2}$ evaluated between the style image $\boldsymbol{I}_s$ and the stylized image $U(\boldsymbol{I}_c, \boldsymbol{I}_s)$ is:

$$\mathcal{L}_p^{U,2} = \sum_{i=1}^{4} ||\phi_i(\boldsymbol{I}_s) - \phi_i(U(\boldsymbol{I}_c, \boldsymbol{I}_s))||_2^2. \tag{4}$$

The perceptual loss $\mathcal{L}_p^{V,1}$ evaluated between the stylized image $U(\boldsymbol{I}_c, \boldsymbol{I}_s)$ and the re-stylized image $V(U(\boldsymbol{I}_c, \boldsymbol{I}_s), \boldsymbol{I}_t)$ is:

$$\mathcal{L}_p^{V,1} = \sum_{i=1}^{4} ||\phi_i(U(\boldsymbol{I}_c, \boldsymbol{I}_s)) - \phi_i(V(U(\boldsymbol{I}_c, \boldsymbol{I}_s), \boldsymbol{I}_t))||_2^2. \tag{5}$$

The perceptual loss $\mathcal{L}_p^{V,2}$ evaluated between the similar content image $\boldsymbol{I}_t$ and the re-stylized image $V(U(\boldsymbol{I}_c, \boldsymbol{I}_s), \boldsymbol{I}_t)$ is:

$$\mathcal{L}_p^{V,2} = \sum_{i=1}^{4} ||\phi_i(\boldsymbol{I}_t) - \phi_i(V(U(\boldsymbol{I}_c, \boldsymbol{I}_s), \boldsymbol{I}_t))||_2^2. \tag{6}$$

The perceptual loss $\mathcal{L}_p^c$ evaluated between the content image $\boldsymbol{I}_c$ and the re-stylized image $V(U(\boldsymbol{I}_c, \boldsymbol{I}_s), \boldsymbol{I}_t)$ is:

$$\mathcal{L}_p^c = \sum_{i=1}^{4} ||\phi_i(\boldsymbol{I}_c) - \phi_i(V(U(\boldsymbol{I}_c, \boldsymbol{I}_s), \boldsymbol{I}_t))||_2^2. \tag{7}$$

E. Overall loss function for training our decoder

Put all SCP and perceptual losses together, the overall loss function $\mathcal{L}$ for training our decoder is:

$$\mathcal{L} =(\sum_{i=1}^{3} \mathcal{L}_r^{U,i} + \sum_{i=1}^{3} \mathcal{L}_r^{V,i}) + \beta \cdot (\mathcal{L}_p^c + \sum_{j=1}^{2} \mathcal{L}_p^{U,j} + \sum_{j=1}^{2} \mathcal{L}_p^{V,j}), \tag{8}$$

where β is a scale factor, and both $\mathcal{L}_r^{U,i}$ and $\mathcal{L}_r^{V,i}$ are computed using Equation (2) since our two stylizers $U(\cdot)$ and $V(\cdot)$ have the same architecture. Please refer Figure 1 for all SCP and perceptual losses used in Equation (8).

F. Automatic image sampling

We adopt an automatic image sampling strategy for content, style, and similar content images, as shown in Figure 1. Our strategy is based on the image similarity measured by perceptual distance. For any two images $\boldsymbol{I}_j$ and $\boldsymbol{I}_k$ from dataset D, their perceptual distance $A(\boldsymbol{I}_j, \boldsymbol{I}_k)$ is:

$$A(\boldsymbol{I}_j, \boldsymbol{I}_k) = \sum_{i=1}^{4} ||\phi_i(\boldsymbol{I}_j) - \phi_i(\boldsymbol{I}_k)||_2^2. \tag{9}$$

The smaller the perceptual distance is, the more perceptually similarity the two images have. Before training, we compute and store the perceptual distances between all image pairs, *i.e.*, $A(\boldsymbol{I}_j, \boldsymbol{I}_k)$ $(\forall j \in [1, M], \forall k \in [1, M])$.

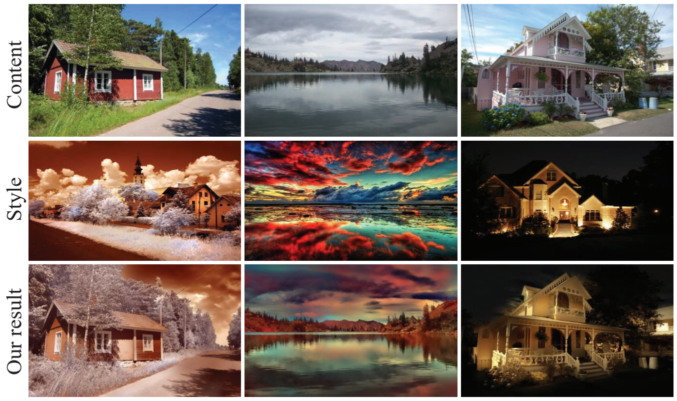

Fig. 4: Our synthesized images. Vertically, each group has three images: a content image, a style image, and our synthesized image.

In each training epoch, first an image randomly sampled from the dataset D is taken as the current content image $\boldsymbol{I}_c$. The similar content image $\boldsymbol{I}_t$ is randomly sampled from an image subset $\{\boldsymbol{I}_w^c\}$ which satisfies:

$$\begin{cases} \boldsymbol{I}_w^c \in D \\ O(\boldsymbol{I}_w^c) = O(\boldsymbol{I}_c) \\ A(\boldsymbol{I}_w^c, \boldsymbol{I}_c) \leqslant T_1^c \end{cases}, \tag{10}$$

where the function $O(\cdot)$ is to get class label of the image, and T_1^c is a pre-defined threshold. According to Equation (10), the similar content image $\boldsymbol{I}_t$ shall have the same class label as the content image $\boldsymbol{I}_c$ and have a high perceptual similarity with $\boldsymbol{I}_c$. The style image $\boldsymbol{I}_s$ is randomly sampled from an image subset $\{\boldsymbol{I}_\gamma^c\}$ which satisfies:

$$\begin{cases} \boldsymbol{I}_\gamma^c \in D \\ A(\boldsymbol{I}_\gamma^c, \boldsymbol{I}_c) > T_1^c, & if\ O(\boldsymbol{I}_\gamma^c) = O(\boldsymbol{I}_c) \\ A(\boldsymbol{I}_\gamma^c, \boldsymbol{I}_c) \leqslant T_2^c, & if\ O(\boldsymbol{I}_\gamma^c) \neq O(\boldsymbol{I}_c) \end{cases}, \tag{11}$$

where T_2^c is a pre-defined threshold. According to Equation (11), the style image $\boldsymbol{I}_s$ can be selected from: the images in the same class except those in $\{\boldsymbol{I}_w^c\}$, or the images that belong to other classes but are perceptually similar with the content image $\boldsymbol{I}_c$ to a large extent.

IV. Experiments

We used MS-COCO 2014 training dataset [45] as our training dataset D by following the recent photorealistic style transfer method [15]. All of our parameters are tuned on MS-COCO 2014 training dataset only.

Given a content image $\boldsymbol{I}_c$ randomly selected from D, its threshold T_1^c defined in Equation (10) is configured to ensure that $\boldsymbol{I}_c$'s subset $\{\boldsymbol{I}_w^c\}$ includes the images with top 50% perceptual similarity in the same class of $\boldsymbol{I}_c$. Similarly, $\boldsymbol{I}_c$'s threshold T_2^c used in Equation (11) is configured to ensure that the images with top 10% perceptual similarity outside $\boldsymbol{I}_c$'s class are included into $\boldsymbol{I}_c$'s subset $\{\boldsymbol{I}_\gamma^c\}$. The subset $\{\boldsymbol{I}_\gamma^c\}$ also includes the other 50% images in the same class of $\boldsymbol{I}_c$. In other words, T_1^c is fixed to "top 50% inside" and T_2^c is fixed to "top 10% outside."

TABLE I: Run time (in seconds) comparison between previous photorealistic style transfer methods and ours. NVIDIA Titan X Pascal is adopted.

Method[1]	512x256	768x384	1024x512
DeepPhoto [14]	186.52	380.82	650.45
PhotoWCT [15]	2.95	7.05	13.16
Ours	**0.73**	**1.95**	**3.24**

[1] Recently, Puy *et al.* [16] report their run time using NVIDIA Tesla P-100 Pascal, however, they do not provide source code. Their run time is 1.86s (512x256), - (768x384), and 8.11s (1024x512). Since speed-performance gap between NVIDIA Titan X Pascal (Single-precision: 11.0 teraFLOPS) and NVIDIA Tesla P-100 Pascal (Single-precision: 10.6 teraFLOPS) is small, ours is also faster than Puy *et al.* [16].

TABLE II: Quantitative comparison of image synthesis quality between previous photorealistic style transfer methods and ours. The testing dataset is MS-COCO 2014 validation dataset. Higher Photorealism (or Style faithfulness) score means higher image quality, while lower FID score means higher image quality.

Method	Subjective user studies in [14]		Objective
	Photorealism	Style faithfulness	FID
DeepPhoto [14]	2.82±0.30	19.1%	171.96
PhotoWCT [15]	3.26±0.20	38.1%	169.12
Ours	**3.40±0.20**	**42.8%**	**167.04**

All convolutional layers in the SCP losses $\mathcal{L}_r^i$ $(i = 1, 2, 3)$ (refer Equation (2)) share the same settings. The convolutional kernel size of $\boldsymbol{W}_r^i$ $(i = 1, 2, 3)$ is fixed to 3×3. Our proposed FA layers have no learnable parameters.

The scale factor β in Equation (8) is set to 1. We use Adam [46] optimization for both photorealistic stylizers in Figure 1. Adam parameters are $\beta_1 = 0.5$ and $\beta_2 = 0.999$. The initial learning rate is set to 0.0001.

Figure 4 shows some synthesized images by our method.

A. Comparison of run time

Table I compares the averaged run time of the previous photorealistic style transfer methods [14], [15] and ours. We follow the settings of run time experiments in PhotoWCT [15]: Using NVIDIA Titan X Pascal and three image resolutions (512x256, 768x384 and 1024x512). DeepPhoto [14] is inherently slower mainly due to the initial value optimization from [10]. Ours is the fastest and reduces at least 72% running time of PhotoWCT [15]. Our advantage comes from the use of an end-to-end CNN without hierarchical stylization and complex auxiliary smoothing adopted in PhotoWCT [15].

B. Comparison of image synthesis quality

As stated in Section II, we adopt both subjective user studies introduced in [14] and objective metric FID score to evaluate the image synthesis quality. The user studies [14] consists of two subjective metrics: Photorealism and Style faithfulness scores. Higher Photorealism (or Style faithfulness) score indicates higher image quality, while lower FID score means higher image quality. The testing dataset is MS-COCO 2014 validation dataset. Please note that the MS-COCO 2014

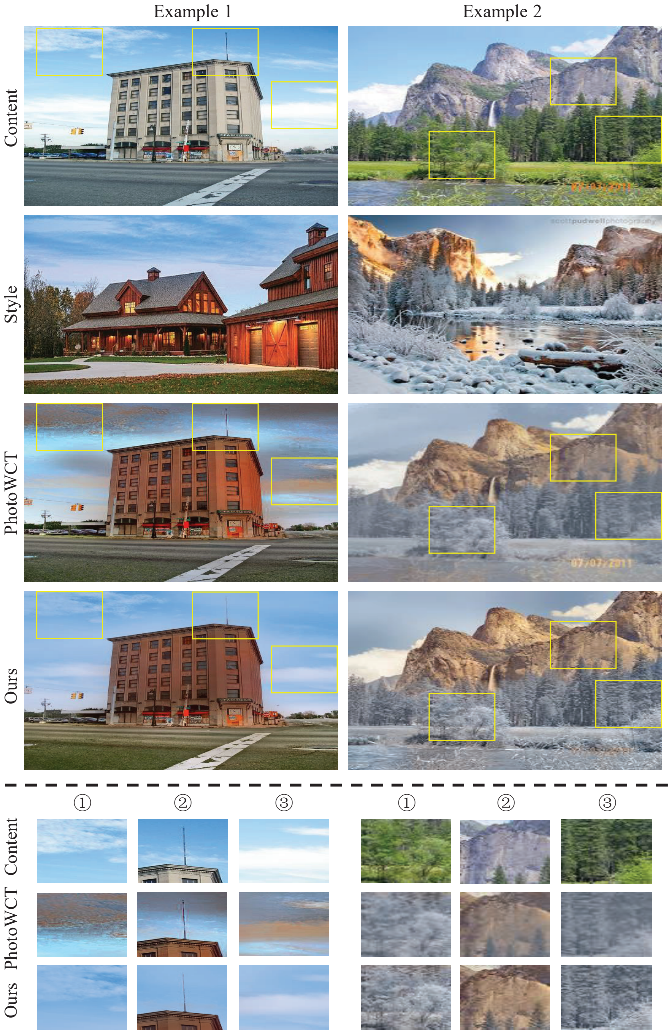

Fig. 5: Comparison between PhotoWCT [15] and ours: Image level (above the black dashed line) and patch level (below the black dashed line). For each example, patches (yellow rectangles) corresponding to three locations (①, ②, and ③) are selected for comparison.

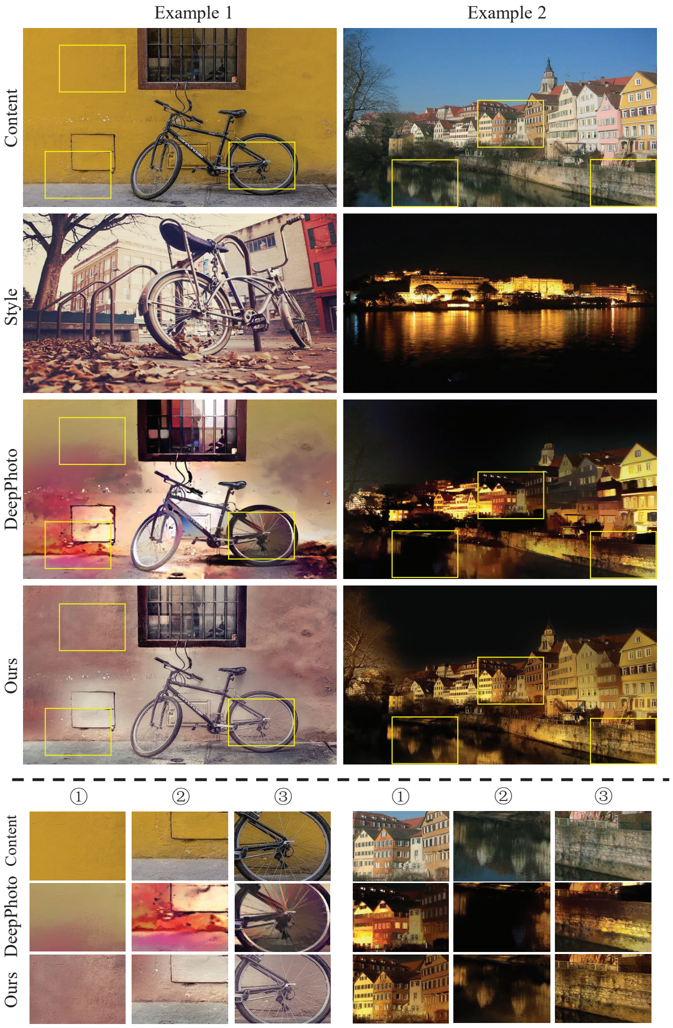

Fig. 6: Comparison between DeepPhoto [14] and ours: Image level (above the black dashed line) and patch level (below the black dashed line). For each example, patches (yellow rectangles) corresponding to three locations (①, ②, and ③) are selected for comparison.

validation dataset and FID score are not used during our training process.

Table II quantitatively compares the image synthesis quality between previous methods [14], [15] and ours. Our method achieves the best image synthesis quality on both subjective metrics (Photorealism and Style faithfulness) and objective metric (FID).

Figure 5 shows two examples of comparison between PhotoWCT [15] and ours. To clearly show our advantage, PhotoWCT and our method are compared at both image level (above the black dashed line) and patch level (below the black dashed line) in Figure 5. In *Example 1*, our synthesized image has fewer intolerable artifacts in the sky and cloud areas of all three patches. In *Example 2*, the tree leaves (patches ① and ③) and mountain rocks (patch ②) are less blurred in our synthesized image.

Figure 6, similarly to Figure 5, shows two examples of comparison between previous method DeepPhoto [14] and ours at both image level and patch level. In *Example 1*, the "dark red" style is more globally smooth in our synthesized image, *e.g.*, patch ①. On the contrary, DeepPhoto's synthesized image has uneven color texture at the bottom left corner of the wall (patch ②) and unexpected color texture on the bicycle spokes (patch ③). In *Example 2*, the semantic details are preserved better in our synthesized image. In contrast, DeepPhoto's synthesized image shows the blurred buildings (patch ①), incomplete reflection of buildings in the river (patch ②), and unexpected "dark" texture in the upper part of the river bank (patch ③).

An explanation for DeepPhoto's worst quantitative synthesis quality in Table II is that DeepPhoto's synthesized images have more uneven/unexpected color texture and incomplete details, compared to PhotoWCT and ours.

TABLE III: Our quantitative ablation studies on the perceptual losses, re-stylization, and SCP losses. The testing dataset is MS-COCO 2014 validation dataset. Lower FID score means higher image quality.

Method	FID score
Ours without perceptual losses	177.14
Ours without re-stylization	179.79
Ours without SCP losses	186.36
Ours (All components)	**167.04**

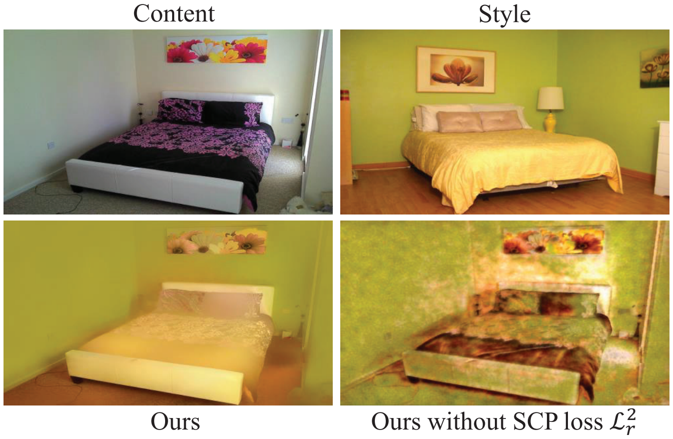

Fig. 7: Ablation study: *Ours* vs. *Ours without SCP loss* $\mathcal{L}_r^2$. Our SCP losses are the most crucial components for guaranteeing image synthesis quality as shown in Table III.

C. Ablation studies and sensitivity analysis

Similarly to Section IV-B, the MS-COCO 2014 validation dataset and FID score are adopted for the quantitative ablation studies and sensitivity analysis. Note that the MS-COCO 2014 validation dataset and FID score are not adopted for our parameter optimization during our training process.

Table III shows our quantitative ablation studies on the perceptual losses, re-stylization, and SCP losses. Removing our proposed SCP losses results in the biggest increase of FID score, *i.e.*, the biggest drop of image synthesis quality, since lower FID score means higher image quality. Our proposed SCP losses are thus the most crucial components on guaranteeing quality of our synthesized images. Similarly, our proposed re-stylization is more important than the traditional perceptual losses, as shown in Table III.

Figure 7 and Figure 8 respectively show the contribution of our proposed SCP losses and re-stylization to image synthesis quality. Quality of the synthesized image *Ours without SCP loss* $\mathcal{L}_r^2$, as shown in Figure 7, is significantly inferior. This result further validates the crucial contribution of our SCP losses. For the synthesized image *Ours without re-stylization*, as shown in Figure 8, the style faithfulness is severely degraded, *e.g.*, the cloud and colors of mountain are lost. As can be seen, the re-stylization stage improves style faithfulness.

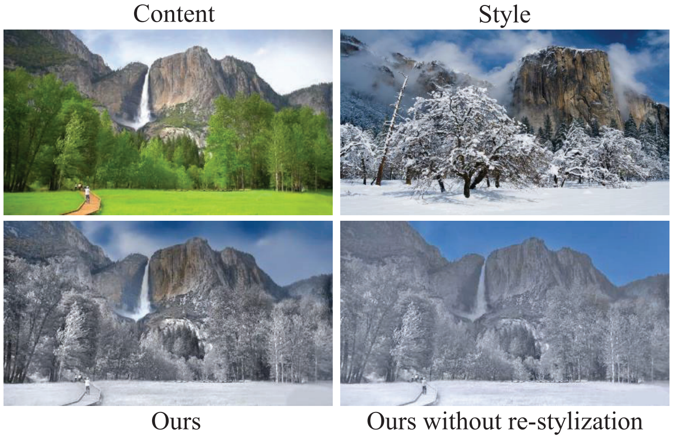

Fig. 8: Ablation study: *Ours* vs. *Ours without re-stylization*. Our proposed re-stylization improves style faithfulness.

TABLE IV: Our sensitivity analysis on the parameters of automatic image sampling, *i.e.*, T_1^c in Equation (10) and T_2^c in Equation (11). For our current parameter settings, T_1^c is fixed to "top 50% inside" and T_2^c is fixed to "top 10% outside." The testing dataset is MS-COCO 2014 validation dataset. Lower FID score means higher image quality.

Method[1][2]	FID score
Ours (T_1^c: top 44% inside, T_2^c: top 10% outside)	167.20
Ours (T_1^c: top 47% inside, T_2^c: top 10% outside)	**167.04**
Ours (T_1^c: top 50% inside, T_2^c: top 10% outside)	**167.04**
Ours (T_1^c: top 53% inside, T_2^c: top 10% outside)	**167.04**
Ours (T_1^c: top 56% inside, T_2^c: top 10% outside)	167.12
Ours (T_1^c: top 50% inside, T_2^c: top 6% outside)	167.36
Ours (T_1^c: top 50% inside, T_2^c: top 8% outside)	167.17
Ours (T_1^c: top 50% inside, T_2^c: top 10% outside)	**167.04**
Ours (T_1^c: top 50% inside, T_2^c: top 12% outside)	**167.04**
Ours (T_1^c: top 50% inside, T_2^c: top 14% outside)	167.09

[1] For T_1^c, "top *% inside" means that images with top *% perceptual similarity in the same class of image $\boldsymbol{I}_c$ are selected into $\boldsymbol{I}_c$'s subset $\{\boldsymbol{I}_w^c\}$.

[2] For T_2^c, "top *% outside" means that images with top *% perceptual similarity outside image $\boldsymbol{I}_c$'s class are selected into $\boldsymbol{I}_c$'s subset $\{\boldsymbol{I}_\gamma^c\}$. Please note that the subset $\{\boldsymbol{I}_\gamma^c\}$ also includes images which are in the same class of image $\boldsymbol{I}_c$ but not in the subset $\{\boldsymbol{I}_w^c\}$.

Table IV shows our sensitivity analysis on the parameters of automatic image sampling, *i.e.*, T_1^c in Equation (10) and T_2^c in Equation (11). The first group in Table IV shows influence of the parameter T_1^c to FID score when the parameter T_2^c is fixed to our current value "top 10% outside." For the first group, the FID score achieves the lowest score, *i.e.*, the best image synthesis quality, when T_1^c is set to "top 47% inside," "top 50% inside," and "top 53% inside." The second group in Table IV shows influence of the parameter T_2^c to FID score when the parameter T_1^c is fixed to our current value "top 50% inside." For the second group, the FID score achieves the lowest score, *i.e.*, the best image synthesis quality, when T_2^c is set to "top 10% outside" and "top 12% outside."

V. Conclusion

We propose a fast photorealistic style transfer method without sacrificing image synthesis quality. In our method, we propose new training strategies, the SCP loss and re-stylization, to enhance image synthesis quality, without adopting time-consuming hierarchical stylization and complex auxiliary smoothing in traditional methods. In addition, our proposed components, the FA layer, SCP loss, and re-stylization, can be easily integrated with other image synthesis techniques to build more advanced photorealistic style transfer models.

References

[1] P. T. Jackson, A. Atapour-Abarghouei, S. Bonner, T. Breckon, and B. Obara, "Style Augmentation: Data Augmentation via Style Randomization," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshop*, pp. 83–92, 2018.

[2] A. Semmo, T. Isenberg, and J. Dollner, "Neural Style Transfer: A Paradigm Shift for Image-based Artistic Rendering?" *International Non-Photorealistic Animation and Rendering Symposium (NPAR)*, pp. 1–13, 2017.

[3] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing," *ACM SIGGRAPH*, pp. 1–10, 2009.

[4] M. Elad and P. Milanfar, "Style Transfer via Texture Synthesis," *IEEE Transactions on Image Processing*, vol. 26, no. 5, pp. 2338–2351, 2017.

[5] T. Q. Chen and M. Schmidt, "Fast Patch-based Style Transfer of Arbitrary Style," *Conference on Neural Information Processing Systems (NeurIPS) Workshop*, pp. 1–5, 2016.

[6] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M.-H. Yang, "Universal Style Transfer via Feature Transforms," *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 1–11, 2017.

[7] X. Huang and S. Belongie, "Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization," *IEEE Conference on Computer Vision (ICCV)*, pp. 1501–1510, 2017.

[8] G. Ghiasi, H. Lee, M. Kudlur, V. Dumoulin, and J. Shlens, "Exploring the Structure of a Real-Time, Arbitrary Neural Artistic Stylization Network," *The British Machine Vision Conference (BMVC)*, 2017.

[9] H. Zhang and K. Dana, "Multi-Style Generative Network for Real-time Transfer," *arxiv preprint arXiv:1703.06953v2*, pp. 1–10, 2017.

[10] L. A. Gatys, A. S. Ecker, and M. Bethge, "A Neural Algorithm of Artistic Style," *arXiv preprint arXiv:1508.06576v2*, pp. 1–16, 2015.

[11] J. Johnson, A. Alahi, and F.-F. Li, "Perceptual Losses for Real-Time Style Transfer and Super-Resolution," *European Conference on Computer Vision (ECCV)*, pp. 694–711, 2016.

[12] H. Huang, H. Wang, W. Luo, L. Ma, W. Jiang, X. Zhu, Z. Li, and W. Liu, "Real-Time Neural Style Transfer for Videos," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 783–791, 2017.

[13] J. Liao, Y. Yao, L. Yuan, G. Hua, and S. B. Kang, "Visual Attribute Transfer through Deep Image Analogy," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–15, 2017.

[14] F. Luan, S. Paris, E. Shechtman, and K. Bala, "Deep Photo Style Transfer," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4990–4998, 2017.

[15] Y. Li, M.-Y. Liu, X. Li, M.-H. Yang, and J. Kautz, "A Closed-Form Solution to Photorealistic Image Stylization," *European Conference on Computer Vision (ECCV)*, pp. 468–483, 2018.

[16] G. Puy and P. Perez, "A Flexible Convolutional Solver for Fast Style Transfer," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8963–8972, 2019.

[17] T. Strothotte and S. Schlechtweg, "Non-Photorealistic Computer Graphics: Modeling, Rendering, and Animation," *Morgan Kaufmann*, pp. 1–496, 2002.

[18] A. A. Efros and W. T. Freeman, "Image Quilting for Texture Synthesis and Transfer," *ACM SIGGRAPH*, pp. 341–346, 2001.

[19] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *International Conference on Learning Representations (ICLR)*, pp. 1–10, 2015.

[20] C. Li and M. Wand, "Combining Markov Random Fields and Convolutional Neural Networks for Image Synthesis," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2479–2486, 2016.

[21] Y.-L. Chen and C.-T. Hsu, "Towards Deep Style Transfer: A Content-Aware Perspective," *British Machine Vision Conference (BMVC)*, pp. 1–11, 2016.

[22] Y. Li, N. Wang, J. Liu, and X. Hou, "Demystifying Neurl Style Transfer," *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2230–2236, 2017.

[23] S. Li, X. Xu, L. Nie, and T.-S. Chua, "Laplacian-Steered Neural Style Transfer," *ACM International Conference on Multimedia (MM)*, 2017.

[24] E. Risser, P. Wilmot, and C. Barnes, "Stable and Controllable Neural Texture Synthesis and Style Transfer Using Histogram Losses," *arxiv preprint arxiv:1701.08893v2*, pp. 1–14, 2017.

[25] R. Mechrez, I. Talmi, and L. Zelnik-Manor, "The Contextual Loss for Image Transformation with Non-Aligned Data," *European Conference on Computer Vision (ECCV)*, pp. 800–815, 2018.

[26] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. Lempitsky, "Texture Networks: Feed-Forward Synthesis of Textures and Stylized Images," *IEEE Conference on Machine Learning (ICML)*, pp. 1349–1357, 2016.

[27] D. Chen, L. Yuan, J. Liao, N. Yu, and G. Hua, "Stylebank: An Explicit Representation for Neural Image Style Transfer," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[28] A. Gupta, J. Johnson, A. Alahi, and F.-F. Li, "Characterizing and Improving Stability in Neural Style Transfer," *IEEE Conference on Computer Vision (ICCV)*, pp. 4067–4076, 2017.

[29] V. Dumoulin, J. Shlens, and M. Kudlur, "A Learned Representation for Artistic Style," *International Conference on Learning Representations (ICLR)*, pp. 1–11, 2017.

[30] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song, "Neural Style Transfer: A Review," *arxiv preprint arXiv:1705.04058v7*, 2018.

[31] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image Style Transfer Using Convolutional Neural Network," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2414–2423, 2016.

[32] C. Li and M. Wand, "Precomputed Real-Time Texture Synthesis with Markovian Generative Adversarial Networks," *European Conference on Computer Vision (ECCV)*, pp. 702–716, 2018.

[33] X. Wang, G. Oxholm, D. Zhang, and Y.-F. Wang, "Multimodal Transfer: A Hierarchical Deep Convolutional Neural Network for Fast Artistic Style Transfer," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5239–5247, 2017.

[34] T. Salimans and D. P. Kingma, "Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks," *Conference on Neural Information Processing Systems (NeurIPS)*, 2016.

[35] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium," *Conference on Neural Information Processing Systems (NeurIPS)*, pp. 6626–6637, 2017.

[36] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity," *IEEE Transactions on Image Processing (TIP)*, vol. 13, no. 4, pp. 600–612, 2004.

[37] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive Growing of GANs for Improved Quality, Stability, and Variation," *International Conference on Learning Representations (ICLR)*, pp. 1–12, 2018.

[38] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved Techniques for Training GANs," *Conference on Neural Information Processing Systems (NeurIPS)*, 2016.

[39] A. Odena, C. Olah, and J. Shlens, "Conditional Image Synthesis with Auxiliary Classifier GANs," *IEEE Conference on Machine Learning (ICML)*, pp. 2642–2651, 2017.

[40] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral Normalization for Generative Adversarial Network," *International Conference on Learning Representations (ICLR)*, pp. 1–14, 2018.

[41] A. Noguchi and T. Harada, "Image Generation From Small Datasets via Batch Statistics Adaptation," *IEEE Conference on Computer Vision (ICCV)*, pp. 2750–2758, 2019.

[42] A. Brock, J. Donahue, and K. Simonyan, "Large Scale GAN Training for High Fidelity Natural Image Synthesis," *International Conference on Learning Representations (ICLR)*, pp. 1–29, 2019.

[43] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal Unsupervised Image-to-Image Translation," *European Conference on Computer Vision (ECCV)*, pp. 179–196, 2018.

[44] T. Karras, S. Laine, and T. Aila, "A Style-based Generator Architecture for Generative Adversarial Networks," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4401–4410, 2019.

[45] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," *European Conference on Computer Vision (ECCV)*, pp. 740–755, 2014.

[46] D. P. Kingma and J. L. Ba, "Adam: A Method for Stochastic Optimization," *International Conference on Learning Representations (ICLR)*, pp. 1–11, 2015.