UPST-NeRF: Universal Photorealistic Style Transfer of Neural Radiance Fields for 3D Scene

Yaosen Chen[1], Qi Yuan[1], Zhiqiang Li[1], Yuegen Liu[1,3], Wei Wang[*1,2], Chaoping Xie[1,2], Xuming Wen[1,2], and Qien Yu[4]

[1]Media Intelligence Laboratory, ChengDu Sobey Digital Technology Co., Ltd
[2]Peng Cheng Laboratory
[3]Southwest Jiaotong University
[4]Sichuan University

{chenyaosen, yuanqi, lizhiqiang, liuyuegen, wangwei, xiechaoping, wenxuming}@sobey.com
yuqien@scu.edu.cn

(a) Input views (b) Style image (c) Stylized novel views

Figure 1. **Transferring photorealistic style with a style image in the 3D scene**. Multi-view images of a given set of 3D scenes (a) and a style image (b), our model is capable of rendering photorealistic stylized novel views (c) with a consistent appearance at various view angles in 3D space.

Abstract

3D scenes photorealistic stylization aims to generate photorealistic images from arbitrary novel views according to a given style image while ensuring consistency when rendering from different viewpoints. Some existing stylization methods with neural radiance fields can effectively predict stylized scenes by combining the features of the style image with multi-view images to train 3D scenes. However, these methods generate novel view images that contain objectionable artifacts. Besides, they cannot achieve universal photorealistic stylization for a 3D scene. Therefore, a styling image must retrain a 3D scene representation network based on a neural radiation field. We propose a novel 3D scene photorealistic style transfer framework to address these issues. It can realize photorealistic 3D scene style transfer with a 2D style image. We first pretrained a 2D photorealistic style transfer network, which can meet the photorealistic style transfer between any given content image and style image. Then, we use voxel features to optimize a 3D scene and get the geometric representation of the scene. Finally, we jointly optimize a hyper network to realize the scene photorealistic style transfer of arbitrary style images. In the transfer stage, we use a pretrained 2D photorealistic network to constrain the photorealistic style of different views and different style images in the 3D scene. The experimental results show that our method not only realizes the 3D photorealistic style transfer of arbitrary style images but also outperforms the existing methods in terms of visual quality and consistency. Project

*Corresponding Author is Wei Wang (wangwei@sobey.com).

page:https://semchan.github.io/UPST_NeRF/.

1. Introduction

In recent years, the 3D implicit representation method based on the neural radiation field [35, 55] has made great progress because of its excellent performance in scene realism. By controlling the appearance in these scenes, style transfer can reduce the time of artistic creation and the need for professional knowledge. Many excellent works achieve this goal through texture generation [11, 22, 52] and semantic view synthesis [15, 17]. Some recent work [5, 9, 16, 18, 20, 36, 54] can transfer artistic features from a single 2D image to a complete real 3D scene, thereby changing the style in the real scene.

Most of these methods focus on how to solve the consistency problem of stylized scenes. LSVN [18] proposed a point cloud-based method for consistent 3D scene stylization. StyleMesh [16] optimized an explicit texture for the reconstructed mesh of a scene and stylized it jointly from all available input images. StylizedNeRF [20] proposed a mutual learning framework for 3D scene stylization, which combines a 2D image stylization network and NeRF to fuse the stylization ability of a 2D stylization network with the 3D consistency of NeRF. To solve the blurry results and inconsistent appearance, Stylizing-3D-Scene [5] utilized a hyper network to transfer the style information into the scene representation. To eliminate the jittering artifacts due to the lack of cross-view consistency, SNeRF [36] investigated 3D scene stylization that provides a strong inductive bias for consistent novel view synthesis. ARF [54] proposed to stylize the more robust radiance field representation and produce consistent stylized novel views of high visual quality. Besides, INS [9] conducted a pilot study on various implicit functions, including 2D coordinate-based representation, neural radiance field, and signed distance function. These methods can only realize the transfer of artistic style but not the transfer of realistic style. If photorealistic images are used as style images to transfer the style of 3D scenes, the stylized scenes will contain objectionable artifacts.

This paper aims to stylize a photorealistic 3D scene following a given set of style examples. Our method allows generating stylized images of the scene from arbitrary novel views while ensuring rendered images from different viewpoints are consistent. To ensure consistency, we formulate the problem as stylizing a NeRF [35] with a given set of style images. Some examples of our stylization method are presented in Fig. 1.

Neural radiance fields (NeRF) [35] use multi-layer perceptron (MLP) to implicitly learn the mapping from the queried 3D point with its colors and densities to reconstruct a volumetric scene representation from a set of images. This method dramatically improves the quality of scene rendering, but it requires a long training time and inefficient novel view rendering. To reconstruct the scene quickly, we are inspired by DirectVoxelGO [44] and use the voxel grid to directly optimize the geometric appearance of the scene in our geometric training stage. It contains two voxel grids: one is the density voxel grid, which is used to predict the occupancy probability; the other is the feature voxel, which is followed by a shallow MLP(RGBNet) for color predicting.

Since the implicit continuous volumetric representation is built on the millions of voxel grid parameters, it is unclear which parameters control the style information of the 3D scene. To overcome this issue, one possible solution is combining existing image/video stylization approaches with novel view rendering techniques [5]by first rendering novel view images and then performing image stylization. Inspired by Stylizing-3D-Scene [5], we use a HyperNet and HyperLinear to handle the ambiguities of the 2D stylized learnable latent codes as conditioned inputs. Unlike Stylizing-3D-Scene, we use StyleEncoder with VGG [41] to extract style features from the style images. Then we use the style features as the input of HyperNet to update the weights of HyperLiner. Finally, we use HyperLiner to change the information of RGBNet to achieve the style of updating the scene. To present more realistically, instead of directly using Adaptive Instance Normalization (AdaIN) [19] to constrain the style of the novel view, we trained a 2D photorealistic style transfer network to process truth value RGB under different views to obtain the target, which use to constrain the predicted color value.

In a nutshell, our main contributions are as follows:

- We propose a novel universal photorealistic style transfer of neural radiance fields for photorealistic stylizing 3D scenes with given style images.
- We propose using a hyper network to control the features of photorealistic style images as the latent codes of scene stylization to use the 2D method to realize the geometric consistency constraint of the neural radiation field.
- To realize the scene's photorealistic style transfer, we designed an efficient 2D style transfer network to process the 2D photorealistic style images under different novel views to constrain the scene style.

2. Related Work

Novel View Synthesis. Novel view synthesis aims to generate the images at arbitrary viewpoints from a set of source images. Some studies use a single image or a pair of stereo images as input and use methods such as Multi Plane Image (MPI) [34, 43, 45, 49, 57], light field techniques [6, 14, 25],

point cloud [37, 48] to represent the scene for synthesizing novel view near the input viewpoint. However, these methods cannot generate a novel view image from an arbitrary viewpoint. To generate images of novel view image from arbitrary viewpoints , these methods need more images as input to reconstruct the scene. Some works build 3D scenes by combining geometric representation with color [39,46], texture [7], light field [3,50] or neural rendering [1,10,30,33,37,42,48]. This 3D implicit representation method based on neural radiance fields (NeRF) [35] greatly improves the quality of novel view generation. Subsequently, some work extended NeRF to octree structure [29], unbounded scenes [55], reflectance decomposition [2] and uncontrolled real-world images [32]. However, NeRF and its variants require a training time from hours to days for a single scene, making it infeasible for many application scenarios. On the other hand, DirectVoxelGO [44] uses gradient-descent to optimize voxel grids directly predict the grid values and can rapidly train from scratch in less than 15 minutes with a single GPU.

Image and video style transfer. There are two important categories of style transfer tasks: artistic style transfer and photorealistic style transfer. Using Gram matrix [13] can transfer the style information from the reference image to the content image. It is widely used in the task of artistic style transfer. For faster stylization, Avatar [40], and AdaIN [19] leverage feed-forward neural networks. DPST [31] proposed a deep photorealistic style transfer method by constraining the transformation to be locally affine in colorspace. To improve the efficiency, PhotoWCT [27], WCT^2 [53] have been proposed. Xia et al. [51]propose an end-to-end model for photorealistic style transfer that is both fast and inherently generates photorealistic results. Qiao et al. [38] proposed Style-Corpus Constrained Learning (SCCL) to relieve the unrealistic artifacts and heavy computational cost issues.

To ensure the consistency between adjacent frames, and make the stylized video not flicker, optical flow or temporal constraint-based methods [4, 12] are applied to video stylization. MCCNet [8] can be trained to fuse the exemplar style features and input content features for efficient style transfer and achieves coherent results. Wang et al. proposed jointly considering the intrinsic properties of stylization and temporal consistency for video style transfer. However, these 2D-based methods lack spatial consistency constraints and 3D scene perception, so they cannot maintain long-term consistency in 3D scene style transfer.

3D sence style transfer. Through texture generation [11, 22, 52] and semantic view synthesis [15, 17] can editing the appearance in 3D scenes. Using an image as a reference and changing the style of the scene has also become a hot topic of recent research for 3D sense style transfer. Spatial consistency becomes one of the main problems to be solved in 3D scene stylization. For example, LSVN [18] proposed a point cloud-based method for consistent 3D scene stylization, and Stylizing-3D-Scene [5] utilized a hyper network to transfer the style into the scene to solve the blurry results and inconsistent appearance. StyleMesh [16] stylized the 3D scene jointly from all available input images and optimized an explicit texture for scene reconstruction. Stylized-NeRF [20] utilize the stylization ability of 2D stylization network and neural radiation field for 3D scene stylization, and ARF [54] proposed to stylize the more robust radiance field representation. SNeRF [36] investigated 3D scene stylization, providing a strong inductive bias for consistent novel view synthesis. INS [9] studied unifying the style transfer for 2D coordinate-based representation, neural radiance field, and signed distance function. These methods can achieve the artistic style transfer in 3D scenes, but it isn't easy to realize the photorealistic style transfer. When these methods are applied to photorealistic style transfer, they will lead to artifacts when rendering the synthesis of a novel view.

3. Preliminaries

NeRF [35] employ multiplayer perceptron (MLP) networks to model a scene as a continuous volumetric field of opacity and radiance. One MLP, indicated as $\mathrm{MLP}^{(pos)}$, for density predicting and the other MLP, indicated as $\mathrm{MLP}^{(rgb)}$, for radiance color predicting:

$$(\sigma, \boldsymbol{e}) = \mathrm{MLP}^{(\mathrm{pos})}(\mathrm{PE}(\boldsymbol{x}))\ , \tag{1a}$$

$$\boldsymbol{c} = \mathrm{MLP}^{(\mathrm{rgb})}(\boldsymbol{e}, \mathrm{PE}(\boldsymbol{d}))\ , \tag{1b}$$

where $\boldsymbol{x} \in \mathbb{R}^3$ is the 3D position, $\boldsymbol{d} \in \mathbb{R}^2$ is the viewing direction, $\sigma \in \mathbb{R}^+$ is the corresponding density, $\boldsymbol{c} \in \mathbb{R}^3$ is the view-dependent color emission, $\boldsymbol{e} \in \mathbb{R}^{D_e}$ is an intermediate embedding tensor with dimension D_e and PE is positional encoding.

The ray $\boldsymbol{r}$ from the camera center through the pixel for rendering the color of a pixel $C(\boldsymbol{r})$:

$$C(\boldsymbol{r}) = \left(\sum_{i=1}^{K} T_i \alpha_i \boldsymbol{c}_i\right) + T_{K+1}\boldsymbol{c}_{\mathrm{bg}}\ , \tag{2a}$$

$$\alpha_i = \mathrm{alpha}(\sigma_i, \delta_i) = 1 - \exp(-\sigma_i \delta_i)\ , \tag{2b}$$

$$T_i = \prod_{j=1}^{i-1}(1 - \alpha_j)\ , \tag{2c}$$

where K is the number of sampling points on $\boldsymbol{r}$ between the pre-defined near and far planes; α_i is the probability of termination at the point i; T_i is the accumulated transmittance from the near plane to point i; δ_i is the distance to the adjacent sampled point, and $\boldsymbol{c}_{bg}$ is a pre-defined background color.

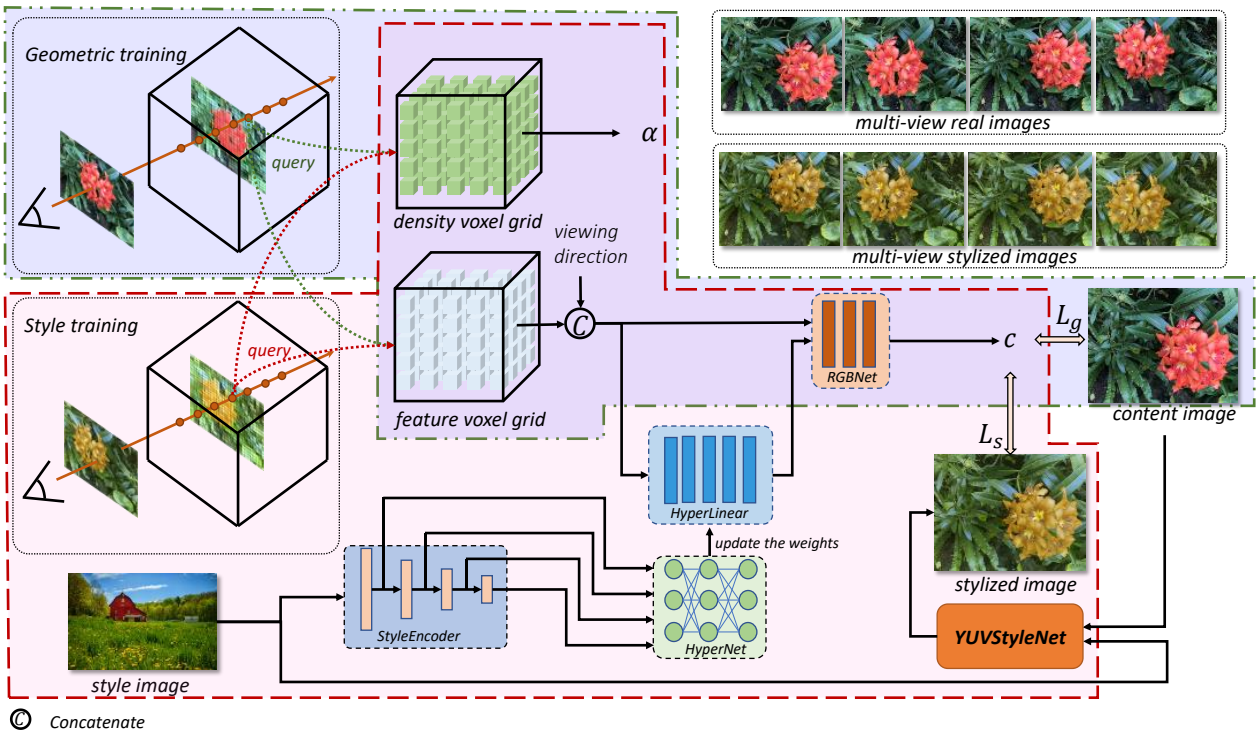

Figure 2. **Overview of Universal Photorealistic Style Transfer of Neural Radiance Fields.** In our framework, the training in photorealistic style transfer in 3D scenes divides into two stages. The first stage is geometric training for a single scene. We use the density voxel grid and feature voxel grid to represent the scene directly, and the density voxel grid is used to output density; the feature voxel grid with a shallow MLP of RGBNet use to predict the color. The second stage is style training. The parameters of the density voxel grid and feature voxel grid will be frozen, and we use a reference style image's features to be the input of the hyper network, which can control the RGBNet's input. Thus, we jointly optimize the hyper network to realize the scene photorealistic style transfer with arbitrary style images.

In the training stage, NeRF optimizes the model by minimizing the Mean Square Error (MSE) between the pixel color $C(\boldsymbol{r})$ of the image in the training set and the rendered pixel color $C_{gt}(\boldsymbol{r})$.

$$\mathcal{L}_{\text{mse}} = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \|C(\boldsymbol{r}) - C_{gt}(\boldsymbol{r})\|_2^2 \ , \tag{3}$$

where $\mathcal{R}$ is the set of rays in a sampled mini-batch.

4. Our Approach

The overview of the universal photorealistic style transfer of neural radiance fields has been shown in Fig. 2. Through several images of a given scene, our goal is to generate a photorealistic styled image of an arbitrary viewpoint in the scene according to the reference style image while maintaining geometric consistency. In our framework, for a single scene, we achieve the training through two stages: geometric training and style training. Then, in the rendering processing, we can synthesize novel viewpoints with photorealistic style transfer according to the style of an arbitrary reference image.

4.1. Scene Geometric Reconstruction

Similar to DirectVoxelGO [44], we adopt voxel grid to represent the 3D scene. Such an scene representation is efficient to query for any 3D positions via interpolation:

$$\text{interp}(\boldsymbol{x}, \boldsymbol{V}) : \left(\mathbb{R}^3, \mathbb{R}^{C \times N_x \times N_y \times N_z}\right) \rightarrow \mathbb{R}^C \ , \tag{4}$$

where $\boldsymbol{x}$ is the queried 3D point, $\boldsymbol{V}$ is the voxel grid, C is the dimension of the modality, and $N_x \cdot N_y \cdot N_z$ is the total number of voxels. Trilinear interpolation is applied if not specified otherwise.

$$\alpha = \text{alpha}(\text{softplus}(\text{interp}(\boldsymbol{x}, \boldsymbol{V}^{(\text{density})}))) \tag{5}$$

where alpha (Eq. (2b)) functions sequentially for volume rendering, softplus is the activation function and $\boldsymbol{V}^{(\text{density})} \in \mathbb{R}^{1 \times N_x \times N_y \times N_z}$ is the density voxel grid.

For view-dependent color emission predicting can be expressed as:

$$\boldsymbol{c} = \text{MLP}^{(\text{rgb})}\left(\text{interp}(\boldsymbol{x}, \boldsymbol{V}^{(\text{feat})}), \boldsymbol{x}, \boldsymbol{d}\right) \tag{6}$$

where $\boldsymbol{c} \in \mathbb{R}^3$ is the view-dependent color emission, $\boldsymbol{V}^{(\text{feat})} \in \mathbb{R}^{D \times N_x \times N_y \times N_z}$ is the feature voxel grid, D is

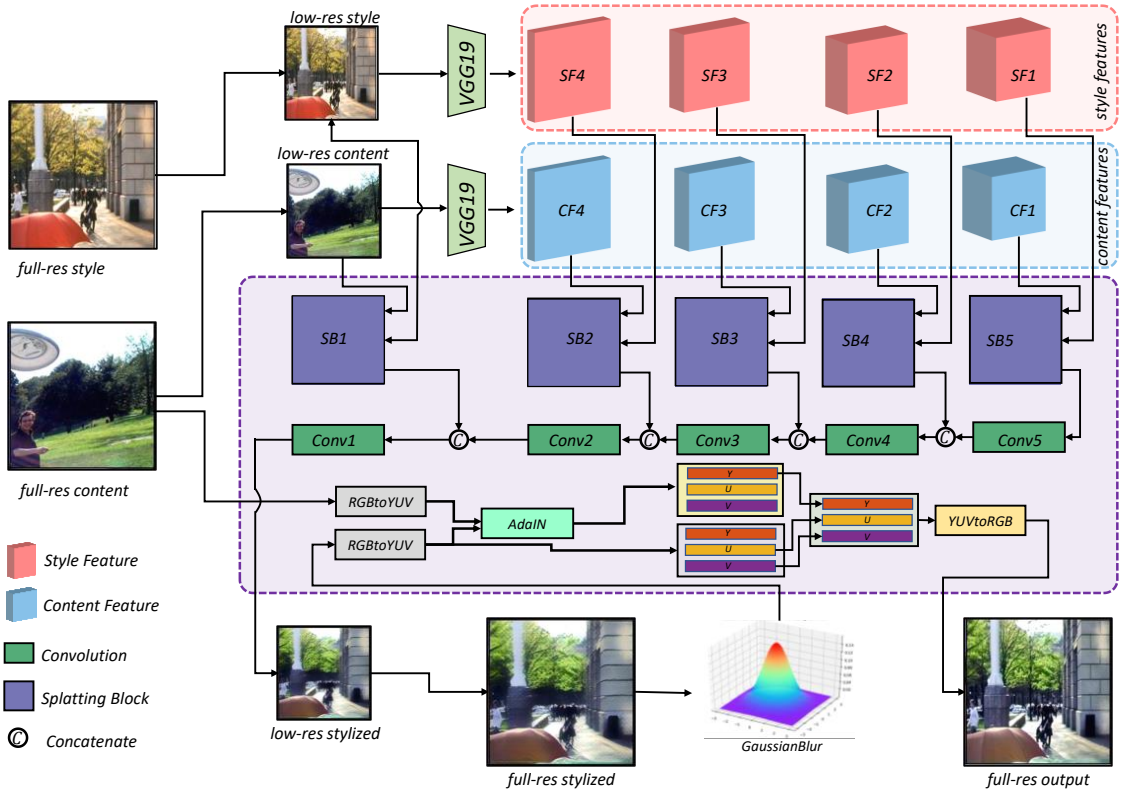

Figure 3. **The architecture of YUVStyleNet.** We designed a framework for 2D photorealistic style transfer, which supports the input of a full resolution style image and a full resolution content image, and realizes the photorealistic transfer of styles from the style image to the content image. In this framework, we transform the image into YUV channels. The final fusion uses the generated stylized UV channel, and the Y channel fusion after the stylized image is fused with the original content image to get the final photorealistic stylized image.

a hyperparameter for feature-space dimension. By default, we set D equal to 128. The MLP is shown in Fig. 2 as RGBNet.

We use the photometric loss in Eq.3. Similar to DirectVoxelGO [44], we incorporate per-point rgb loss and background entropy loss and the modification loss as below:

$$\mathcal{L}_{\text{pt\_rgb}} = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \sum_{i=1}^{K} \left(T_i \alpha_i \left\| \boldsymbol{c}_i - C(\boldsymbol{r}) \right\|_2^2 \right). \quad (7)$$

The background entropy loss regularizes the rendered background probability, T_{K+1} in Eq. (2), to concentrate on either foreground or background:

$$\mathcal{L}_{\text{bg}} = -T_{K+1} \log(T_{K+1}) - (1 - T_{K+1}) \log(1 - T_{K+1}). \quad (8)$$

Finally, the overall training objective of the geometric training stage is

$$L_g = \lambda^{(c)}_{\text{photo}} \cdot \mathcal{L}_{\text{photo}} + \lambda^{(c)}_{\text{pt\_rgb}} \cdot \mathcal{L}_{\text{pt\_rgb}} + \lambda^{(c)}_{\text{bg}} \cdot \mathcal{L}_{\text{bg}}, \quad (9)$$

where $\lambda^{(c)}_{\text{photo}}, \lambda^{(c)}_{\text{pt\_rgb}}, \lambda^{(c)}_{\text{bg}}$ are hyper parameters of the loss weights.

4.2. YUVStyleNet for 2D photorealistic stylization

2D photorealistic stylization is the task of transferring the style of a reference image onto a content target, which makes a photorealistic result that is plausibly taken with a camera. In our work, we designed a 2D photorealistic style transfer network for the photorealistic style transfer of images from a novel view of the scene to get a photorealistic style transfer network in the style training stage. We name the 2D photorealistic style transfer network YUVStyleNet, and its detailed framework shows in Fig. 3.

To reduce the GPU memory and improve the processing speed, the input full-resolution style image I^s_f and the content image I^c_f will be downsampled to the corresponding low-resolution images I^s_l and I^c_l. In our experiment, the size of low-resolution image is 512 by default. Then, we use a pre-trained VGG model to extract style features $\{F^s_j\}|1 \leq j \leq 4$ and content features $\{F^c_j\}|1 \leq j \leq 4$ at different scales respectively. The style features and content features of the corresponding scale, as well as the corresponding low-resolution style image and content image, are used as feature pairs as the input of the splatting block module to obtain the output $\{F^{sb}_i\}|1 \leq i \leq 5$ under the

corresponding scale.

We first extract the input s feature and c feature through a convolution and then use adaptive instance normalization (AdaIN) [19] to fuse the features in the splatting block module. Splatting block output features are concatenated with low-scale features, respectively, and then through convolution operation, the low resolution stylized image I_l^{sed} is finally obtained. By upsampling, we get a stylized image I_f^{sed} with the same scale as the input I_f^c. To make the color transfer smoother in space, we use a Gaussian filter to process I_f^{sed} and get I_f^{sg}. We convert the original content image and I_f^{sg} to YUV domain as I_f^{cyuv} and I_f^{sgyuv} , and then get I_f^{scyuv} through AdaIN. To make the generated photorealistic stylized image consistent with the original image in brightness, we extract Y channels from I_f^{scyuv} and UV channels from I_f^{sgyuv} and to get a new style image I_f^{sedyuv}. Then the final I_f^{sedrgb} is obtained by converting to RGB space.

We refer to AdaIN [19] to define our style loss L_s and content loss L_c. In addition, to obtain a more photorealistic effect, We compared Peak Signal to Noise Ratio (PSNR) and Structural Similarity (SSIM) as constraints between the generated style image I_f^{sedrgb} and the original content image I_f^c, we add PSNR loss L_{psnr} and SSIM loss L_{ssim}:

$$L_{yuvs} = \lambda_c \cdot L_c + \lambda_s \cdot L_s + \lambda_{psnr} \cdot L_{psnr} + \lambda_{ssim} \cdot L_{ssim} \quad (10)$$

We randomly select content image and style image in MS-COCO [28] to train YUVStyleNet, and finally optimize to get a better effect of photorealistic style transfer.

4.3. Style Learning in 3D Scene

To change the style of the scene by using arbitrary style images as input, we designed a hyper network (HyperNet) and a hyper linear network (HyperLinear) to control the input features of RGBNet when rendering the scene. As shown in Fig.2, in the style training stage, the feature voxel grid feature queried under the corresponding view is spliced with the view direction feature as the input of HyperLinear network, and the output of HyperLinear will be directly used as the input of RGBNet to control the generation of color. The style image is extracted through the pre-trained feature extraction network, VGGNet, and then used as the input of HyperNet. The output of HyperNet is used to control the weight of HyperLinear, to modify the scene's color through style features.

In the stage of 3D style training, we constrain the training process of style transfer by optimizing Eq. 7, 8 and 9. The difference from that introduced in Section 4.1 is that we have changed $C(\boldsymbol{r})$ with YUVStyleNet. We get the corresponding content image through a mini-batch of rays,demonstrate in Fig. 4, and the style image randomly collected from MS-COCO [28] is used as the input of

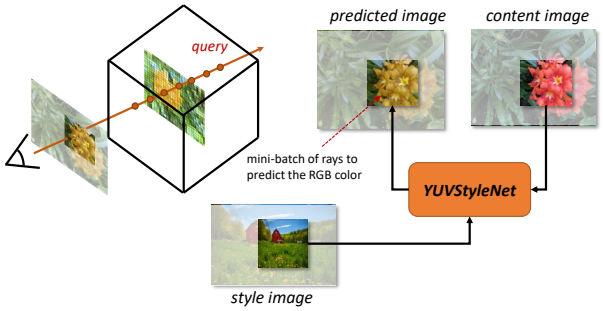

Figure 4. **RGB color predicting with mini-batch of rays in style training**

YUVStyleNet, and the predicted image is used as $C_s(\boldsymbol{r})$. Therefore, Eq. 7 can be adjusted to:

$$\mathcal{L}_{\text{s\_pt\_rgb}} = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \sum_{i=1}^{K} \left(T_i \alpha_i \left\| \boldsymbol{c}_i - C_s(\boldsymbol{r}) \right\|_2^2 \right). \quad (11)$$

Thus, the loss constraint of scene style transfer is:

$$L_s = \lambda_{\text{photo}}^{(c)} \cdot \mathcal{L}_{\text{photo}} + \lambda_{\text{s\_pt\_rgb}}^{(c)} \cdot \mathcal{L}_{\text{s\_pt\_rgb}} + \lambda_{\text{bg}}^{(c)} \cdot \mathcal{L}_{\text{bg}}. \quad (12)$$

where $\lambda_{\text{photo}}^{(c)}, \lambda_{\text{s\_pt\_rgb}}^{(c)}, \lambda_{\text{bg}}^{(c)}$ are hyper parameters of the loss weights.

In the style training stage, we froze the density voxel grid and feature voxel grid, which is optimized in the process of geometry training. At the same time, we also froze the parameters of YUVStyleNet, and the parameters of StyleEncoder, which uses VGG as the decoder for feature extraction.

5. Experiments

We have done the qualitative and quantitative evaluation tests for our method and also comparisons with the state-of-the-art stylization methods for video and 3D scenes, respectively. In our geometric training stage, we use the Adam optimizer with a batch size of 8,192 rays to optimize the scene representations for 20k iterations; in our style training stage, we use the Adam optimizer with a batch size of 10,000 rays to optimize the scene style representation for 200k iterations. The base learning rates are 0.1 for all voxel grids and 10^{-3} for RGBNet, HyperLinear, and HyperNet. We test our method on two types of datasets: NeRF-Synthetic datasets [35] and Local Light Field Fusion(LLFF) datasets [34]. On the other hand, we use images in the MS-COCO [28] as the reference style images in the style training stage. All experiments are performed on a single NVIDIA TITAN RTX GPU.

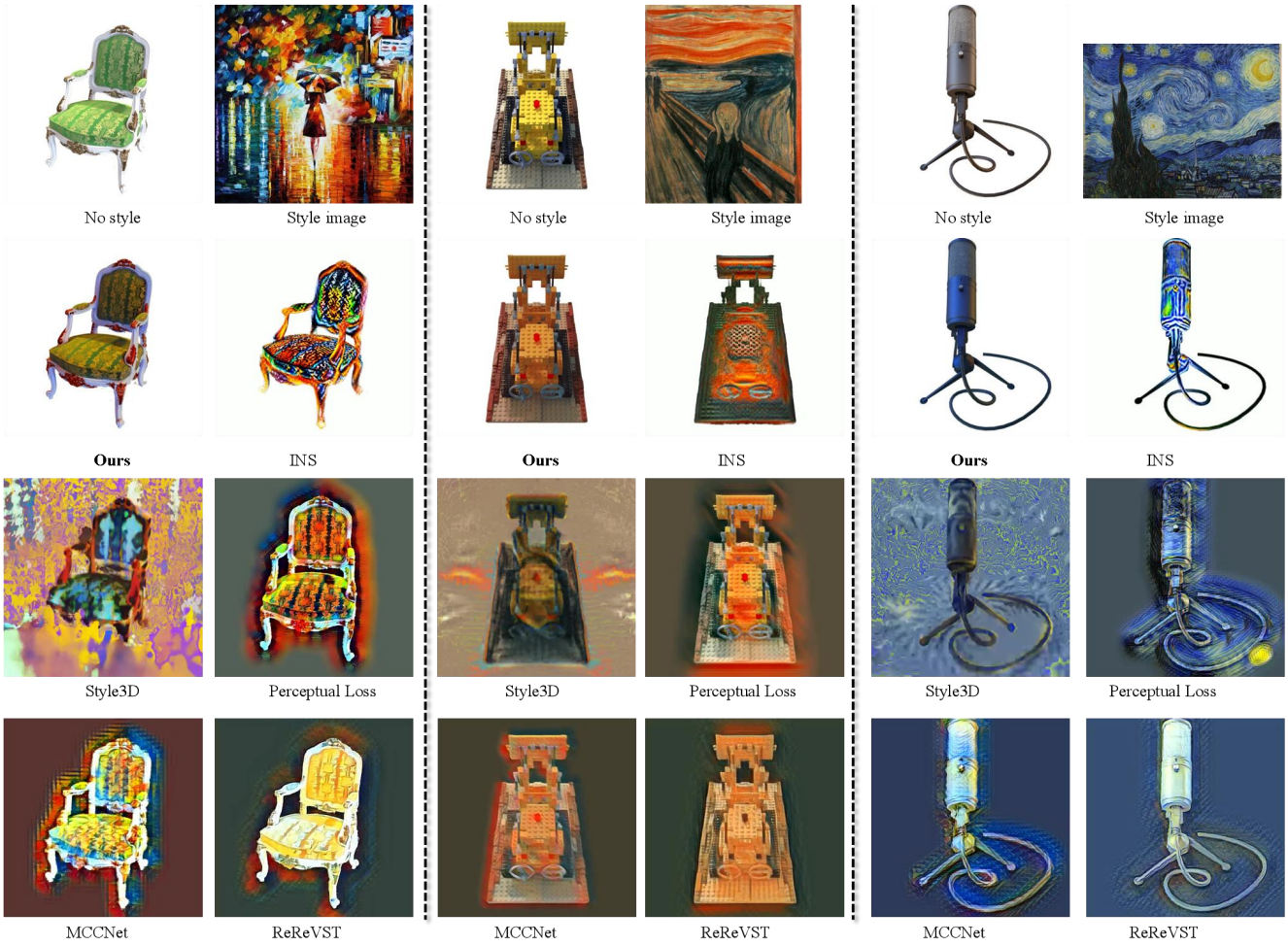

Figure 5. **Qualitative comparisons with artistic style images**. We compare the stylized results of 3 scenes on NeRF-Synthetic dataset. Our method stylizes scenes with more precise geometry and competitive stylization quality.

5.1. Qualitative Results

Photorealistic style transfers with artistic style images. In Fig. 5, we qualitatively compare the photorealistic style transfer results generated by INS [9], Style3D [5], Perceptual Loss [21], MCCNet [8], ReReVST [47] and ours. Although INS has targeted training for the style image in the stylized scene, its geometric representation is still far from our results. Other results in the absence of good geometry and the loss of precision, which further damages the stylization results. For example, the edge of the chair is not clear enough, and even one leg of the chair cannot be seen in the result of Style3D. At the same time, the artifacts of other methods are also severe. In contrast, our approach retains a clear geometric representation and can migrate a more realistic style from the style image, thereby changing the color in the scene.

Photorealistic style transfer with photorealistic style images. In Fig. 6, we qualitatively compare the photorealistic style transfer results generated by ARF [54], AdaIN [19], MCCNet [8], ReReVST [47] and ours with photorealistic style images. According to the default configuration of ARF, we retrained scenes with different realistic style images. MCCNet [8] and ReReVST [47] are two state-of-the-art video stylization methods. We should point out that ARF needs to retrain the scene according to the style image when rendering a new style scene, but our method does not require retraining. Instead, we can get a stylized scene by inputting the embedded features of the new style image into the network during rendering. We can see from the results that ARF will disorderly integrate the visual features in the style image into the scene when stylizing a new scene. However, our results only perfectly migrate the color information according to the scene's style, preserving the scene's geometric features to the greatest extent.

Video stylization. In Fig. 7, we compare our results in multiple views with and without photorealistic stylization. The results show our method almost has no effect on the depth

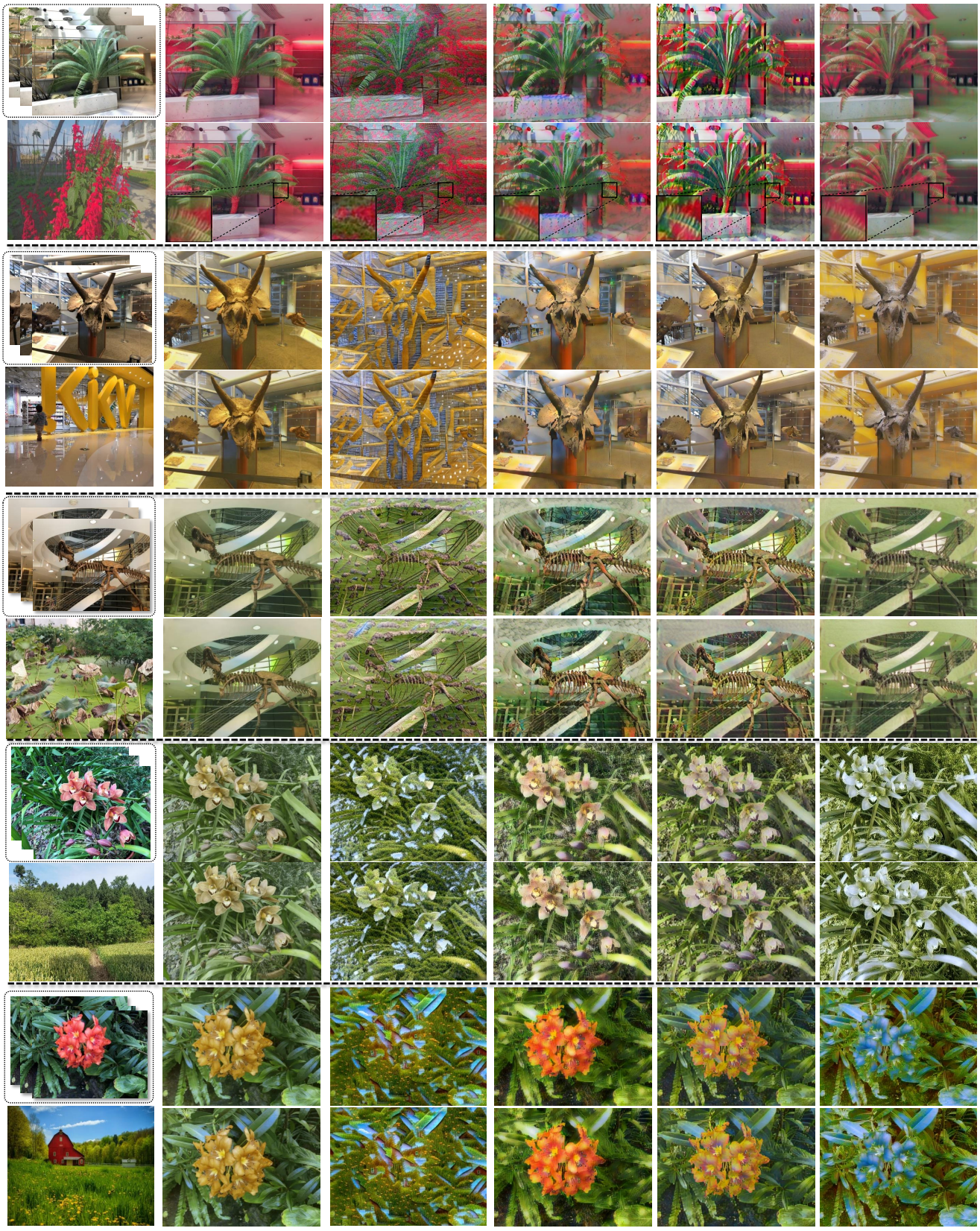

Figure 6. **Qualitative comparisons with photorealistic style images.** We compare the stylized results of 5 scenes on Local Light Field Fusion(LLFF) [34] dataset. Our method stylizes scenes with more precise geometry and competitive stylization quality.

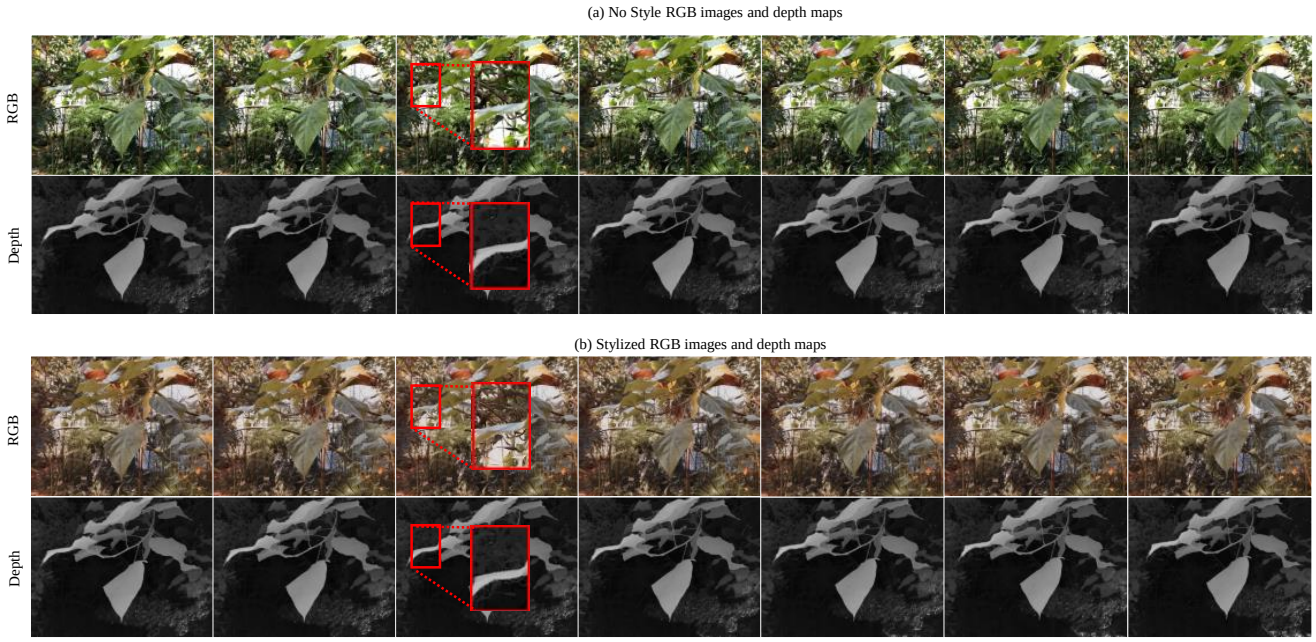

Figure 7. **Qualitative comparisons with no style multi-view images and stylized multi-view images.** The upper is the results without photorealistic style transfer, and the lower is the results with photorealistic style transfer from our method.

value except the color of RGB. This is because we separate geometry training from style training and fix the voxel grid representing geometric features during style training stage.

5.2. Quantitative Results

Consistency Measurement.

We measure the short and long-term consistency using the warped LPIPS metric [56]. A view v is warped with the depth expectation estimated by the learning from our geometric training. We use the measurement implemented from [24]. The consistency score formulates as:

$$E(O_i, O_j) = LPIPS(O_i, M_{i,j}, W_{i,j}(O_j)) \qquad (13)$$

where W is the warping function and M is the warping mask. When calculating the average distance across spatial dimensions in [56], only pixels within the mask $M_{i,j}$ are taken. We compute the evaluation values on 5 scenes in the LLFF [34] dataset, using 20 pairs of views for each scene. The test views are upsampled three times the training views to ensure the density of frames for video-based methods. We use every two adjacent novel views (O_i, O_{i+1}) and view pairs of gap 5 (O_i, O_{i+5}) for short and long-range consistency calculation. The comparisons of short and long-range consistency are shown in Tab. 1 and Tab. 2, respectively. Our method outperforms other methods by a significant margin.

User study. A user study is conducted to compare our method's stylization and consistent quality with other state-of-the-art methods. We stylize ten series of views of the 3D scenes in the LLFF [34] dataset, using different methods [8], [47], [18] and invite 30 participants (including 25 males, 5 females, aged from 20 to 43). First, we showed the participants a style image, two stylized videos generated by our method, and a random compared method. Then we asked the participants their votes for the video in two evaluating indicators, quality of the stylized results and whether to keep the consistency. We collected 600 votes for each evaluating indicator and presented the result in Fig. 8 in the form of the boxplot. Our scores stand out from other methods in photorealistic stylization quality and consistency.

Table 1. **Short-range consistency.** We compare the short-range consistency using warping error(↓). **Best** results are highlighted.

Method	Fern	Flower	Horns	Orchids	Trex	Average
AdaIN	0.0051	0.0033	0.0055	0.0066	0.0041	0.0049
MCCNet	0.0038	0.0022	0.0039	0.0044	0.0027	0.0034
ReReVST	0.0011	0.0007	0.0011	0.0019	0.0009	0.0011
ARF	0.0010	0.0006	0.0013	0.0022	0.0015	0.0013
Ours	**0.0005**	**0.0001**	**0.0003**	**0.0009**	**0.0003**	**0.0004**

Table 2. **Long-range consistency.** We compare the long-range consistency using warping error(↓). **Best** results are highlighted.

Method	Fern	Flower	Horns	Orchids	Trex	Average
AdaIN	0.0087	0.0063	0.0097	0.0100	0.0078	0.0085
MCCNet	0.0070	0.0042	0.0078	0.0074	0.0058	0.0065
ReReVST	0.0035	0.0025	0.0035	0.0053	0.0024	0.0035
ARF	0.0042	0.0027	0.0053	0.0075	0.0051	0.0050
Ours	**0.0024**	**0.0009**	**0.0020**	**0.0032**	**0.0015**	**0.0020**

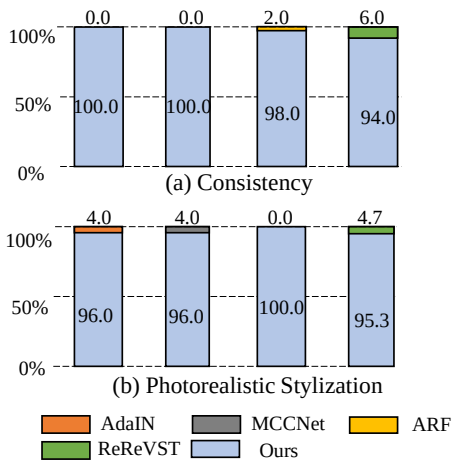

Figure 8. **User study.** We record the user preference in the form of boxplot. Our results win more preferences both in the photorealistic stylization and consistency quality.

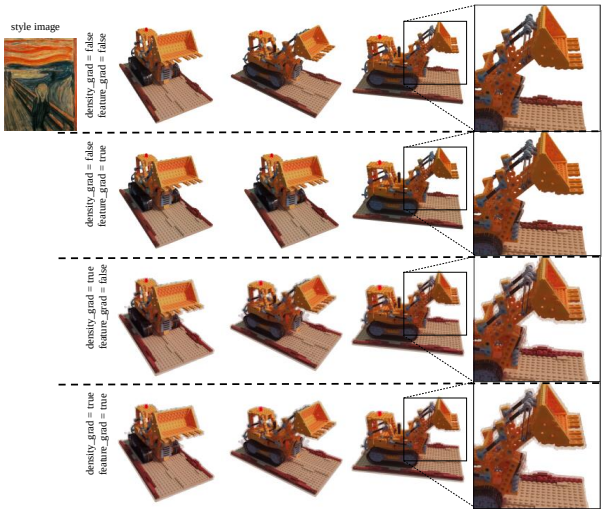

Figure 9. **The impact of voxel grid gradient propagation in style training stage.** L_d clusters latent codes of the same style and avoids the artifacts in test results.

5.3. Ablation Study

The impact of voxel grid gradient propagation in the style training stage. We believe that the most critical performance of photorealistic style transfer is that the scene's color needs to be consistent with the reference style image, and the methods cannot change the geometric information of the scene. That is, photorealistic style transfer should not change the geometric shape of the scene. Based on this principle, we first trained the scene's geometry and then froze the parameters of the voxel grid for style training. We explored the impact of voxel gradient propagation in the style training stage. We try to freeze the parameters in the density voxel grid and feature voxel grid, respectively, in the process of gradient promotion and then compare the results of style transfer of the trained network. The result shown in Fig. 9. density_grad = true and feature_grad = true indicate the parameters in density voxel grid and feature voxel grid not be frozen in style training, respectively. From the results, we can see that as long as we freeze the parameters of the density voxel grid in the style training stage, we can get a better photorealistic style transfer effect while keeping the geometric information of the scene unchanged.

The impact of using a 2D photorealistic style network to constrain scene style. In our method, we design a virtual 2D photorealistic style transfer network, YUVStyleNet, which is used to generate photorealistic style images in the style training stage to constrain the style of the scene. This will significantly ensure that our realistic style transfer scene is more photorealistic. To verify this, we directly use AdaIN as a loss to constrain the style training process. As a result, the direct use of AdaIN constraints is more blurred in the generated novel view images than in our method, as shown in Fig. 10.

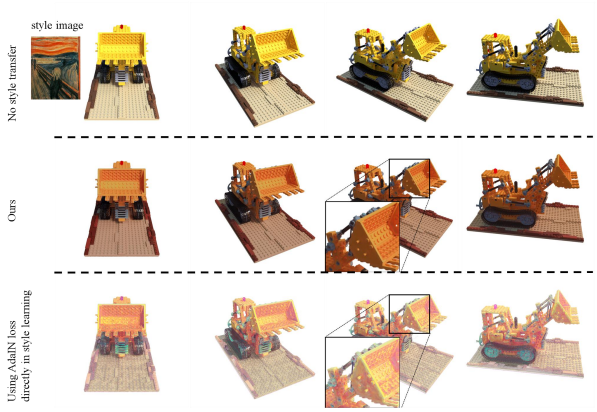

Figure 10. **The impact of using 2D photorealistic style network to constrain scene style.**

The impact of a batch size of rays in style training. We conduct stylization constraints in the style training stage by processing 2D style images and small batches of novel view images. When the batch size is larger, the novel view of the captured scene will be larger, with higher global constraints. On the contrary, it will be closer to local constraints. We studied this in Fig. 11. HW is the space through which a rectangular batch of rays passes. For example, if HW = 10, the batch size of the rays is $10 \times 10 = 100$. We can see from the results that the change from 10 to 100 causes the change of color, but the overall impact is negligible.

Limitations. The quality of the photorealistic stylization results is limited by the geometric training stage. We use a voxel grid to represent the geometric of the scene. When the scene to be represented is large enough, this method will consume huge storage space. Therefore, the maximum value of the size of the voxel grid is limited, so the method cannot reconstruct some large scenes well enough. This

Figure 11. **Ablation study on a batch size of rays in style training.** We compare the results with different batches of rays in style training. HW=10 indicate the batch size is $10 \times 10 = 100$ of rays.

Figure 12. **Comparisions on Tanks and Temples [23] datasets.** We compare the results on the large scale of sense datasets, artifacts in the results may exist.

also affects the final photorealistic transfer results. Fig. 12 shows the results of a large scene dataset. It can be seen that our method has artifacts in the sky.

6. Conclusion

We present a universal photorealistic style transfer method with neural radiance fields for the 3D scene. We directly reconstruct the geometric representation through the voxel grid and then introduce the features of different 2D style images for scene style control in the style training stage. To achieve this, we use a hyper network to control the weights. Further, we use the pre-trained 2D photorealistic style network to perform photorealistic style transfer on the input style image and the novel view image of the scene to constrain the training of scene photorealistic style transfer. Our method outperforms state-of-the-art methods both in terms of visual quality and consistency. However, our direct optimization of scene geometry via voxel grid has limitations in large 3D scenes. In the future, we will explore the problem of photorealistic style transfer in large scenes. At the same time, we will focus on exploring the use of neural radiation fields to solve the problem of color consistency in different scenes.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China under Grant No. 62172061; National Key R&D Program of China under Grant No. 2020YFB1711800 and 2020YFB1707900. We are grateful to thank the support from Peng Cheng Laboratory. We sincerely appreciate all participants in the user study.

References

[1] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 696–712. Springer, 2020. 3

[2] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. NeRD: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12684–12694, 2021. 3

[3] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 425–432, 2001. 3

[4] Dongdong Chen, Jing Liao, Lu Yuan, Nenghai Yu, and Gang Hua. Coherent online video style transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1105–1114, 2017. 3

[5] Pei-Ze Chiang, Meng-Shiun Tsai, Hung-Yu Tseng, Wei-Sheng Lai, and Wei-Chen Chiu. Stylizing 3d scene via implicit representation and hypernetwork. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1475–1484, 2022. 2, 3, 7

[6] Abe Davis, Marc Levoy, and Fredo Durand. Unstructured light fields. *Computer Graphics Forum*, 31(2pt1):305–314, 2012. 2

[7] Paul E Debevec, Camillo J Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 11–20, 1996. 3

[8] Yingying Deng, Fan Tang, Weiming Dong, haibin Huang, Ma chongyang, and Changsheng Xu. Arbitrary video style transfer via multi-channel correlation. In *AAAI*, 2021. 3, 7, 9

[9] Zhiwen Fan, Yifan Jiang, Peihao Wang, Xinyu Gong, Dejia Xu, and Zhangyang Wang. Unified implicit neural stylization. *arXiv preprint arXiv:2204.01943*, 2022. 2, 3, 7

[10] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. DeepView: View synthesis with learned gradient descent. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2367–2376, 2019. 3

[11] Lin Gao, Tong Wu, Yu-Jie Yuan, Ming-Xian Lin, Yu-Kun Lai, and Hao Zhang. TM-NET: Deep generative networks for textured meshes. *ACM Transactions on Graphics (TOG)*, 40(6):263:1–263:15, 2021. 2, 3

[12] Wei Gao, Yijun Li, Yihang Yin, and Ming-Hsuan Yang. Fast video multi-style transfer. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3222–3230, 2020. 3

[13] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *Nature Communications*, 2015. 3

[14] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 43–54, 1996. 2

[15] Tewodros Habtegebrial, Varun Jampani, Orazio Gallo, and Didier Stricker. Generative view synthesis: From single-view semantics to novel-view images. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2, 3

[16] Lukas Höllein, Justin Johnson, and Matthias Nießner. Stylemesh: Style transfer for indoor 3d scene reconstructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6198–6208, 2022. 2, 3

[17] Hsin-Ping Huang, Hung-Yu Tseng, Hsin-Ying Lee, and Jia-Bin Huang. Semantic view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 592–608. Springer, 2020. 2, 3

[18] Hsin-Ping Huang, Hung-Yu Tseng, Saurabh Saini, Maneesh Singh, and Ming-Hsuan Yang. Learning to stylize novel views. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13869–13878, 2021. 2, 3, 9

[19] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1501–1510, 2017. 2, 3, 6, 7, 14

[20] Yi-Hua Huang, Yue He, Yu-Jie Yuan, Yu-Kun Lai, and Lin Gao. Stylizednerf: consistent 3d scene stylization as stylized nerf via 2d-3d mutual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18342–18352, 2022. 2, 3

[21] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Proceedings of the European conference on computer vision (ECCV)*, pages 694–711. Springer, 2016. 7

[22] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 371–386, 2018. 2, 3

[23] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (TOG)*, 36(4):1–13, 2017. 11

[24] Wei-Sheng Lai, Jia-Bin Huang, Oliver Wang, Eli Shechtman, Ersin Yumer, and Ming-Hsuan Yang. Learning blind video temporal consistency. In *Proceedings of the European conference on computer vision (ECCV)*, pages 170–185, 2018. 9

[25] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 31–42, 1996. 2

[26] Xueting Li, Sifei Liu, Jan Kautz, and Ming-Hsuan Yang. Learning linear transformations for fast image and video style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3809–3817, 2019. 15

[27] Yijun Li, Ming-Yu Liu, Xueting Li, Ming-Hsuan Yang, and Jan Kautz. A closed-form solution to photorealistic image stylization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 453–468, 2018. 3, 15

[28] T. Y. Lin, M. Maire, S. Belongie, J. Hays, and C. L. Zitnick. Microsoft coco: Common objects in context. *Springer International Publishing*, 2014. 6

[29] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3

[30] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Transactions on Graphics (TOG)*, 2019. 3

[31] Fujun Luan, Sylvain Paris, Eli Shechtman, and Kavita Bala. Deep photo style transfer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4990–4998, 2017. 3

[32] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7210–7219, 2021. 3

[33] Moustafa Meshry, Dan B Goldman, Sameh Khamis, Hugues Hoppe, Rohit Pandey, Noah Snavely, and Ricardo Martin-Brualla. Neural rerendering in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6878–6887, 2019. 3

[34] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019. 2, 6, 8, 9, 15

[35] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European conference on computer vision (ECCV)*, pages 405–421. Springer, 2020. 2, 3, 6, 15

[36] Thu Nguyen-Phuoc, Feng Liu, and Lei Xiao. Snerf: Stylized neural implicit representations for 3d scenes. *arXiv preprint arXiv:2207.02363*, 2022. 2, 3

[37] Simon Niklaus, Long Mai, Jimei Yang, and Feng Liu. 3D Ken Burns effect from a single image. *ACM Transactions on Graphics (TOG)*, 38(6):1–15, 2019. 3

[38] Yingxu Qiao, Jiabao Cui, Fuxian Huang, Hongmin Liu, Cuizhu Bao, and Xi Li. Efficient style-corpus constrained learning for photorealistic style transfer. *IEEE Transactions on Image Processing*, 30:3154–3166, 2021. 3

[39] Steven M Seitz and Charles R Dyer. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision*, 35(2):151–173, 1999. 3

[40] Lu Sheng, Ziyi Lin, Jing Shao, and Xiaogang Wang. Avatar-net: Multi-scale zero-shot style transfer by feature decoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8242–8250, 2018. 3

[41] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 2

[42] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. DeepVoxels: Learning persistent 3D feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2437–2446, 2019. 3

[43] Pratul P Srinivasan, Richard Tucker, Jonathan T Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 175–184, 2019. 2

[44] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022. 2, 3, 4, 5

[45] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 551–560, 2020. 2

[46] Michael Waechter, Nils Moehrle, and Michael Goesele. Let there be color! large-scale texturing of 3D reconstructions. In *Proceedings of the European conference on computer vision (ECCV)*, pages 836–850. Springer, 2014. 3

[47] Wenjing Wang, Shuai Yang, Jizheng Xu, and Jiaying Liu. Consistent video style transfer via relaxation and regularization. *IEEE Transactions on Image Processing*, 29:9125–9139, 2020. 7, 9

[48] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. SynSin: End-to-end view synthesis from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7467–7477, 2020. 3

[49] Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. NeX: Real-time view synthesis with neural basis expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8534–8543, 2021. 2

[50] Daniel N Wood, Daniel I Azuma, Ken Aldinger, Brian Curless, Tom Duchamp, David H Salesin, and Werner Stuetzle. Surface light fields for 3D photography. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 287–296, 2000. 3

[51] Xide Xia, Meng Zhang, Tianfan Xue, Zheng Sun, Hui Fang, Brian Kulis, and Jiawen Chen. Joint bilateral learning for real-time universal photorealistic style transfer. In *European Conference on Computer Vision*, pages 327–342. Springer, 2020. 3

[52] Fanbo Xiang, Zexiang Xu, Milos Hasan, Yannick Hold-Geoffroy, Kalyan Sunkavalli, and Hao Su. NeuTex: Neural texture mapping for volumetric neural rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7119–7128, 2021. 2, 3

[53] Jaejun Yoo, Youngjung Uh, Sanghyuk Chun, Byeongkyu Kang, and Jung-Woo Ha. Photorealistic style transfer via wavelet transforms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9036–9045, 2019. 3, 15

[54] Kai Zhang, Nick Kolkin, Sai Bi, Fujun Luan, Zexiang Xu, Eli Shechtman, and Noah Snavely. Arf: Artistic radiance fields. *arXiv preprint arXiv:2206.06360*, 2022. 2, 3, 7

[55] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. NeRF++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 2, 3

[56] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 586–595, 2018. 9

[57] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Transactions on Graphics (TOG)*, 2018. 2

A. Supplementary Material

A.1. Detailed Configuration of Neural Network

Tab. 3, 4 and 5 are the detailed configurations of the neural networks used in our framework of universal photorealistic style transfer which shown in Fig.2. In these tables, OP refers to Operation, IN refers to number of the Input channels of the features, OUT refers to number of the Output channels of the features and ACT refers to the Activation function. Our HyperNet takes the features of the style image as input to control the weight of HyperLiner, so as to change the color style of the scene. Therefore, the number of output channels in HyperNet is determined according to the number of output and input channels in HyperLiner. For example, the number of output channel of the Hyper0 layer in HyperNet is 5120, which is determined by the number of input channel IN (39) and the number of output channel OUT (128) of the 0 layer in hyperliner.

Table 3. Detailed configuration of RGBNet.

Layers	OP	IN	OUT	ACT
0	Linear	39	128	ReLU
1	Linear	128	128	ReLU
2	Linear	128	3	

Table 4. Detailed configuration of HyperNet.

Layers	OP	IN	OUT	ACT
Hyper0	Linear	512	64	ReLU
	Linear	64	64	ReLU
	Linear	64	5120(39*128+128)	ReLU
Hyper1	Linear	512	64	ReLU
	Linear	64	64	ReLU
	Linear	64	16512(128*128+128)	ReLU
Hyper2	Linear	512	64	ReLU
	Linear	64	64	ReLU
	Linear	64	16512(128*128+128)	ReLU
Hyper3	Linear	512	64	ReLU
	Linear	64	64	ReLU
	Linear	64	8256(64*128+64)	ReLU
Hyper4	Linear	512	64	ReLU
	Linear	64	64	ReLU
	Linear	64	2535(64*39+64)	ReLU

Table 5. Detailed configuration of HyperLiner.

Layers	OP	IN	OUT	ACT
0	BatchLinear	39	128	ReLU
1	BatchLinear	128	128	ReLU
2	BatchLinear	128	128	ReLU
3	BatchLinear	128	64	ReLU
4	BatchLinear	64	39	ReLU

Tab. 6 and 7 show the detailed configurations of our 2D photorealistic stylization framework YUVStyleNet which shown in Fig. 3. k refers to the kernel size of the convolution and s refers to the stride size.

Table 6. Detailed configuration of Convolutional Network in YUVStyleNet.

Layers	OP	IN	OUT	k	s	ACT
Conv5	Conv2d	512	16	3	1	LeakyReLU
	Conv2d	16	256	3	1	Sigmoid
Conv4	Conv2d	512	16	3	1	LeakyReLU
	Conv2d	16	128	3	1	Sigmoid
Conv3	Conv2d	256	16	3	1	LeakyReLU
	Conv2d	16	64	3	1	Sigmoid
Conv2	Conv2d	128	16	3	1	LeakyReLU
	Conv2d	16	3	3	1	Sigmoid
Conv1	Conv2d	6	16	3	1	LeakyReLU
	Conv2d	16	3	3	1	Sigmoid

Table 7. Detailed configuration of Splatting Blocks in YUVStyleNet.

Layers	OP	IN	OUT	k	s
SB1	low-res style				
	ReflectionPad2d	3	3	/	/
	Conv2d	3	3	3	1
	low-res content				
	ReflectionPad2d	3	3	/	/
	Conv2d	3	3	3	1
	AdaIN				
SB2	SF4				
	ReflectionPad2d	64	64	/	/
	Conv2d	64	64	3	1
	CF4				
	ReflectionPad2d	64	64	/	/
	Conv2d	64	64	3	1
	AdaIN				
SB3	SF3				
	ReflectionPad2d	128	128	/	/
	Conv2d	128	128	3	1
	CF3				
	ReflectionPad2d	128	128	/	/
	Conv2d	128	128	3	1
	AdaIN				
SB4	SF2				
	ReflectionPad2d	256	256	/	/
	Conv2d	256	256	3	1
	CF2				
	ReflectionPad2d	256	256	/	/
	Conv2d	256	256	3	1
	AdaIN				
SB5	SF1				
	ReflectionPad2d	512	512	/	/
	Conv2d	512	512	3	1
	CF1				
	ReflectionPad2d	512	512	/	/
	Conv2d	512	512	3	1
	AdaIN				

In Tab. 7, we use adaptive instance normalization (AdaIN) [19] to fuse s feature and c feature from the splatting block module. Specifically, let $s, c \in \mathbb{R}^{C\times H\times W}$, then AdaIN is defined as:

$$\mathrm{AdaIN}(c, s) = \sigma(s)\frac{c-\mu(c)}{\sigma(c)} + \mu(s) \qquad (14)$$

where $\mu(c)$ and $\sigma(c)$ (resp. $\mu(s)$ and $\sigma(s)$) are the mean and standard deviation of c (resp.s) over its spatial dimension.

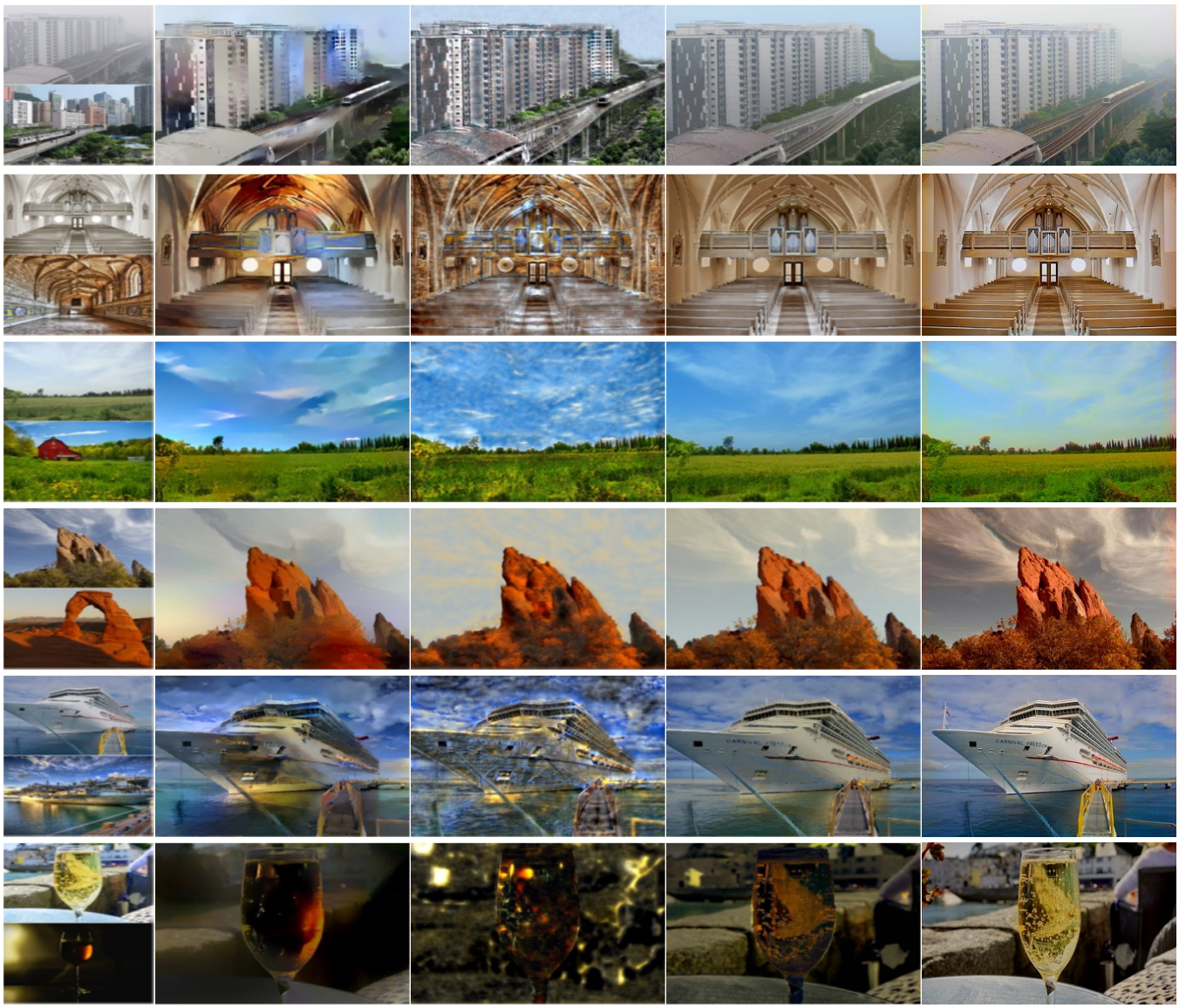

Figure 13. YUVStyleNet qualitative comparison for 2D photorealistic stylization. Our method against three state of the art baselines on some challenging examples.

A.2. Additional Visual Results

We show more results, including comparing 2D photorealistic stylization methods and more stylized results of 3D scenes. Fig. 13 is a comparison between our designed 2D photorealistic stylization method and other 2D photorealistic stylization methods. Our results have better visual quality than others.

Fig. 14, 15, 16, 17, 18, 19 and 20 shows more photorealistic stylization results of fern, flower, leaves, orchids, room, trex and horns 3D scenes respectively with different style images on Local Light Field Fusion(LLFF) [34] dataset.

Fig. 21, 22, 23, 24, 25 and 26 shows more photorealistic stylization results of chair, lego, hotdog, mic, drums and ficus 3D scenes respectively with different style images on NeRF-Synthetic [35] dataset.

From these results, we can see that the color features of different style images will change the color of the 3D scene, which realizes the photorealistic style transfer of the 3D scene and ensures consistency in space.

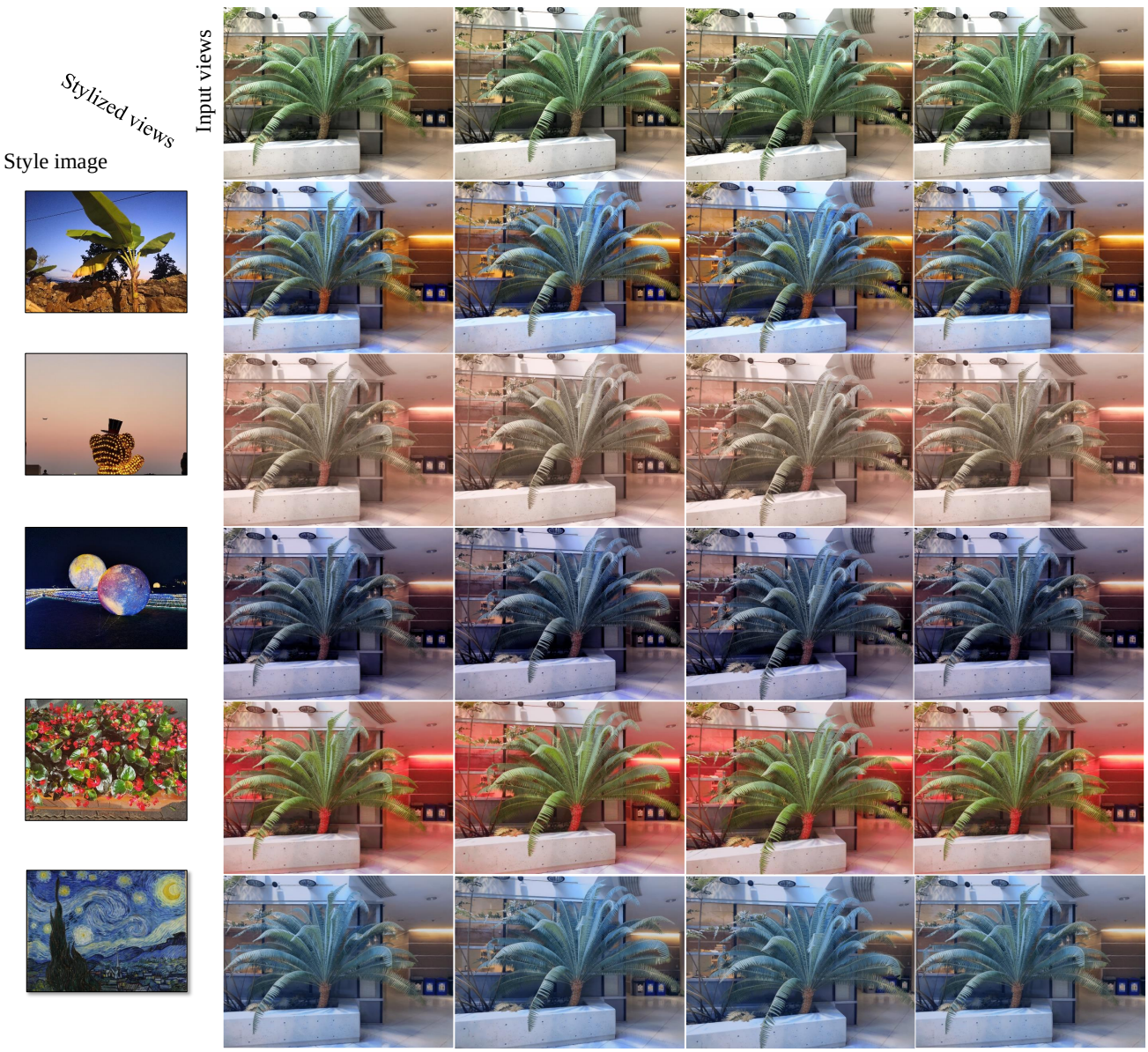

Figure 14. Photorealistic stylization results with the fern 3D scene on LLFF dataset.

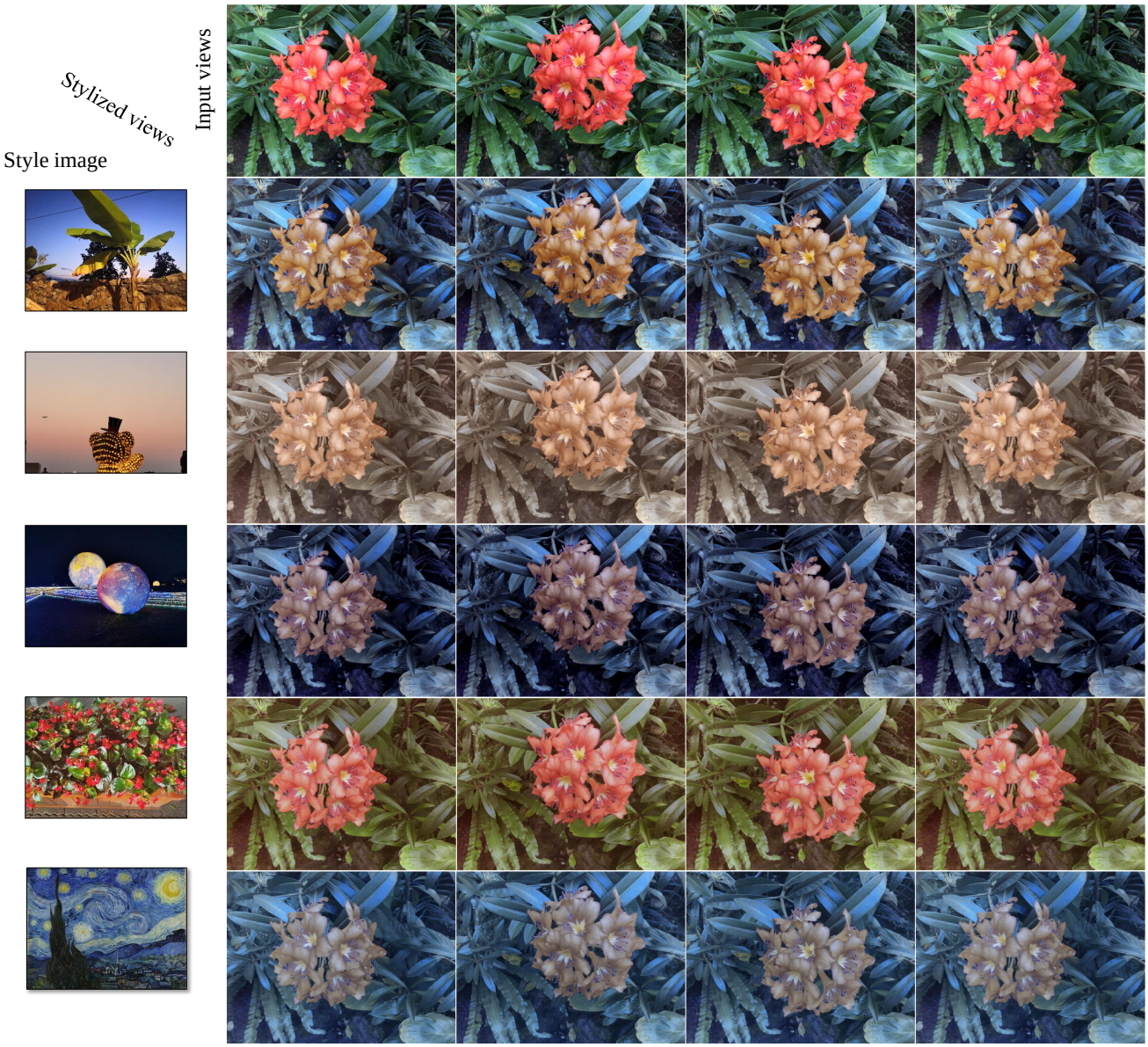

Figure 15. Photorealistic stylization results with the flower 3D scene on LLFF dataset.

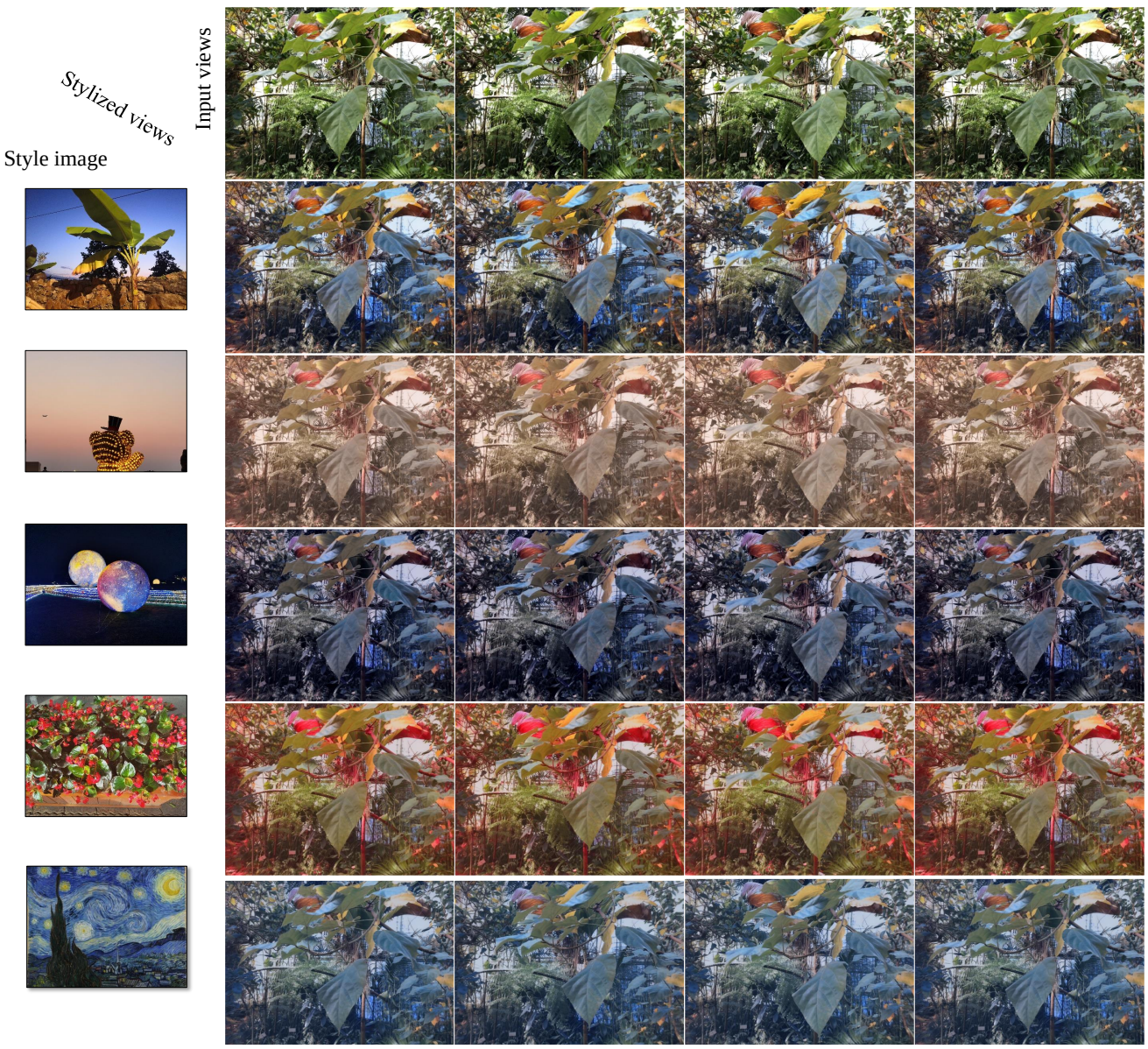

Figure 16. Photorealistic stylization results with the leaves 3D scene on LLFF dataset.

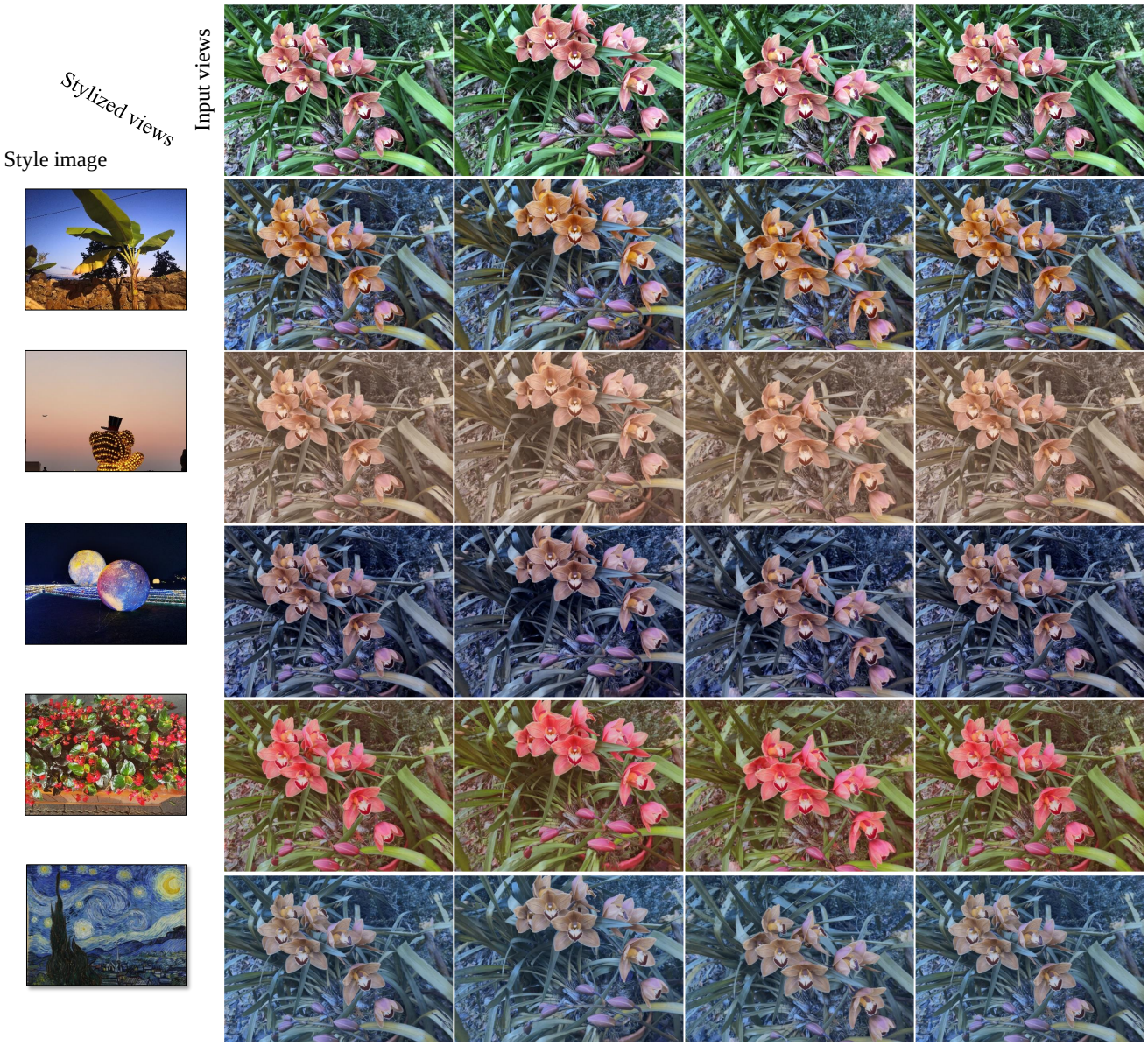

Figure 17. Photorealistic stylization results with the orchids 3D scene on LLFF dataset.

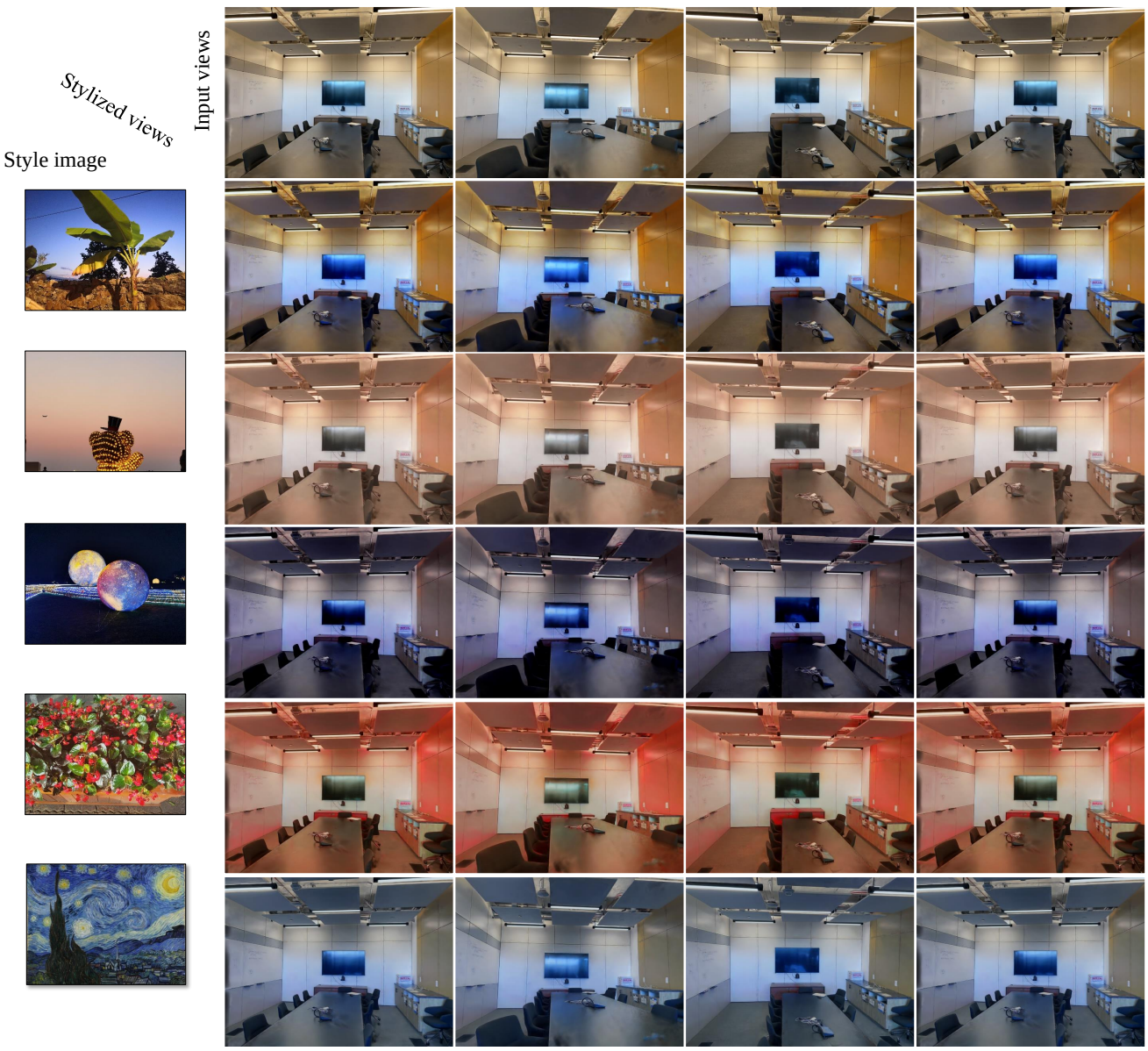

Figure 18. Photorealistic stylization results with the room 3D scene on LLFF dataset.

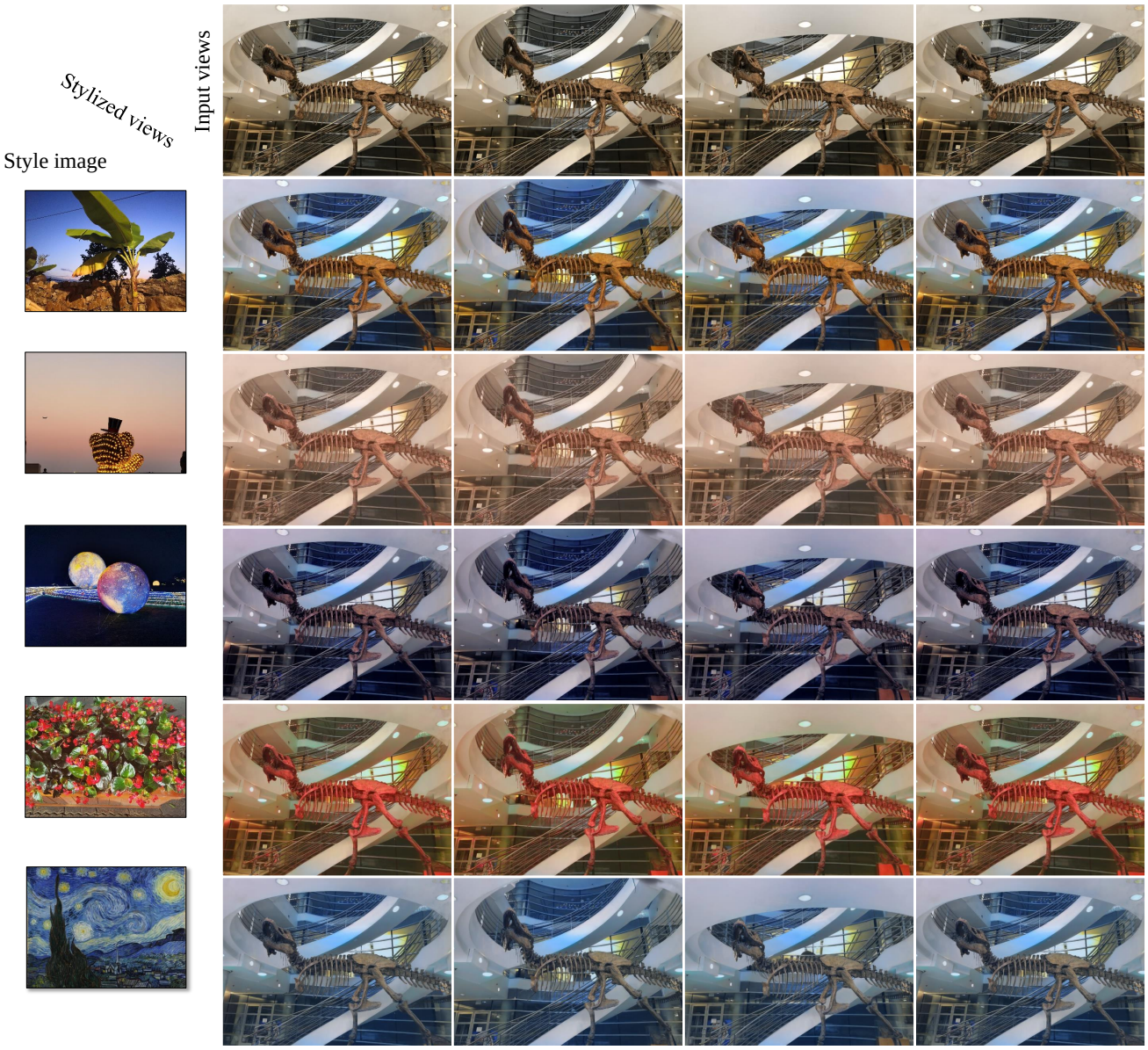

Figure 19. Photorealistic stylization results with the trex 3D scene on LLFF dataset.

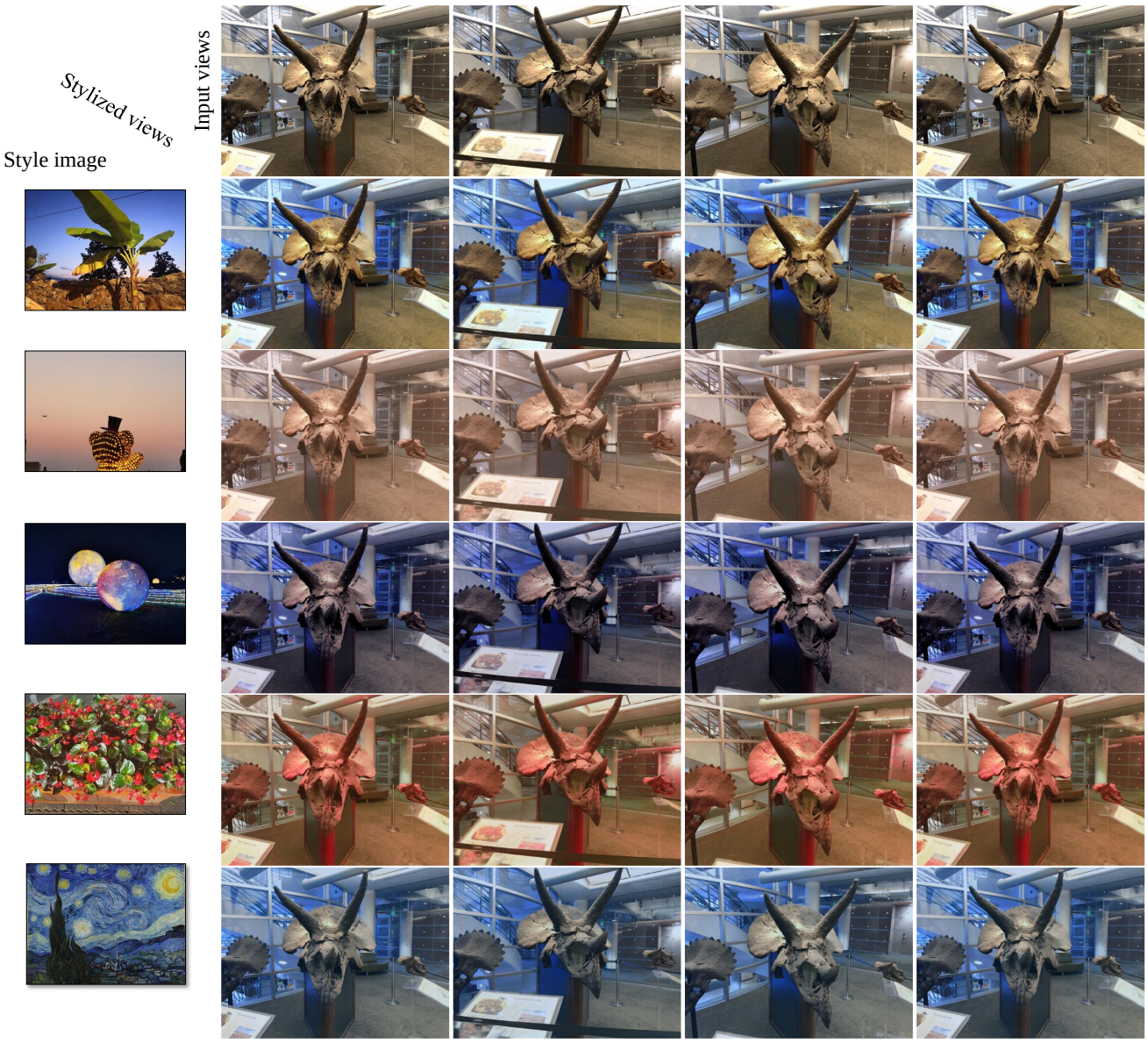

Figure 20. Photorealistic stylization results with the horns 3D scene on LLFF dataset.

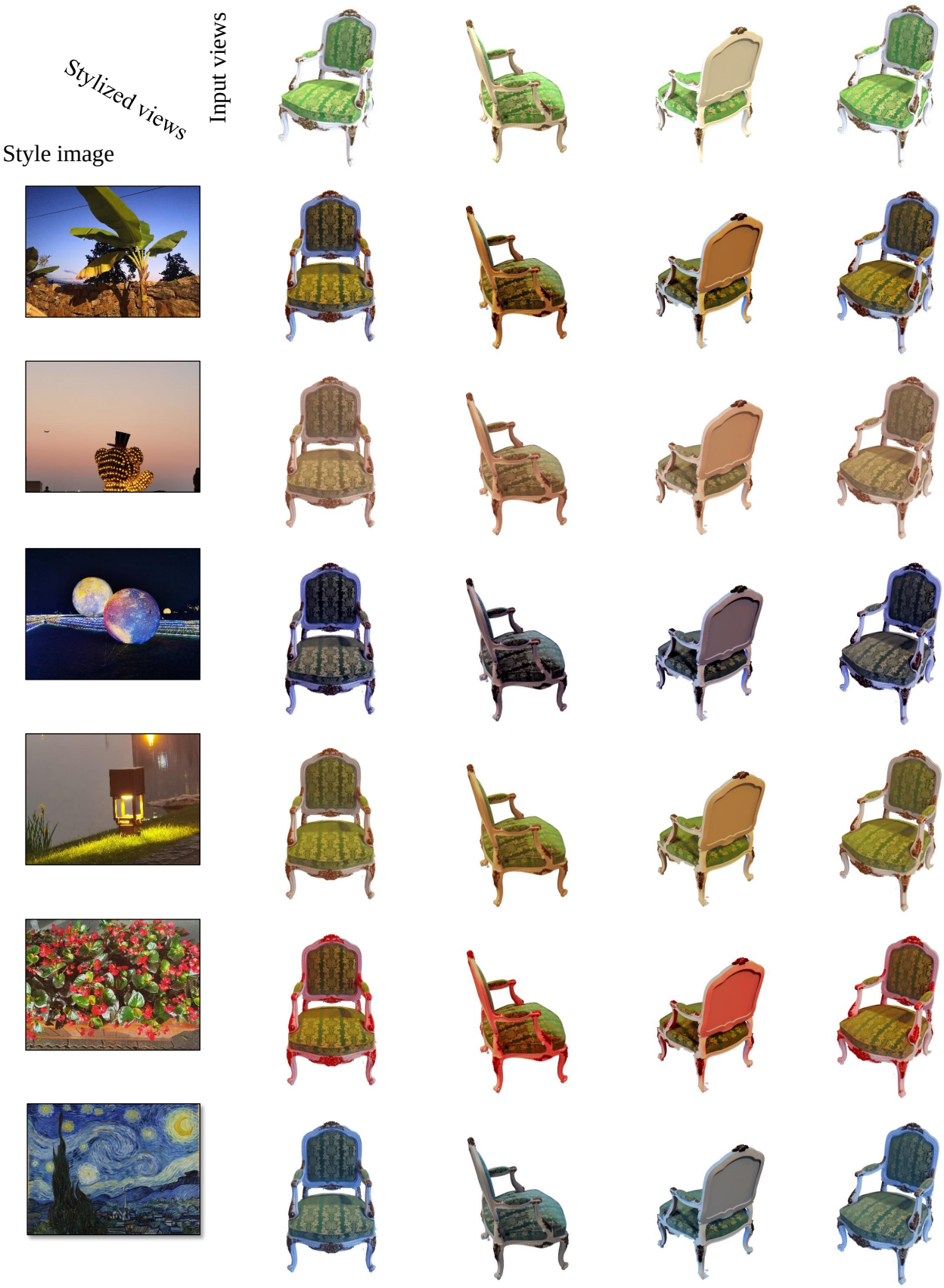

Figure 21. Photorealistic stylization results with the chair 3D scene on NeRF-Synthetic dataset.

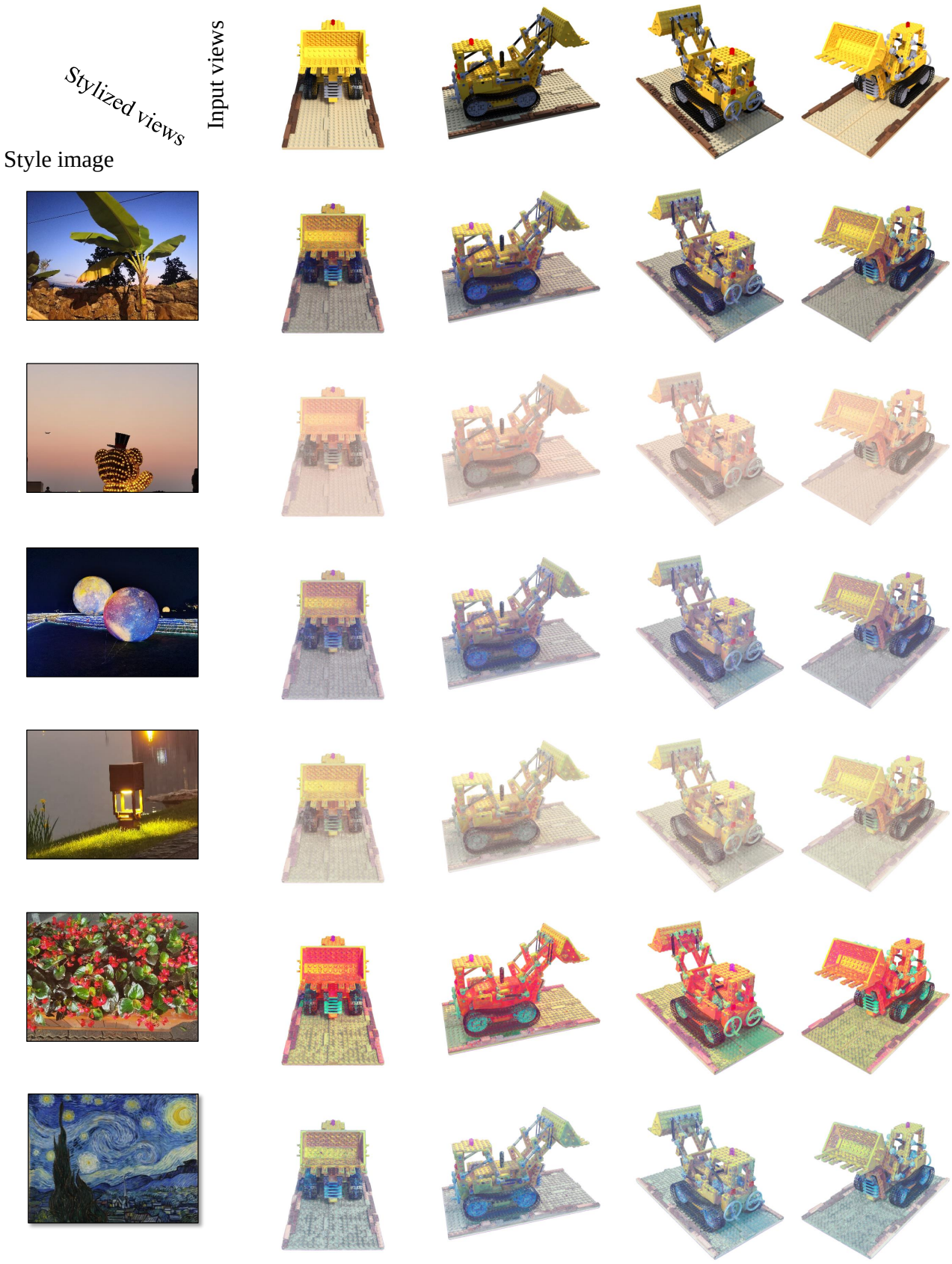

Figure 22. Photorealistic stylization results with the lego 3D scene on NeRF-Synthetic dataset.

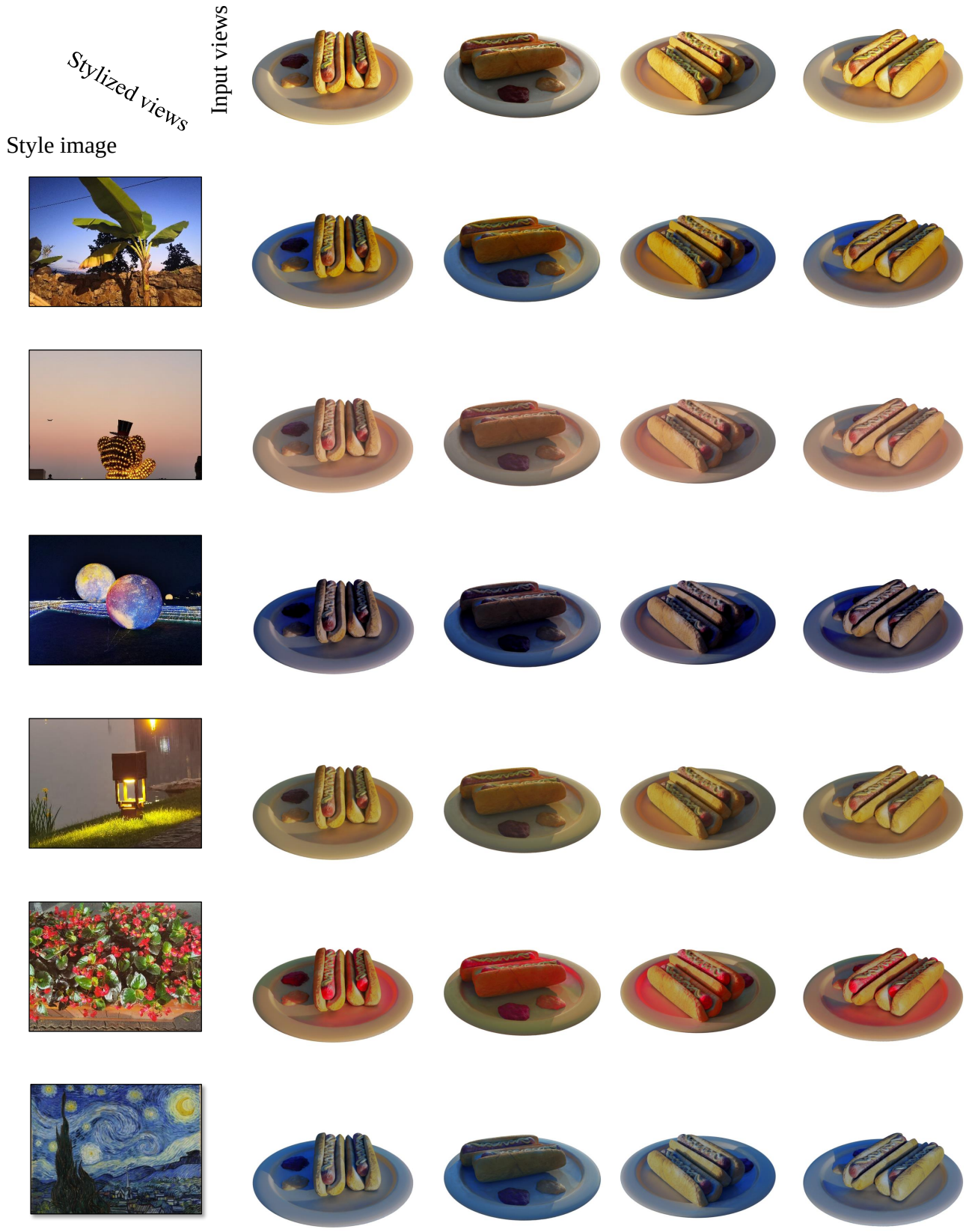

Figure 23. Photorealistic stylization results with the hotdog 3D scene on NeRF-Synthetic dataset.

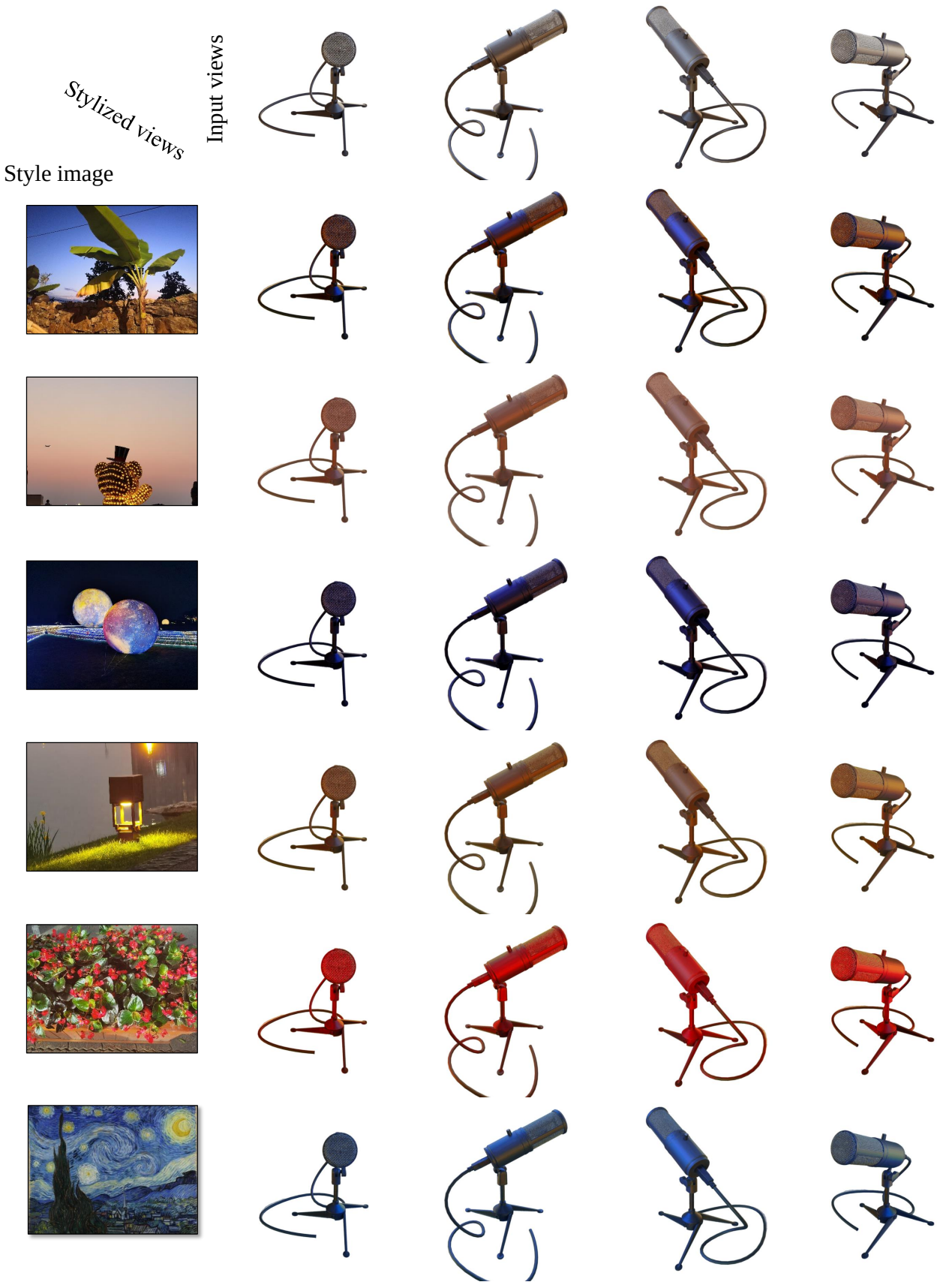

Figure 24. Photorealistic stylization results with the mic 3D scene on NeRF-Synthetic dataset.

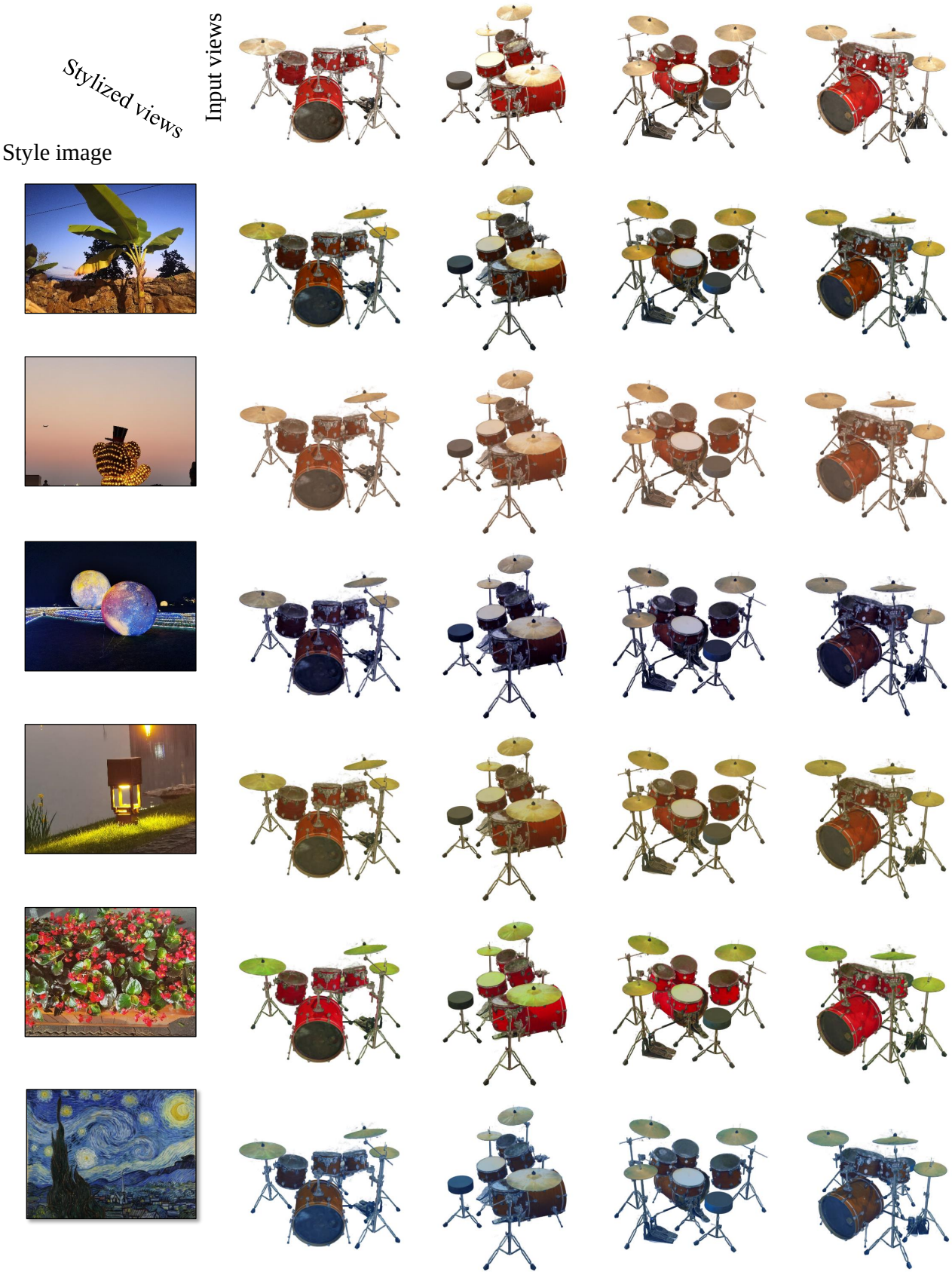

Figure 25. Photorealistic stylization results with the drums 3D scene on NeRF-Synthetic dataset.

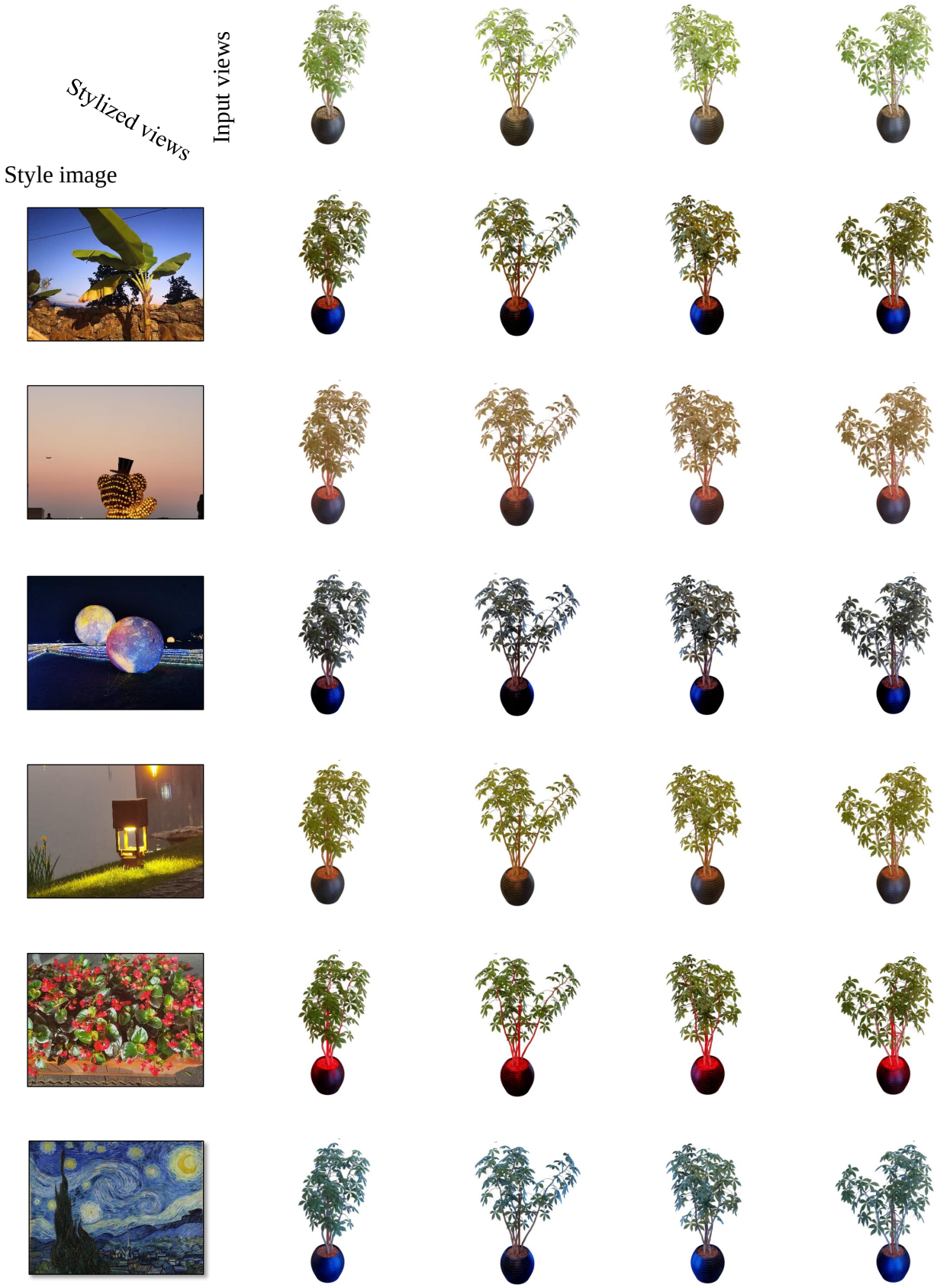

Figure 26. Photorealistic stylization results with the ficus 3D scene on NeRF-Synthetic dataset.