This CVPR workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

DeSRF: Deformable Stylized Radiance Field

Shiyao Xu
Wangxuan Institute of Computer Technology
Peking University
xusy@stu.pku.edu.cn

Lingzhi Li
Alibaba Group
Beijing, China
llz273714@alibaba-inc.com

Li Shen
Alibaba Group
Beijing, China
jinyan.sl@alibaba-inc.com

Zhouhui Lian
Wangxuan Institute of Computer Technology
Peking University
lianzhouhui@pku.edu.cn

Figure 1. Given a style exemplar and a realistic scene, our method can synthesize the 3D artistic scene with the target geometry. Our method not only learns textures, but also learns the geometry of the reference artworks into 3D scenes, such as the distortions in "The Scream".

Abstract

When stylizing 3D scenes, current methods need to render the full-resolution images from different views and use the style loss, which is proposed for 2D style transfer and needs to be calculated on the whole image, to optimize the stylized radiance fields. It is quite inefficient when we need to stylize a large-scale scene. This paper proposes a more efficient method, DeSRF, to stylize the radiance fields, which also transfers style information to the geometry according to the input style. To achieve this goal, on the one hand, we first introduce a deformable module, which can learn the geometric style contained in the input style image and transfer it to radiance fields. On the other hand, although the style loss needs to be calculated for the entire image, actually we do not need to process all the rays when updating the stylized radiance fields. Motivated by this observation, we propose a new training strategy called Dilated Sampling (DS) for efficient stylization propagation. Experimental results show that our method works more efficiently and produces more visually-reasonable stylized 3D scenes with geometry style information compared to other existing approaches.

1. Introduction

In the past few years, research progress on image stylization has grown successfully and spawned many popular stylization applications. With the development of 3D scene generation technology [31] and the applications such as AR/VR, stylization for 3D scenes naturally drew attention from the research community. Style Transfer, first introduced by [10], is the task of creating synthetic images that simultaneously match the semantic content of the input image and the aesthetic style of a given artwork. Besides low-level style like color and texture, geometric trait of the artistic images is also a critical feature that needs to be transferred [18, 28]. For example, the bold distortion of the human face had been seen as a major symbol of the famous artwork "The Scream".

3D scenes, represented by point clouds, voxels, or neural radiance fields, have obvious 3D multi-view continuity. When stylizing a 3D scene, it is not only necessary to consider the visual quality of stylization but also essential to ensure multi-view consistency. Current works [5, 33, 47] combine the optimization-based neural style transfer method in the 2D domain with the neural radiance fields. They do maintain multi-view consistency and successfully capture the target style to some degree. However, the generated results look more like directly sticking texture to the surface of objects in the scene rather than an artwork created with the designated style. Similarly, just like the rendering process in neural radiance fields, stylizing a 3D scene is very time-consuming and requires a lot of GPU memory. This is because that style loss needs to be calculated on the full-resolution image, which means we need to render $\mathbf{H} \times \mathbf{W}$ rays and store all the intermediate results in one forward and backward pass. This drawback makes the training procedure extremely slow when facing high-resolution multi-view input.

Chiang et al. [5] address this issue by only rendering local patch rays during training time for style transfer. However, this patch-based method would break global style consistency and is prone to patch mismatching. ARF [47] also adopts a similar idea of decomposing the whole image into different batches, designs a different style loss matched on the nearest neighbor features, and proposes a deferred back-propagation mechanism, allowing the loss backward propagating on the full-resolution image. Though this kind of deferred back-propagation mechanism successfully avoids OOM problem, it is still not efficient enough as it requires redundant forward pass. SNeRF [33] splits the 3D scene style transfer process into NeRF optimization step and stylization optimization steps, and excute the two step in a round-robin style. This modification makes the training scheme more GPU-friendly, as it uses GPU memory efficiently, but the whole training process is still very time-consuming. What's more, existing methods [5, 33, 47] still do not learn the geometric information well from input example style into the 3D scene.

We propose an efficient 3D deformable stylization method, called **DeSRF**. Our method not only learns the innate geometric information of the input style image via a deformation module but also introduces a more efficient optimization methodology for radiance field stylization from the perspective of sparse grid updating and ray sampling, named dilated sample.

Our method learns geometric changes in the style image to make the stylized 3D scene deformed more reasonable and closer to the artistic style. At the same time, from the perspective of network efficiency, our dilated sampling can significantly reduce the times of forward-backward propagation to the radiance field, making the stylization more efficient without losing overall structural information and details.

The main contributions of our DeSRF are the followings:

- We propose a novel framework for 3D scene stylization that not only learns the style representation but also the geometric changes that also be an important component of style using a deformable network.
- We are the first to produce reasonable and deformed style scenes for radiance fields.
- We introduce an efficient training strategy and dilated ray sampling that alleviates both the memory and time burden when stylizing the radiance fields.
- Our DeSRF can stylize high-resolution scenes like images at size 756×1008 and produce high-quality, multi-view continuous results.

2. Related Work

2.1. Neural Radiance Field

Radiance Fields reconstruction is a classic task that synthesizes a scene from a set of images capturing the scene to the unobserved viewpoints. Unlike traditional explicit methods using point-clouds [7, 37], voxel grids [12, 13, 44] or octrees [40, 42], NeRF [31] uses a fully-connected MLP and a continuous volumetric representation function, to project the output colors and densities into an image. The success of NeRF spurred growing attention and interest on 3D scenes research, such as NeRF editing [20, 23, 27, 46], dynamic scene modeling [9, 24, 29, 36, 43] and fast rendering [3, 8, 26, 32, 39, 48]. It takes at least 12 hours in the original NeRF to reconstruct a scene, but now, it only takes several seconds [32], which makes us see a lot of potential in the neural radiance field. In this paper, we aim to stylize the radiance field with a deformation network and a special strategy when stylizing.

2.2. Image Style Transfer

Style transfer aims at creating synthetic images with the aesthetic style of the given style images and maintaining their content. Starting from neural style transfer [10], [11], it is usually a process of optimizing the output image according to the Gram matrix loss and the content loss calculated on the VGG-Net [38] extracted features. Research on image style transfer mainly starts from the improvement of gram loss and proposing some different training processes. The original optimization-based process [10, 11] is time-consuming and typically requires 200-300 iterations to converge. Subsequently, Perceptual Loss [17], [6, 16, 34], etc. proposed a feed-forward transfer method, fixed the Loss Network, and transfer the input images using a single forward pass, which makes it faster when stylizing. Recent

work on style loss improvements [21,25] replaces the global gram matrix with the nearest neighbor feature matrix, for better texture preservation. Some other methods generate the image using patch matching, not the optimization for the entire image, like [1,41]. These image stylization methods provide a good reference when we perform the stylizing on 3D scenes.

2.3. 3D Scene Style Transfer

3D scene style transfer aims to transfer the style into the scene with the reference appearance and multi-view consistency. Similar to 3D scene synthesis tasks, 3D scene stylization can be related to image style transfer tasks and different 3D scene representations, like the explicit expression as the points cloud, SDF, mesh, etc., and the implicit expression such as NeRF. In the following parts, we will take it as a task that combined the 3D scene generation with the way of image style transfer. There are several works dedicated to it. [2,15,22] and [14,45] stylize the points cloud and mesh respectively. Such results are not as good as the results of the radiance field, since they are limited by the 3D scene expression they used. However, it is necessary to do a good trade-off between rendering the entire image, which is time-consuming and high-GPU-memory demanded, and the high quality of stylization results, which works better on the entire image. Chiang et al. [5] render patches for style transfer, which sacrifices the quality of stylized scenes. ARF [47] accumulates the gradient, and SNeRF [33] repeats the rendering process and the stylization process. The two closest works to ours are ARF and SNeRF. In the following experiments, we will mainly focus on these two methods for comparison.

3. Method

Given radiance fields reconstructed with a set of real-world photos and an arbitrary artistic image, we not only want to transfer the reference style texture to the scene but also transfer the geometric structure of the target style while still keeping the semantic content of the scene recognizable. Following the previous 3D stylizing works [4, 33, 47], we start from a well-trained radiance field and then fine-tune the radiance fields with carefully designed style loss. The main contributions lie in our highly efficient optimization pipeline equipped with dilated ray sampling and a geometric deformation module that captures the structural characteristics of the style image. Fig. 2 shows an overview of our proposed method. We will introduce our DeSRF in detail in this section.

3.1. Overview and Formulation

The essential idea beneath neural radiance field(NeRF) is to learn a continuous mapping from the 3D coordinates $\mathbf{x} = (x, y, z)$ and view direction $\mathbf{d} = (\theta, \phi)$ to the volume density σ and view-dependent color $\mathbf{c} = (r, g, b)$. which can be formulated as $F_\Theta : (\mathbf{x}, \mathbf{d}) \mapsto (\sigma, \mathbf{c})$. According to volume rendering technique The expected color $\hat{C}(\mathbf{r})$ of camera ray $\mathbf{r} = \mathbf{o} + t\mathbf{d}$ for the pixel is then rendered as:

$$\hat{C}(\mathbf{r}) = \sum_{i=0}^{N-1} T_i(1 - \exp(-\sigma_i\delta_i))\mathbf{c}_i, \quad \text{where } T_i = \exp(-\sum_{j=0}^{i-1}\sigma_i\delta_i) \tag{1}$$

Algorithm 1 Pseudocode of our DeSRF process in a PyTorch-like style

```
# DeSRF: our network
# rays: total rays for training
# steps: training steps
DeSRF.initial()  # network initialize
DeSRF.deform_net_initial()  # deform net initialize
for i_init in range(steps):
    full_rays = rays.sample()
    full_rays_deformed = DeSRF.deform_net(full_rays)
    rgb = DeSRF.volume_render(full_rays_deformed)
    loss = compute_reconstruction_loss(rgb, rgb_gt)
    optimizer.zero_grad()
    loss.backward()
    optimizer.steps()
for i_style in range(steps):
  # dilated sampling from the training rays
    full_rays_rays = rays.dilated_sample()
    full_rays.requires_grad = False

    dilated_rays = full_rays.dilated_sample()
    dilated_rays.requires_grad = True

    full_rays[dilated_position] = dilated_rays
    full_rays_deformed = DeSRF.deform_net(full_rays)
    rgb = DeSRF.volume_render(full_rays_deformed)
    loss = compute_stylization_loss(rgb, rgb_gt)
    optimizer.zero_grad()
    dilated_rays.backward()
    optimizer.steps()
```

where δ_i denotes the distance between the adjacent points. The radiance field stylizing can be seen as an optimization process, where the mapping function F_Θ is learned given the rendered full-resolution image $\{x_i\}_{i=1}^N$ with different camera poses $\{\theta_i\}_{i=1}^N$ and the style image, to generate the stylized scene.

In order to transfer the geometric trait from the style image to the target 3D scene, we introduce a deformation network (Sec. 3.2) to learn a suitable deformation on the points sampled along the camera rays following the network design methodology from dynamic neural radiance

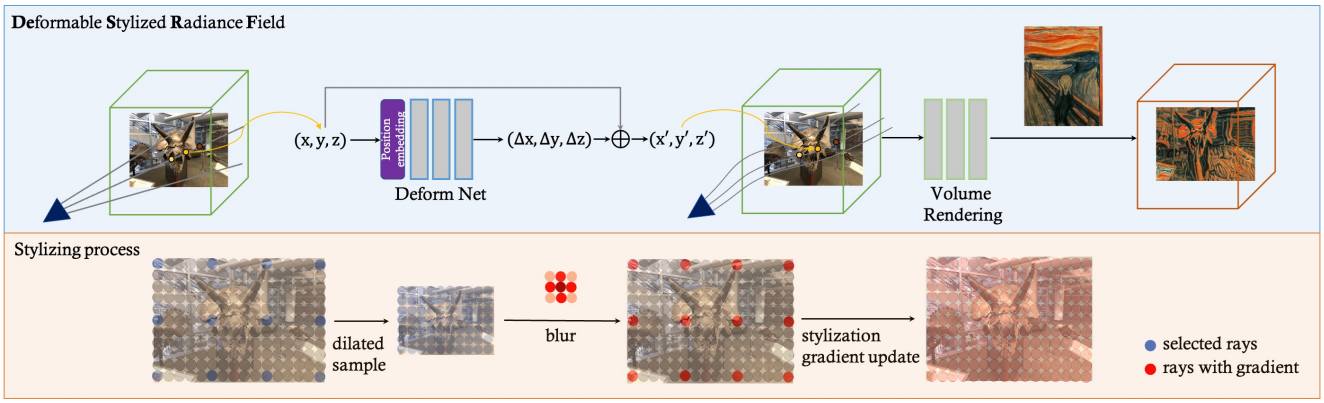

Figure 2. Overview of our **DeSRF**. We introduce a Deform Net into the Radiance Field when stylizing 3D scenes. The deformation module will learn the coordinate changes of the rays, which will further change the geometric shape of the scene. We also introduce a dilated ray sample method that sampled a set of rays after the deformation module, set with gradients, for calculating the style loss, and then average the gradient to the total image for updating the network. The second half of the image shows the sampling process of the dilated sample during the stylization process. Where the blue point represents the selected rays, and the red point represents the process of applying gradients to ray for updating the entire network.

field [35, 36] and NeRF editing tasks [46]. Hence, the proposed deformation network can modify the global geometry manifold by adjusting each sample location during a ray casting. The overview design of our DeSRF is illustrated in Fig.2. We train the network by the constraints of rays' distortion and the style-transfer-related loss performed on the rendered images. And the geometric features of the input style are learned by the overall training process.

However, the style loss computation demands the entire image to be rendered, where requiring $N_{points} \times H \times W$ times network forward-propagating every iteration in the volume rendering process, makes the training pipeline unacceptable memory costing and exceptionally time-consuming. We then investigate the sampling and gradient updating process when training our DeSRF. To alleviate this overhead, an intuitive idea is to minimize the number of total rays that must be rendered. In the realistic scene reconstruction process, the network needs as many points and rays as possible to calculate the pixel value according to Eq. 1 to render the image with more details. When stylization, we found that the stylized results frequently have repetitive textures, with some lower details maintained but more like the input style. Therefore, we make an assumption: when updating the stylized radiance field, it actually does not need so many rays and points. Starting from the signal sampling principle and the design of fast NeRF, we explored the influence of the numbers of rays and different ray sampling methods on updating the stylized radiance field, and propose our dilated sample (Sec. 3.3). The experimental results verify the effectiveness of our assumption.

3.2. Deformation Radiance Fields

As illustrated in Fig. 2, we introduce a Deform Network $\mathcal{D}(\cdot)$ performing on the points sampled from the rays.

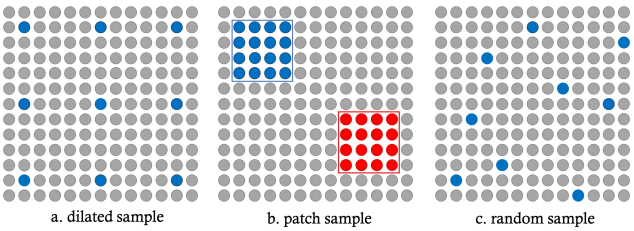

Figure 3. Different ray sample methods when training. a. indicates uniform and equidistant sampling for every N_c rays in the ray array. b. random select a patch with $\frac{H \times W}{N_c{}^2}$ size from the rays array. c. random selected $\frac{H \times W}{N_c{}^2}$ from total rays.

Deform Network $\mathcal{D}(\mathbf{x})$ is estimated with a 3-layers MLP, which refers to the design of NeRF [31]. Given a 3D point $\mathbf{x}$, $\mathcal{D}(\mathbf{x})$ learns geometry coordinate changes $\Delta\mathbf{x}$ for each point sampled on a ray and then outputs $\mathbf{x} + \Delta\mathbf{x}$ for the latter appearance calculation. Due to the poor expression of 3-dimension points, we project the points into a higher-dimensional space first with a set of sine and cosine functions $\gamma : \mathbb{R}^3 \rightarrow \mathbb{R}^{3+6L}$ of increasing frequencies, which is actually same as used in NeRF [31]:

$$\gamma(p) =< (sin(2^l \pi p), cos(2^l \pi p)) >_0^L \qquad (2)$$

for $\mathbf{x} \in \mathbb{R}^3$, we use $L = 5$ for the best performance. The 3D points will be mapped into a 256-dimensional space and then encoded back to 3D to get the Δx. The distorted coordinates $(x + \Delta x)$ are next sent to the radiance field, and the color $\mathbf{c}$ and density σ are sampled from the radiance feature. The rendered image x_i is then used for stylization optimization.

3.3. Dilated Sampling

When optimizing the stylized radiance field, due to the limitation of style loss that needs to be calculated on the entire image, a straightforward idea is to render the whole image completely. However, as discussed before, both ARF [47] and SNeRF [33]'s training strategies require multiple forward and backward updates to all rays at once. Note that the ray sample here is not the *ray selection process* when the radiance field is forward propagating. It is actually the process of gradient backward propagation, in which rays are selected to be calculated for the gradient and loss.

In fact, when realistic-scenes 3D rendering, it uses the image x_i rendered in some view θ to calculate the loss and gradient for updating the entire radiance field, which can be regarded as using a part of the total rays (for $N \times H \times W$) to update the entire radiance field. The more rays for the better details. While for stylizing, not so many rays are needed due to the abstract, stylized, and local repeated texture. Also comes from the sampling process in traditional convolution, a "downsampled" rays collection can represent the whole image. We propose the strategy to select and update only part of the rays, whose gradient will be activated while the others will be masked, and the gradient will be averaged into the entire network to achieve the total rays updating. Note that although only part of the rays is updated, which may bring concerns with requiring more training steps, the experiments Sec. 4 show that no extra is needed.

We consider three different samplings as shown in Fig. 3–random sample, patch sample, and dilated sample when updating the network. Sec. 4.3.2 will show the results of different sampling strategies when backward propagation of the stylizing process. We found that the proposed dilated sample works better for the quality of rendered stylized images. Fig.3 illustrates how dilated sample works. For the size of $H \times W$ rays, dilated sampling random and uniform selects one for every N_c ray equidistantly along the side of the entire rays array. Here will get $\frac{H \times W}{{N_c}^2}$ rays with gradient, the remaining gradients will be masked out, and then the selected rays will be used to update the total network, which can be regarded as an average process. N_c here is a hyperparameter. The more detailed training process can be found in Sec. 3.4.

3.4. Train Strategy and Loss Functions

As is in previous work [33, 47], our training process also starts with realistic scene reconstruction. Here we use the pretrained TensoRF [3] model as a representative. What the difference is that we also need to initialize the deform module combined with the radiance field, where the deform module is constrained by the reconstruction of realistic images $\{X_i\}_{i=1}^N$:

$$L_{init} = ||X, \mathbf{DeSRF}(\mathbf{x})||_2 + L_{tv} \tag{3}$$

where we randomly selected rays from all-views images for training, and $\mathbf{DeSRF}(\mathbf{x})$ denotes the rendered result.

In the next stylizing step, we first dilated sampled a set of rays for loss calculation. Note that the others will be set with no gradient. Then the selected gradient will be plugged back into the full-size image and averaged with the total rays array to update the entire image. When the dilated rays are back-propagated if the *dilated_factor* N_c is small, which means it will propagate a some-kind-of large of rays backward, we will propagate in batches, like what the original NeRF [31] do when processing all of the rays.

We treat the deform net and the radiance field network as a total, just using the stylize loss $L_{style} + L_{content}$ to encourage the deformation, a deform regularization term L_{dx} to prevent the geometry from over changing, and a tv loss to encourage the smooth deformation. The total loss functions are the following:

$$L_{total} = \lambda_{sty} L_{sty} + \lambda_{con} L_{con} + \lambda_{dx} L_{dx} + \lambda_{tv} L_{tv} \tag{4}$$

where the style loss L_{style} used from what is used in NNST [21] and ARF [47]. The content loss $L_{content}$ comes from the original neural style transfer [10]. The deform net regularization term performs on the coordinate changes $\Delta\mathbf{x}$:

$$L_{dx} = \frac{{N_c}^2}{H \times W} ||\Delta\mathbf{x}|| \tag{5}$$

And the total variation loss L_{tv} is used to smooth the rendered $\{x_i'\}_{i=1}^N$ in 2D domain, which is computed by:

$$L_{tv} = \frac{1}{H \times W} ||\nabla_u(x_i') + \nabla_v(x_i')|| \tag{6}$$

where u and v denote horizontal and vertical directions, H, W is the size of the entire rays same as the image x_i. The total train process is described in Alg. 1.

4. Experiments

This section provides a thorough evaluation of our DeSRF. We will introduce the details of our experiments (Sec. 4.1), show our experimental results (see Fig. 4), also the results compared with the current state-of-the-art method in 3D scene stylization, like ARF [47] and SNeRF [33], and demonstrate the effectiveness of our deform net and the dilated sample.

4.1. Implementation Details

As is described in Sec. 3, our DeSRF can be represented by any kind of Radiance Field. We adopt TensoRF [3] as the Radiance Field. Limited by the application scenarios of TensoRF, we mainly use the LLFF dataset [30] to represent our DeSRF. Regarding the training process Alg. 1 of our DeSRF, we first initialize the deform net with steps=2000,

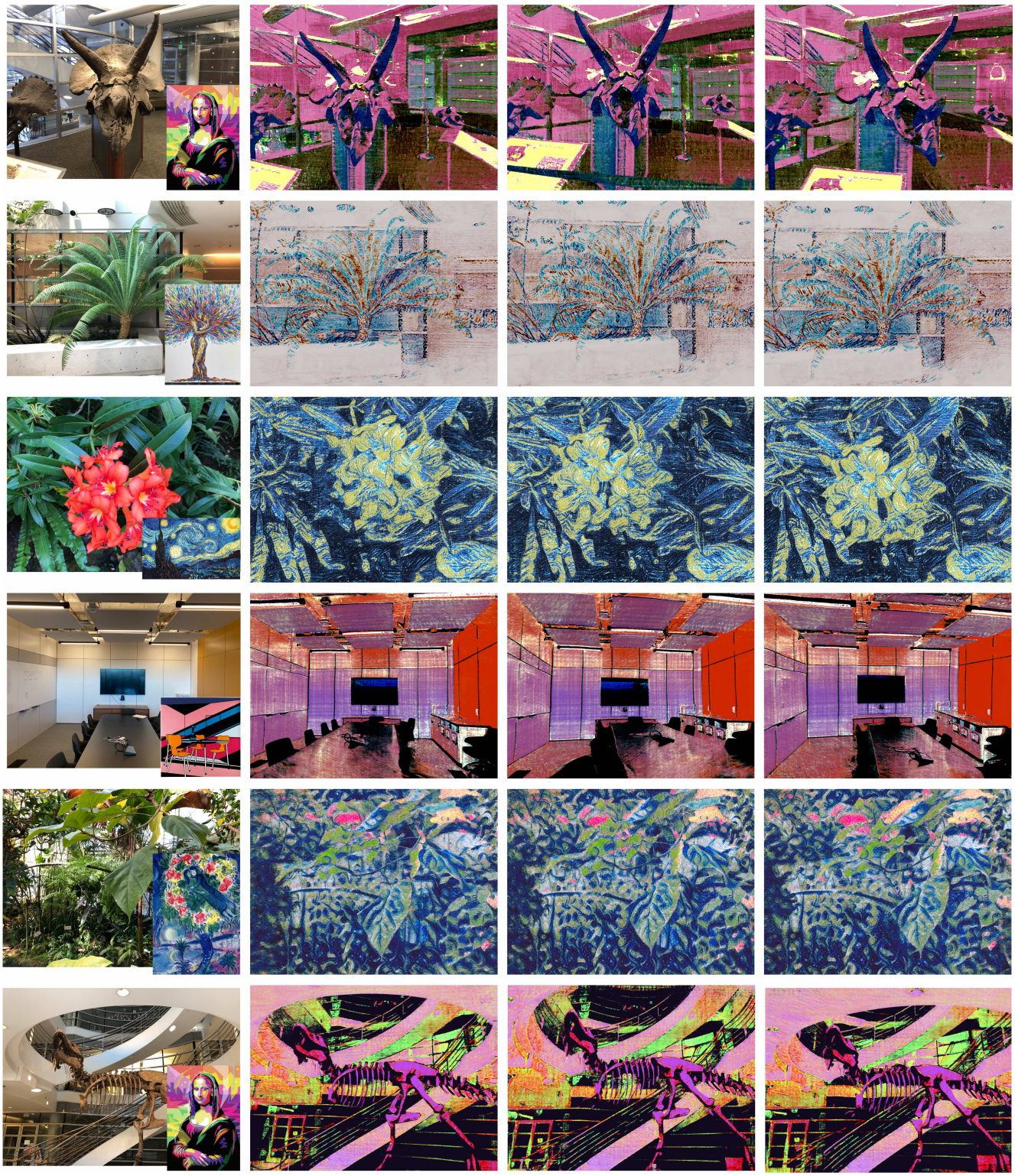

Figure 4. Results of our DeSRF in different scenes with different styles.

and the training process iterations is set to 40. The content features are extracted from the layer $conv\_3$ of the pre-trained VGG16 model as is suggested in ARF. We also use the L_{style} with the same setting in NNST [21] and ARF [47], also with the color preserve matrix in ARF [47]. When performing dilated sample for updating the styliza-

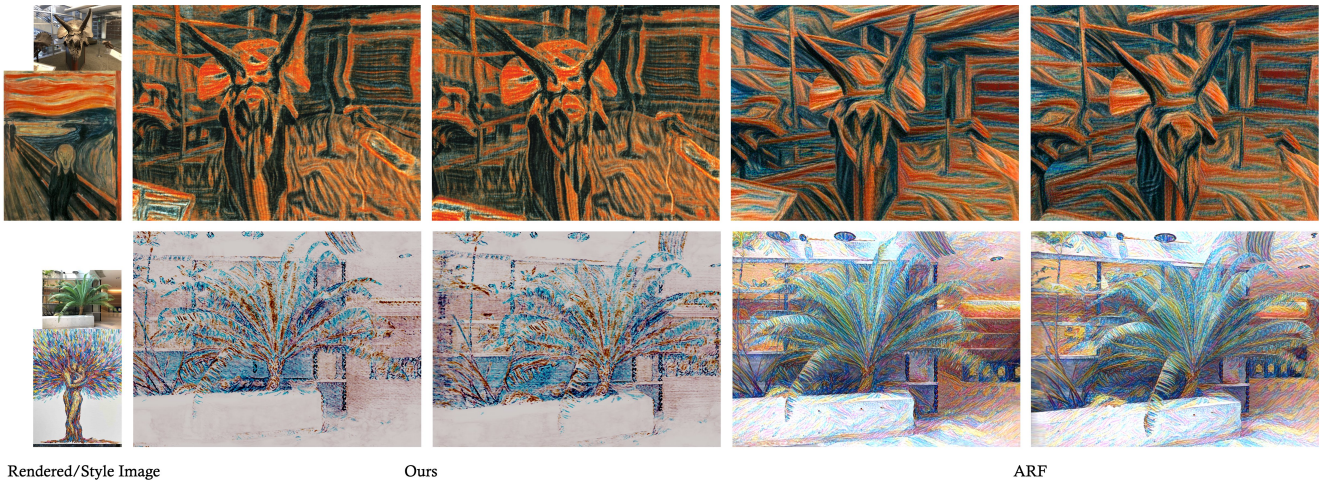

Figure 5. Our DeSRF results compares with ARF [47].

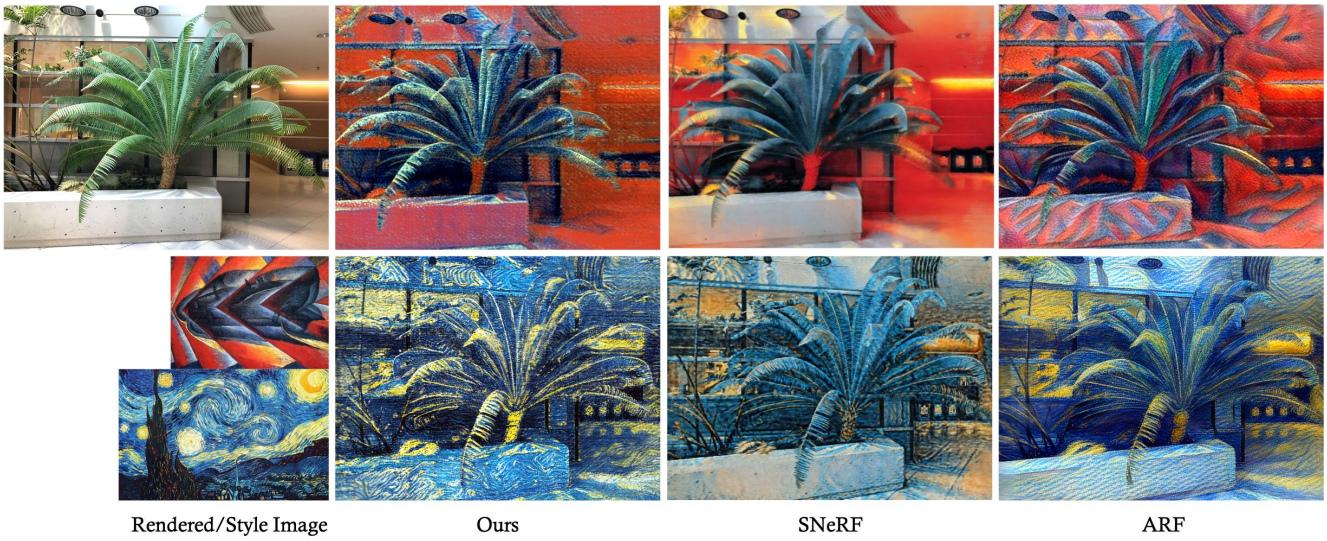

Figure 6. Comparison results with some state-of-the-art methods SNeRF [33], and ARF [47]. Since SNeRF did not release their code, the results of SNeRF are cropped from the paper.

Figure 7. our full DeSRF and the DeSRF w/o Deform Net.

tion network, we found it works better with the dilated factor $N_c = 2$ and the blur kernel size for 3. Adam optimizer [19] is adopted with $(\beta_1, \beta_2) = (0.9, 0.99)$ with the learning rate 0.02 for appearance features learning in TensoRF and 0.001 for the deform net. $(\lambda_{sty}, \lambda_{con}, \lambda_{dx}, \lambda_{tv})$ is set to (5, 0.005, 10, 0.5). We train all of our models on a single NVIDIA A100 GPU.

4.2. Comparison

In Fig. 6 and Fig. 5, we compare our results with current state-of-the-art 3D scene stylization methods. For SNeRF, Fig. 6, the synthesis texture is not exquisite enough and also does not transfer the complex textures like the twisted sky in Van Gogh's The Starry Night. It is also unable to learn the sharp edge features, such as the leaves. For ARF, Fig. 6, Fig. 5, although the results look more transparent, it learns more of the average texture of the whole image, which is not what we expect. In contrast, our deformable network learns not only finer details but also the overall geometric

Figure 8. Results of different ray sample methods in Sec. 3.3. From top to bottom are the results of random sampling, patch sampling, and out dilated sampling results.

style. Furthermore, the results show that the stylization results of current methods are all overly realistic, without abstract stylistic features, which are instead the strength of our deformable stylization method. For more results, please see our supplementary materials.

4.3. Ablation Studies

This section will demonstrate the effectiveness of our deform net and the dilated sample.

4.3.1 Deform Net

To verify the effectiveness of our proposed Deform Net, we evaluate the performance of our method by removing the deformable module. The results are shown in Fig. 7. The model w/o Deform Net clearly misses geometric changes, since there is no distortion of the ray coordinates, and the w/o model does not produce the desired results.

4.3.2 Different Sampling Method

To verify the effectiveness of our dilated sample, we compare the results (Fig. 8)of different sampling methods as described in Sec. 3.3. The sampling process is illustrated in Fig. 3.

Figure 9. The limitation of our DeSRF. Our stylized results are blurry when meeting some corner cases and repeating textures.

4.4. Discussion

Our DeSRF can learn the geometric changes from the referred style image, and synthesize more reasonable stylized scenes with better aesthetics. However, in our experiments, we found that the Deform Net is not always sensitive to all styles, and therefore the deformation effect is somewhat weak for those styles with weak geometry. Furthermore, to alleviate the GPU-memory overhead when stylizing the 3D radiance field, we proposed the dilated sample to update the entire network with a small number of rays. In fact, like dilated convolution, our results sometimes will produce a slightly blurry result in some repeating textures, which is a problem to be solved in further work.

5. Conclusion

This paper proposed DeSRF, a deformable stylized radiance field, to achieve high-fidelity and efficient style transfer from a given 2D style image to arbitrary 3D scenes. The major technical contributions of the proposed method are twofold. First, we introduced a deformable module to the radiance field network to capture and learn the implicit geometry style information from the style image. Second, we designed a new training strategy named Dilated Sampling to propagate the style loss to the radiance field more efficiently compared to existing methods. Both qualitative and quantitative experiments were conducted to demonstrate the effectiveness and efficiency of the proposed method, which is capable of transferring both geometry and texture styles from input 2D style images to 3D scenes.

Acknowledgements

This work was supported by National Language Committee of China (Grant No.: ZDI135-130), Center For Chinese Font Design and Research, and Key Laboratory of Science, Technology and Standard in Press Industry (Key Laboratory of Intelligent Press Media Technology).

References

[1] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 28(3):24, 2009. 3

[2] Xu Cao, Weimin Wang, Katashi Nagao, and Ryosuke Nakamura. Psnet: A style transfer network for point cloud stylization on geometry and color. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3337–3345, 2020. 3

[3] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision (ECCV)*, 2022. 2, 5

[4] Yaosen Chen, Qi Yuan, Zhiqiang Li, Yuegen Liu Wei Wang Chaoping Xie, Xuming Wen, and Qien Yu. Upst-nerf: Universal photorealistic style transfer of neural radiance fields for 3d scene. *arXiv preprint arXiv:2208.07059*, 2022. 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

[6] Tai-Yin Chiu and Danna Gurari. Iterative feature transformation for fast and versatile universal style transfer. In *European Conference on Computer Vision*, pages 169–184. Springer, 2020. 2

[7] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017. 2

[8] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, 2022. 2

[9] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5712–5721, 2021. 2

[10] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015. 1, 2, 5

[11] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016. 2

[12] Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *European Conference on Computer Vision*, pages 484–499. Springer, 2016. 2

[13] Tong He, John Collomosse, Hailin Jin, and Stefano Soatto. Deepvoxels++: Enhancing the fidelity of novel view synthesis from 3d voxel embeddings. In *Proceedings of the Asian Conference on Computer Vision*, 2020. 2

[14] 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. 3

[15] 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*, pages 13869–13878, 2021. 3

[16] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE international conference on computer vision*, pages 1501–1510, 2017. 2

[17] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016. 2

[18] Sunnie SY Kim, Nicholas Kolkin, Jason Salavon, and Gregory Shakhnarovich. Deformable style transfer. In *European Conference on Computer Vision*, pages 246–261. Springer, 2020. 1

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 7

[20] Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. Decomposing nerf for editing via feature field distillation. *arXiv preprint arXiv:2205.15585*, 2022. 2

[21] Nicholas Kolkin, Michal Kucera, Sylvain Paris, Daniel Sykora, Eli Shechtman, and Greg Shakhnarovich. Neural neighbor style transfer. *arXiv e-prints*, pages arXiv–2203, 2022. 3, 5, 6

[22] Georgios Kopanas, Julien Philip, Thomas Leimkühler, and George Drettakis. Point-based neural rendering with per-view optimization. In *Computer Graphics Forum*, volume 40, pages 29–43. Wiley Online Library, 2021. 3

[23] Verica Lazova, Vladimir Guzov, Kyle Olszewski, Sergey Tulyakov, and Gerard Pons-Moll. Control-nerf: Editable feature volumes for scene rendering and manipulation. *arXiv preprint arXiv:2204.10850*, 2022. 2

[24] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021. 2

[25] Jing Liao, Yuan Yao, Lu Yuan, Gang Hua, and Sing Bing Kang. Visual attribute transfer through deep image analogy. *ACM Trans. Graph.*, 36(4):120:1–120:15, July 2017. 3

[26] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020. 2

[27] Steven Liu, Xiuming Zhang, Zhoutong Zhang, Richard Zhang, Jun-Yan Zhu, and Bryan Russell. Editing conditional radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5773–5783, October 2021. 2

[28] Xiao-Chang Liu, Yong-Liang Yang, and Peter Hall. Learning to warp for style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3702–3711, 2021. 1

[29] 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*, pages 7210–7219, 2021. 2

[30] 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)*, 2019. 5

[31] 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 *ECCV*, 2020. 1, 2, 4, 5

[32] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022. 2

[33] Thu Nguyen-Phuoc, Feng Liu, and Lei Xiao. Snerf: stylized neural implicit representations for 3d scenes. *ACM Transactions on Graphics (TOG)*, 41(4):1–11, 2022. 2, 3, 5, 7

[34] Dae Young Park and Kwang Hee Lee. Arbitrary style transfer with style-attentional networks. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5880–5888, 2019. 2

[35] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 4

[36] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 2, 4

[37] Albert Pumarola, Stefan Popov, Francesc Moreno-Noguer, and Vittorio Ferrari. C-flow: Conditional generative flow models for images and 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7949–7958, 2020. 2

[38] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

[39] 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

[40] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *Proceedings of the IEEE international conference on computer vision*, pages 2088–2096, 2017. 2

[41] Ondrej Texler, Jakub Fišer, Michal Lukáč, Jingwan Lu, Eli Shechtman, and Daniel Sỳkora. Enhancing neural style transfer using patch-based synthesis. In *Proceedings of the 8th ACM/Eurographics Expressive Symposium on Computational Aesthetics and Sketch Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering*, pages 43–50, 2019. 3

[42] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions On Graphics (TOG)*, 36(4):1–11, 2017. 2

[43] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9421–9431, 2021. 2

[44] Xinchen Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. *Advances in neural information processing systems*, 29, 2016. 2

[45] Kangxue Yin, Jun Gao, Maria Shugrina, Sameh Khamis, and Sanja Fidler. 3dstylenet: Creating 3d shapes with geometric and texture style variations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12456–12465, 2021. 3

[46] Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao. Nerf-editing: geometry editing of neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18353–18364, 2022. 2, 4

[47] Kai Zhang, Nick Kolkin, Sai Bi, Fujun Luan, Zexiang Xu, Eli Shechtman, and Noah Snavely. Arf: Artistic radiance fields. In *European Conference on Computer Vision*, pages 717–733. Springer, 2022. 2, 3, 5, 6, 7

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