StyleRF: Zero-shot 3D Style Transfer of Neural Radiance Fields

Kunhao Liu¹ Fangneng Zhan² Yiwen Chen¹ Jiahui Zhang¹ Yingchen Yu¹ Abdulmotaleb El Saddik³,⁵ Shijian Lu¹∗ Eric Xing⁴,⁵

¹Nanyang Technological University ²Max Planck Institute for Informatics ³University of Ottawa ⁴Carnegie Mellon University ⁵MBZUAI

Figure 1. Zero-shot 3D Style Transfer. Given a set of Multi-view Content Images of a 3D scene, StyleRF can transfer arbitrary Reference Styles to the 3D scene in a zero-shot manner, rendering high-quality Stylized Novel Views with superb multi-view consistency.

Abstract

3D style transfer aims to render stylized novel views of a 3D scene with multi-view consistency. However, most existing work suffers from a three-way dilemma over accurate geometry reconstruction, high-quality stylization, and being generalizable to arbitrary new styles. We propose StyleRF (Style Radiance Fields), an innovative 3D style transfer technique that resolves the three-way dilemma by performing style transformation within the feature space of a radiance field. StyleRF employs an explicit grid of high-level features to represent 3D scenes, with which high-fidelity geometry can be reliably restored via volume rendering. In addition, it transforms the grid features according to the reference style which directly leads to high-quality zero-shot style transfer. StyleRF consists of two innovative designs. The first is sampling-invariant content transformation that makes the transformation invariant to the holistic statistics of the sampled 3D points and accordingly ensures multi-view consistency. The second is deferred style transformation of 2D feature maps which is equivalent to the transformation of 3D points but greatly reduces memory footprint without degrading multi-view consistency. Extensive experiments show that StyleRF achieves superior 3D stylization quality with precise geometry reconstruction and it can generalize to various new styles in a zero-shot manner. Project website: https://kunhao-liu.github.io/StyleRF/

1. Introduction

Given a set of multi-view images of a 3D scene and an image capturing a target style, 3D style transfer aims to generate novel views of the 3D scene that have the target style consistently across the generated views (Fig. 1). Neural style transfer has been investigated extensively, and state-of-the-art methods allow transferring arbitrary styles in a
zero-shot manner. However, most existing work focuses on style transfer across 2D images [15, 21, 24] but cannot extend to a 3D scene that has arbitrary new views. Prior studies [19, 22, 37, 39] have shown that naively combining 3D novel view synthesis and 2D style transfer often leads to multi-view inconsistency or poor stylization quality, and 3D style transfer should optimize novel view synthesis and style transfer jointly.

However, the current 3D style transfer is facing a three-way dilemma over accurate geometry reconstruction, high-quality stylization, and being generalizable to new styles. Different approaches have been investigated to resolve the three-way dilemma. For example, multiple style transfer [11, 22] requires a set of pre-defined styles but cannot generalize to unseen new styles. Point-cloud-based style transfer [19, 37] requires a pre-trained depth estimation module that is prone to inaccurate geometry reconstruction. Zero-shot style transfer with neural radiance fields (NeRF) [8] cannot capture detailed style patterns and textures as it implicitly injects the style information into neural network parameters. Optimization-based style transfer [17, 39, 63] suffers from slow optimization and cannot scale with new styles.

In this work, we introduce StyleRF to resolve the three-way dilemma by performing style transformation in the feature space of a radiance field. A radiance field is a continuous volume that can restore more precise geometry than point clouds or meshes. In addition, transforming a radiance field in the feature space is more expressive with better stylization quality than implicit methods [8], and it can also generalize to arbitrary styles. We construct a 3D scene representation with a grid of deep features to enable feature transformation. In addition, multi-view consistent style transformation in the feature space could be achieved by either transforming the whole feature grid or transforming the sampled 3D points. We adopt the latter as the former incurs much more computational cost during training to stylize the whole feature grid in every iteration, whereas the latter can reduce computational cost through decreasing the size of training patch and the number of sampled points. However, applying off-the-shelf style transformations to a batch of sampled 3D points impairs the multi-view consistency as they are conditioned on the holistic statistics of the batch. Beyond that, transforming every sampled 3D point is memory-intensive since NeRF needs to query hundreds of sampled points along each ray for rendering a single pixel.

We decompose the style transformation into sampling-invariant content transformation (SICT) and deferred style transformation (DST), the former eliminating the dependency on holistic statistics of sampled point batch and the latter deferring style transformation to 2D feature maps for better efficiency. In SICT, we introduce volume-adaptive normalization that learns the mean and variance of the whole volume instead of computing them from a sampled batch. In addition, we apply channel-wise self-attention to transform each 3D point independently to make it conditioned on the feature of that point regardless of the holistic statistics of the sampled batch. In DST, we defer the style transformation to the volume-rendered 2D feature maps based on the observation that the style transformation of each point is the same. By formulating the style transformation by pure matrix multiplication and adaptive bias addition, transforming 2D feature maps is mathematically equivalent to transforming 3D point features but it saves computation and memory greatly. Thanks to the memory-efficient representation of 3D scenes and deferred style transformation, our network can train with 256 × 256 patches directly without requiring sub-sampling like previous NeRF-based 3D style transfer methods [8, 11, 22].

The contributions of this work can be summarized in three aspects. First, we introduce StyleRF, an innovative zero-shot 3D style transfer framework that can generate zero-shot high-quality 3D stylization via style transformation within the feature space of a radiance field. Second, we design sampling-invariant content transformation and deferred style transformation, the former achieving multi-view consistent transformation by eliminating dependency on holistic statistics of sampled point batch while the latter greatly improves stylization efficiency by deferring style transformation to 2D feature maps. Third, extensive experiments show that StyleRF achieves superior 3D style transfer with accurate geometry reconstruction, high-quality stylization, and great generalization to new styles.

2. Related Work

Neural scene representations. 3D scene representation has been extensively studied in recent years with different ways of representations such as volumes [23, 27, 46, 49, 59], point clouds [1, 45], meshes [25, 55], depth maps [20, 30], and implicit functions [7, 34, 41, 61]. These methods adopt differentiable rendering which enables model optimization by using 2D multi-view images. Among them, Neural Radiance Field (NeRF) [36] can render a complex 3D scene with high fidelity and accurate geometry. It represents scenes with an implicit coordinate function that maps each 3D coordinate to a density value and a color value, and employs volume rendering to generate images of novel views. However, the implicit coordinate function is represented by a large multilayer perceptron (MLP) that is often hard to optimize and slow to infer. Serval studies adopt a hybrid representation [3, 4, 10, 14, 31, 33, 38, 44, 52, 62] to speed up the reconstruction and rendering. They employ explicit data structures such as discrete voxel grids [14, 52], decomposed tensors [3, 4, 13], hash maps [38], etc. to store features or spherical harmonics, enabling fast convergence and inference.
Although most existing work extracts features as middle-level representations of scenes, the extracted features are usually an intermediate output of neural networks which have little semantic meanings and are not suitable for the style transfer task. We introduce decomposed tensors [4] to store high-level features extracted by pre-trained CNNs, which enables transformations in feature space as well as efficient training and inference. Though [3, 16, 40] also render feature maps instead of RGB maps, they are computationally intensive and usually work with low-resolution feature maps. StyleRF can instead render full-resolution feature maps (the same as the output RGB images) efficiently and it uses high-level features largely for transformation only.

**Neural style transfer.** Neural style transfer aims at rendering a new image that contains the content structure of one image and the style patterns of another. The seminal work in [15] shows that multi-level feature statistics extracted from intermediate layers of pre-trained CNNs could be used as a representation of the style of an artistic image, but it treats style transfer as a slow and iterative optimization task. [9, 21, 24, 28, 29, 32, 43, 50, 58] utilize feed-forward networks to approximate the optimization procedure to speed up rendering. Among them, [9, 21, 28, 29, 32, 43, 50, 58] can achieve zero-shot style transfer by applying transformations to the high-level features extracted by pre-trained CNNs, where the feature transformations can be achieved by matching second-order statistics [21, 29], linear transformation [28, 58], self-attention transformation [9, 32, 43], etc. Video style transfer extends style transfer to videos for injecting target styles consistently across adjacent video frames. Several studies leverage optical flow [4, 18, 47, 56, 57] as temporal constraints to estimate the movement of video contents. They can produce smooth videos, but have little knowledge of the underlying 3D geometry and cannot render consistent frames in arbitrary views [19, 37].

Huang et al. first tackle stylizing complex 3D scenes [19]. They construct a 3D scene by back-projecting image features into the 3D space to form a point cloud and then perform style transformation on the features of 3D points. Their method can achieve zero-shot style transfer, but requires an error-prone pre-trained depth estimator to model scene geometry. [37] also constructs a point cloud for stylization but it mainly focuses on monocular images. Instead, [6, 8, 11, 22, 39, 63] use NeRF [36] as the 3D representation which can reconstruct scene geometry more faithfully. [6] is a photorealistic style transfer method that can only transfer the color tone of style images. [39, 63] achieve 3D style transfer via optimization and can produce visually high-quality stylization, but they require a time-consuming optimization procedure for every reference style. [11, 22] employ latent codes to represent a set of pre-defined styles, but cannot generalize to unseen styles. [8] can achieve arbitrary style transfer by implicitly instilling the style information into MLP parameters. However, it can only transfer the color tone of style images but cannot capture detailed style patterns. StyleRF can transfer arbitrary style in a zero-shot manner, and it can capture style details such as strokes and textures as well.

### 3. Method

The overview of StyleRF is shown in Fig. 2. For a batch of sampled points along a ray $r$, the corresponding features $F_i, i \in [1, 2, \ldots, N]$ are first extracted from the feature grid described in Sec. 3.1, each of which is transformed to $F_i$ independently via Sampling-Invariant Content Transformation, regardless of the holistic statistics of the point batch. $F_i$ is then transformed to a feature map $F_{cs}$ via Volume Rendering. After that, the Deferred Style Transformation transforms $F_i$ to the feature map $F_{cs}$ adaptively using the sum weight of the sampled points $w_r$ along the ray $r$ and the style information $T, \mu(F_s), \sigma(F_s)$. Finally, a stylized novel view is generated via a CNN decoder.
3.1. Feature Grid 3D Representation

To model a 3D scene with deep features, we use a continuous volumetric field of density and radiance. Different from the original NeRF [36], for every queried 3D position $x \in \mathbb{R}^3$, we get a volume density $\sigma(x)$ and a multi-channel feature $F(x) \in \mathbb{R}^C$ instead of an RGB color, where $C$ is the number of the feature channels. Then we can get the feature of any rays $r$ passing through the volume by integrating sampled points along the ray via approximated volume rendering [36]:

$$F(r) = \sum_{i=1}^{N} w_i F_i, \quad (1)$$

where $w_i = \exp \left( -\sum_{j=1}^{i-1} \sigma_j \delta_j (1 - \exp (-\sigma_i \delta_i)) \right), \quad (2)$ where $\sigma_i, F_i$ denotes the volume density and feature of sampled point $i$, $w_i$ denotes the weight of $F_i$ in the ray $r$, and $\delta_i$ is the distance between adjacent samples. We disable view-dependency effect for better multi-view consistency.

Then we can generate feature maps capturing high-level features and map them to a RGB space using a 2D CNN decoder. However, unlike [3,16,40], we render full-resolution feature maps which have the same resolution as the final RGB images rather than down-sampled feature maps. Rendering full-resolution feature maps has two unique features: 1) it discards up-sampling operations which cause multi-view inconsistency in general [16], 2) it removes aliasing when rendering low-resolution feature maps [2] which causes severe flickering effects in stylized RGB videos.

Directly using 3D voxel grid to store features is memory-intensive. We thus adopt vector-matrix tensor decomposition [4] that relaxes the low-rank constraints for two modes of a 3D tensor and factorizes tensors into compact vector and matrix factors, which lowers the space complexity from $O(n^3)$ to $O(n^2)$, massively reducing the memory footprint. We employ a density grid to store volume density and a feature grid to store multi-channel features respectively.

3.2. Feature Transformation for Style Transfer

Once we have the feature grid representation of a scene, we can tackle the task of stylizing 3D scenes. Given a reference style image, our goal is to render stylized novel views of the 3D scene with multi-view consistency. To achieve this, we apply transformations to the features of the grid.

One plausible solution to this task is to apply style transfer to the feature grid directly. This solution is efficient in evaluations as it can render any stylized views with a single style transfer process only. However, it is impractical to train such transformation as it needs to stylize the whole feature grid in every iteration. Another solution is to apply an off-the-shelf zero-shot style transfer method to the features of the sampled 3D points. While this solution can reduce computational cost through decreasing the size of training patch and the number of sampled points, it has two problems: 1) vanilla zero-shot style transformation is conditioned on holistic statistics of the sampled point batch [21,28,32], which violates multi-view consistency in volume rendering as the feature transformation of a specific 3D point will vary across different sampled points; 2) volume rendering requires sampling hundreds of points along a single ray, which makes transformation on the point batch memory-intensive.

Motivated by the observation that style transformation is conditioned on both content information and style information, we decompose the style transformation into sampling-invariant content transformation (SICT) and deferred style transformation (DST). After the decomposition, SICT will be conditioned solely on the content information while DST conditioned solely on the style information, more details to be elaborated in the ensuing subsections.

3.2.1 Sampling-invariant Content Transformation

Given a batch of sampled points, we can get their corresponding features $F_i \in \mathbb{R}^C$, $i \in [1, 2, ..., N]$ from the feature grid, where $N$ is the number of the sampled points along a ray and $C$ is the number of the feature channels. The goal of SICT is to transform the extracted features $F_i$ so that they can be better stylized. We formulate SICT as a channel-wise self-attention operation to the features after instance normalization (IN) [54]. Specifically, we formulate $Q$(query), $K$(key), and $V$(value) as:

$$Q = q(Norm(F_i)), \quad (3)$$
$$K = k(Norm(F_i)), \quad (4)$$
$$V = v(Norm(F_i)), \quad (5)$$

where $q, k, v$ are $1 \times 1$ convolution layers which reduce the channel number from $C$ to $C'$ for computational efficiency, and $Norm$ denotes the IN. However, as shown in Fig. 3,
vanilla IN calculates per-dimension mean and standard-deviation of the batch of sampled points, which varies with different sampled points and incurs multi-view inconsistency accordingly. Thus we design volume-adaptive IN which, during training, keeps running estimates of the computed mean and standard-deviation, and uses them for normalization during evaluations (instead of computing from the sampled point batch). Through volume-adaptive IN, we can ensure that the content transformation is consistent regardless of the sampled point batch’s holistic statistics.

Channel-wise self-attention can thus be implemented by:

\[
\hat{F}_i = V \otimes \text{Softmax}(\tilde{\text{cov}}(Q, K)),
\]

where \(\otimes\) denotes matrix multiplication and \(\tilde{\text{cov}}(Q, K) \in \mathbb{R}^{N \times C' \times C'}\) denotes the covariance matrix in the channel dimension.

### 3.2.2 Deferred Style Transformation

After applying SICT to the features of each 3D point, we apply DST to the volume-rendered 2D feature maps \(\hat{F}_c\) rather than 3D point features \(\tilde{F}_i\). To ensure multi-view consistency, we formulate the transformation as matrix multiplication and adaptive bias addition as illustrated in Fig. 4.

Specifically, we first extract feature maps \(F_s\) of the reference style \(\bar{S}\) using a pre-trained VGG [51], and then generate the style transformation matrix \(T \in \mathbb{R}^{C' \times C'}\) using feature covariance \(\text{cov}(F_s)\) following [28]. Next, we apply matrix multiplication with \(T\) to the feature maps \(\hat{F}_c\), and use a \(1 \times 1\) convolution layer \(\text{conv}\) without bias to restore the channel number from \(C'\) to \(C\). Though these operations can partially instill style information, they are not expressive enough without bias addition containing style information [58]. Thus following [21], we multiply the feature maps with the standard-deviation value \(\sigma(F_s)\) and add the mean value \(\mu(F_s)\). To ensure it is equivalent when applying the transformation to either 3D point features or 2D feature maps, we adaptively modulate the mean value \(\mu(F_s)\) with the sum weight of sampled points along each ray \(w_r\). DST can be mathematically formulated by:

\[
F_{cs} = \text{conv}(T \otimes \hat{F}_c) \times \sigma(F_s) + w_r \times \mu(F_s), \quad (7)
\]

where \(\hat{F}_c = \sum_{i=1}^{N} w_i \hat{F}_i, w_r = \sum_{i=1}^{N} w_i, r \in \mathcal{R}\) (8)

where \(w_i\) denotes the weight of sampled point \(i\) (Eq. (2)), \(\hat{F}_i\) denotes the feature of sample \(i\) after SICT, and \(\mathcal{R}\) is the set of rays in each training batch.

Note \(\text{conv}\) is a \(1 \times 1\) convolution layer without bias, so it is basically a matrix multiplication operation. And \(\sigma(S), \mu(S)\) are scalars. Together with the adaptive bias modulation \(w_r\), Eq. (7) can be reformulated by:

\[
F_{cs} = \sum_{i=1}^{N} w_i \left( \text{conv}(T \otimes \hat{F}_i) \times \sigma(F_s) + \mu(F_s) \right), \quad (9)
\]

where part (i) can be seen as applying style transformation on every 3D point feature independently before volume rendering. This proves that applying DST on 2D feature maps is equivalent to applying the transformation on 3D points’ features, maintaining multi-view consistency. The full derivation of Eq. (9) is provided in the appendix.

Finally, we adopt a 2D CNN decoder to project the stylized feature maps \(F_{cs}\) to RGB space to generate the final stylized novel view images.

### 3.3. Two-stage Model Training

The training of our model is divided into the feature grid training stage and the stylization training stage, the former is trained with the target of novel view synthesis, and the latter is trained with the target of style transfer.

**Feature grid training stage (First stage).** We first learn the feature grid 3D representation for the novel view synthesis task, in preparation for performing feature transformation for style transfer. We train the feature grid and the 2D CNN decoder simultaneously, with the supervision of both RGB images and their bilinearly up-sampled feature maps extracted from ReLU3_1 layer of pre-trained VGG [51]. By aligning the VGG features with the feature grid, the reconstructed features acquire semantic information. We use density grid pre-trained solely on RGB images since the supervising feature maps are not strictly multi-view consistent. The training objective is the mean square error (MSE) between the predicted and ground truth feature maps and RGB images. Following [19, 37], we use perceptual
loss [24] as additional supervision to increase reconstructed image quality. The overall loss function is:

\[
\mathcal{L}_{\text{grid}} = \sum_{\mathbf{r} \in \mathcal{R}} \left\| \hat{\mathbf{F}}(\mathbf{r}) - \mathbf{F}(\mathbf{r}) \right\|_2^2 + \left\| \hat{\mathbf{I}}_\mathcal{R} - \mathbf{I}_\mathcal{R} \right\|_2^2 + \sum_{l \in \mathcal{I}_p} \left\| \mathcal{F}^l(\hat{\mathbf{I}}_\mathcal{R}) - \mathcal{F}^l(\mathbf{I}_\mathcal{R}) \right\|_2^2 \tag{10}
\]

where \( \mathcal{R} \) is the set of rays in each training batch, \( \hat{\mathbf{F}}(\mathbf{r}), \mathbf{F}(\mathbf{r}) \) are the predicted and ground truth feature of ray \( \mathbf{r} \), \( \hat{\mathbf{I}}_\mathcal{R}, \mathbf{I}_\mathcal{R} \) are the predicted and ground truth RGB image, \( l_p \) denotes the set of VGG layers that compute perceptual loss, \( \mathcal{F}^l \) denotes the feature maps of the \( l \)th layer of pre-trained VGG network.

**Stylization training stage (Second stage).** Our model learns to stylize novel views in the second stage. We freeze the feature grid, train the style transfer module, and fine-tune the CNN decoder. Thanks to the memory-efficient representation of 3D scenes and DST, unlike [8, 11, 48], our model can be trained directly on 256 × 256 patches, making patch sub-sampling algorithm [8, 11, 22, 48] unnecessary. We use the same loss as [21] where the content loss \( \mathcal{L}_c \) is the MSE of the feature maps and the style loss \( \mathcal{L}_s \) is the MSE of the channel-wise feature mean and standard-deviation:

\[
\mathcal{L}_{\text{stylization}} = \mathcal{L}_c + \lambda \mathcal{L}_s, \tag{11}
\]

where \( \lambda \) balances the content preservation and the stylization effect.

4. Experiments

We evaluate StyleRF extensively with qualitative experiments in Sec. 4.1, quantitative experiments in Sec. 4.2 and ablation studies in Sec. 4.3. We demonstrate two applications of StyleRF in Sec. 4.4. The implementation details are provided in the appendix.

4.1. Qualitative Experiments

We evaluate StyleRF over two public datasets including LLFF [35] that contains real scenes with complex geometry structures and Synthetic NeRF [36] that contains 360° views of objects. In addition, we benchmark StyleRF with two state-of-the-art zero-shot 3D style transfer methods LSNV [19] and Hyper [8] with their released codes. We perform comparisons on LLFF dataset [35].

Fig. 5 shows qualitative comparisons. We can see that StyleRF achieves clearly better stylization with more precise geometry reconstruction. Specifically, StyleRF can generate high-definition stylization with realistic textures and patterns of style images. The superior stylization is largely attributed to our transformation design that allows working in the feature space with full-resolution feature maps. As illustrated in the highlight boxes, StyleRF can successfully restore the intricate geometry of complex scenes thanks to its radiance field representations. In addition, only StyleRF faithfully transfers the squareness texture in the second style image. Furthermore, StyleRF can robustly generalize to new styles in a zero-shot manner and can adapt well to 360° dataset as illustrated in Fig. 1. As a comparison, LSNV [19] fails to capture fine-level geometry like the bones of the T-Rex and the petals of the flower while Hyper [8] produces very blurry stylization.

4.2. Quantitative Results

3D style transfer is a very new and under-explored task and there are few metrics for quantitative evaluation of stylization quality. Hence, we manage to evaluate the multi-view consistency only. In our experiments, we warp one view to the other according to the optical flow [53] using softmax splatting [42], and then computed the masked RMSE score and LPIPS score [64] to measure the stylization consistency. Following [8, 11, 19], we compute the short-range and long-range consistency scores which compare adjacent views and far-away views respectively. We compare StyleRF against two state-of-the-art zero-shot 3D style transfer methods Hyper [8] and LSNV [19], one SOTA single-frame-based video style transfer method CCPL [60], one SOTA multi-frames-based video style transfer method ReReVST [57], and one classical image style transfer method AdaIN [21].

It can be seen from Tab. 1 that StyleRF significantly outperforms image style transfer approach [21] and video style transfer approach [57, 60] which capture little information about the underlying 3D geometry. In addition, StyleRF achieves better consistency than point-cloud-based 3D style transfer [19] as well. Note Hyper [8] achieves slightly better LPIPS and RMSE scores than our method, largely because it produces over-smooth results and inadequate stylization as shown in Fig. 5.

4.3. Ablation Studies

We design two innovative techniques to improve the stylization quality and maintain the multi-view consistency.

<table>
<thead>
<tr>
<th>Method</th>
<th>Short-range Consistency</th>
<th>Long-range Consistency</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdaIN [21]</td>
<td>0.152</td>
<td>0.123</td>
</tr>
<tr>
<td>CCPL [60]</td>
<td>0.110</td>
<td>0.106</td>
</tr>
<tr>
<td>ReReVST [57]</td>
<td>0.098</td>
<td>0.080</td>
</tr>
<tr>
<td>LSNV [19]</td>
<td>0.093</td>
<td>0.092</td>
</tr>
<tr>
<td>Hyper [8]</td>
<td>0.084</td>
<td>0.068</td>
</tr>
<tr>
<td>Ours</td>
<td>0.072</td>
<td>0.082</td>
</tr>
</tbody>
</table>

Table 1. Results on consistency. We compare StyleRF with the state-of-the-art on consistency using LPIPS (\( \downarrow \)) and RMSE (\( \downarrow \)).
The first is volume-adaptive instance normalization which uses the learned mean and variance of the whole volume during inference, eliminating the dependency on holistic statistics of the sampled point batch. The second is the adaptive bias addition in DST, which improves the stylization quality using bias capturing style information. We evaluate the two designs to examine how they contribute to the overall stylization of our method.

**Volume-adaptive instance normalization.** We compare our volume-adaptive instance normalization (IN) with vanilla IN and StyleRF without IN. As Fig. 6 (c) shows, vanilla IN produces severe block-shape artifacts as the transformation of each batch is conditioned on the holistic statistics of itself, thus each batch (i.e. block in the image) produces inconsistent stylization which leads to the artifacts. However, if we discard IN as shown in Fig. 6 (d), the multi-view consistency can maintain but the stylization quality compromises a lot, failing to capture the correct color tone of the reference style image. This is because IN removes the original style information of the content image which facilitates the transfer of the reference style [21].

**Adaptive bias addition.** As illustrated in Fig. 6 (b), the stylization quality degrades a lot if we eliminate the adaptive bias addition in DST (Sec. 3.2.2), producing unnatural
Figure 6. Ablation studies. (a) shows the stylization of our full pipeline. (b) shows the stylization without the adaptive bias. (c) shows the stylization when replacing the volume-adaptive instance normalization (IN) with vanilla IN. (d) shows the stylization without any IN.

Figure 7. Multi-style interpolation. StyleRF can smoothly interpolate between arbitrary styles by interpolating features of the scene.

stylization compared to the stylization of our full pipeline in Fig. 6 (a). This is because bias usually contains crucial style information such as the overall color tone [58]. StyleRF employs bias addition that is adaptively modulated by the weight of each ray, improving the stylization quality and keeping multi-view consistency concurrently.

4.4. Applications

StyleRF can be easily extended along different directions with different applications. We provide two possible extensions in the ensuing subsections.

Multi-style interpolation. StyleRF can smoothly interpolate different styles thanks to its high-level feature representation of a 3D scene. As illustrated in Fig. 7, we linearly interpolate the feature maps of a specific view by using four different styles at four corners. Unlike previous NeRF-based 3D style transfer that supports style interpolation by interpolating one-hot latent vectors [11], StyleRF can interpolate arbitrary numbers of unseen new styles by interpolating features of the scene, yielding more smooth and harmonious interpolation. Hence, StyleRF can not only transfer arbitrary styles in a zero-shot manner but also generate non-existent stylization via multi-style interpolation.

Compositional 3D style transfer. Thanks to its precise geometry reconstruction, StyleRF can be seamlessly integrated with NeRF-based object segmentation [12, 26, 65] for compositional 3D style transfer. As shown in Fig. 8, we apply 3D-consistent segmentation masks to the feature maps and apply different styles to stylize the contents inside and outside the masks separately. We can see that the edges of the masks can be blended more softly by applying the segmentation masks to the feature maps instead of RGB images. Due to its zero-shot nature, StyleRF can create infinite combinations of styles without additional training, producing numerous artistic creations and inspirations.

5. Conclusion

In this paper, we present StyleRF, a novel zero-shot 3D style transfer method that balances the three-way dilemma over accurate geometry reconstruction, high-quality stylization, and being generalizable to arbitrary new styles. By representing the 3D scene with an explicit grid of high-level features, we can faithfully restore high-fidelity geometry through volume rendering. Then we perform style transfer on the feature space of the scene, leading to high-quality zero-shot stylization results. We innovatively design sampling-invariant content transformation to maintain multiview consistency and deferred style transformation to increase efficiency. We demonstrate that StyleRF achieves superior 3D stylization quality than previous zero-shot 3D style transfer methods, and can be extended to various interesting applications for artistic 3D creations.

Acknowledgement

This project is funded by the Ministry of Education Singapore, under the Tier-1 project scheme with project number RT18/22.
References


[65] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J. Davison. In-place scene labelling and understanding with implicit scene representation. In ICCV, 2021. 8