

---

# FPGS: Feed-Forward Semantic-aware Photorealistic Style Transfer of Large-Scale Gaussian Splatting

GeonU Kim · Kim Youwang · Lee Hyoseok · Tae-Hyun Oh

**Abstract** We present FPGS, a feed-forward photorealistic style transfer method of large-scale radiance fields represented by Gaussian Splatting. FPGS stylizes large-scale 3D scenes with arbitrary, multiple style reference images without additional optimization while preserving multi-view consistency and real-time rendering speed of 3D Gaussians. Prior arts required tedious per-style optimization or time-consuming per-scene training stage and were limited to small-scale 3D scenes. FPGS efficiently stylizes large-scale 3D scenes by introducing a style-decomposed 3D feature field, which inherits AdaIN’s feed-forward stylization machinery, supporting arbitrary style reference images. Furthermore, FPGS supports multi-reference stylization with the semantic correspondence matching and local AdaIN, which adds diverse user control for 3D scene styles. FPGS also preserves multi-view consistency by applying semantic matching and style transfer processes directly onto queried features in 3D space. In experiments, we demonstrate that FPGS achieves favorable photorealistic quality scene stylization for large-scale static and dynamic

3D scenes with diverse reference images. Project page: <https://kim-geonu.github.io/FPGS/>

**Keywords** Photorealistic Style Transfer · 3D Computer Vision · Computer Graphics · Gaussian Splatting · Neural Radiance Fields

## 1 Introduction

Large-scale 3D scene reconstruction is a long-standing problem in computer vision and graphics (Früh and Zakhor, 2004; Snavely et al, 2006; Pollefeys et al, 2008; Agarwal et al, 2009; Zhu et al, 2018; Tancik et al, 2022), which aims to build realistic 3D virtual scenes from measurements, *e.g.*, a set of images. Recently, Neural Radiance Fields (Mildenhall et al, 2020; Barron et al, 2021; Fridovich-Keil et al, 2022; Jun-Seong et al, 2022; Müller et al, 2022; Fridovich-Keil et al, 2023) and 3D Gaussian Splatting (Kerbl et al, 2023) have enabled photorealistic reconstruction of 3D scenes by modeling the scenes as radiance fields. Also, their large-scale extensions (Tancik et al, 2022; Turki et al, 2022; Zhenxing and Xu, 2022; Kerbl et al, 2024; Lin et al, 2024; Liu et al, 2024b) have shown remarkable progress in modeling coherent outdoor 3D scenes, suggesting promising future directions, *e.g.*, VR/AR applications. In this work, we take a step further and address a task, photorealistic style transfer (PST) of large-scale 3D scenes, which we call it 3D scene PST.

The 3D scene PST task aims to transfer the visual styles of style reference images onto a large-scale 3D scene represented by a *radiance field*. With this objective, the resulting stylized output is expected to be photorealistic and preserve the geometric structure of the original scene. The large-scale 3D scene PST has various applications, where it can enrich virtual 3D spaces of XR applications and realistically augment existing

---

GeonU Kim  
School of Artificial Intelligence, POSTECH, Pohang, Republic of Korea  
E-mail: [gukim@postech.ac.kr](mailto:gukim@postech.ac.kr)

Kim Youwang  
Department of Electrical Engineering, POSTECH, Pohang, Republic of Korea  
E-mail: [youwang.kim@postech.ac.kr](mailto:youwang.kim@postech.ac.kr)

Lee Hyoseok  
Department of Electrical Engineering, POSTECH, Pohang, Republic of Korea  
E-mail: [hyos99@postech.ac.kr](mailto:hyos99@postech.ac.kr)

Tae-Hyun Oh  
School of Computing, KAIST, Daejeon, Republic of Korea  
E-mail: [taehyun.oh@kaist.ac.kr](mailto:taehyun.oh@kaist.ac.kr)

Table 1: Our FPGS provides a photorealistic, feed-forward style transfer of 3D Gaussians, while supporting multiple style reference images in the stylization stage. FPGS also achieves real-time rendering, different from competing NeRF-based methods.

	UPST-NeRF	LipRF	StyleGaussian	FPGS
Photorealistic	✓	✓	✗	✓
Feed-forward	✓	✗	✓	✓
Multi-reference	✗	-	✗	✓
Real-time render	✗	✗	✓	✓

real-world autonomous driving datasets (Geiger et al, 2012; Cordts et al, 2016).

Recent studies (Chen et al, 2024; Zhang et al, 2023b; Li and Pan, 2024) have developed PST methods for 3D scenes represented by neural radiance fields and shown plausible visual qualities. However, all of them require additional time-consuming training stages to enable feed-forward style transfer (Chen et al, 2024) or tedious per-style optimization steps to apply even just a single style reference to the scene (Zhang et al, 2023b; Li and Pan, 2024). More importantly, due to the aforementioned drawbacks, the previous 3D scene PST methods do not scale to large 3D scenes. In addition, their neural radiance field representation limits the rendering speed.

This motivates us to develop an efficient PST method for large-scale 3D scenes that 1) can stylize the whole 3D scene *without* additional time-consuming training or per-style optimization, 2) can render in real-time, and 3) can support diverse multiple reference images. In this work, we propose a feed-forward PST method of large-scale 3D scenes represented by 3D Gaussian Splattings (FPGS). To implement a feed-forward 3D PST method, we employ the adaptive instance normalization (AdaIN) (Huang and Belongie, 2017) layer before the rasterization process to dress up the decoded colors with the styles. Specifically, we propose a photorealistic *stylizable radiance field* consisting of a scene *content field* and a scene *semantic field*. Given a large set of photos of the target scene, we first train a scene *content field* to embed the scene geometry and content features that can be later decoded to a large-scale radiance field of arbitrary styles. The scene *semantic field* is trained together to aid to match proper local styles to the local scene. With the obtained scene content field, FPGS stylizes the whole 3D scene via the feed-forward AdaIN, which manipulates the scene content field with the style reference images’ feature statistics in a feed-forward manner without any nuisance optimization. After the feed-forward style transfer, we can render the stylized scene in real-time, which is not achieved by prior meth-

ods (Chen et al, 2024; Zhang et al, 2023b; Li and Pan, 2024) based on neural radiance fields.

In addition to efficiency, stylizing a large-scale scene requires dealing with diverse objects and contents that are not covered with a single reference image. Also, it is challenging to identify a single reference that can effectively encompass the entire semantics of a large-scale scene. Thus, extending the existing single reference-based methods (Chen et al, 2024; Liu et al, 2023; Li and Pan, 2024) is not straightforward due to diverse contents in the large-scale scene. To overcome this challenge, we propose a style dictionary module and style attention for efficiently retrieving the style matches of each local part of the 3D scene from a given set of diverse style references. The proposed style dictionary consists of pairs of a local semantic code and a local style code extracted from the style references. To form a compact style dictionary, we exploit the clustering of styles and semantics that notably reduces the computational complexity of the style retrieval. Using a style dictionary, we find semantic correspondences between the local semantic codes and the style semantic field.

This work is an extension of our conference version, FPRF (Kim et al, 2024), which focuses on stylizing static 3D scenes represented by neural radiance fields. We notably extend its representation, process, computational speed, and applications. We achieve real-time rendering of the stylized scene by two extended features: 1) representing scene with 3D Gaussians instead of neural radiance field and 2) redesigning the stylization transfer process. Specifically, while FPRF requires a stylization process for each new view rendering due to its implicit representation, FPGS stylizes the whole scene at once before rendering, leveraging the explicit nature of 3D Gaussians. This decoupling of stylization and rendering process enables real-time rendering after a feed-forward scene style transfer. We also propose a compact per-*RGB-to-* scene content feature encoder for stylization, called MLP VGGNet, which does not require any training or feature distillation, and an iterative style transfer method which enhances local style transfer quality. We further extend the target domain to 4D scenes including large-scale urban scenes, which has not been explored in previous 3D PST methods (Chen et al, 2024; Zhang et al, 2023b; Li and Pan, 2024; Kim et al, 2024). In this work, we introduce both FPRF and FPGS to clearly contrast the benefits of FPGS.

Our experiments demonstrate that our FPGS obtains superior stylization qualities on both large/small-scale scenes with multi-view consistent and semantically matched style transfer. We also show the exceptional efficiency of FPGS in terms of training and rendering time. Furthermore, we demonstrate versatility of our method

with various applications including 4D scene stylization, multi-reference stylization, and scribble-based stylization, which are not supported by other prior methods. Our main contributions are summarized as follows:

- We propose a stylizable radiance field represented by 3D Gaussians where we can perform photorealistic style transfer in a feed-forward manner and achieve real-time rendering.
- We propose the style dictionary and its style attention for style retrieval, which allows us to deal with multiple style references via semantic correspondence matching.
- To the best of our knowledge, our work is the first multi-reference based 3D/4D PST without any per-style optimization, which is scalable for large-scale scenes.

## 2 Related Work

Our task relates to large-scale scene reconstruction and photorealistic style transfer for large-scale 3D scenes.

**Large-scale 3D scene reconstruction.** Realistic 3D reconstruction of large-scale scenes has been considered as an important task, which could be a stepping stone to achieve a comprehensive 3D scene understanding and immersive virtual reality. Recently, several studies (Tancik et al, 2022; Turki et al, 2022; Zhenxing and Xu, 2022; Kerbl et al, 2024; Lin et al, 2024; Liu et al, 2024b) tackled the task based on advances in Neural Radiance Fields (Mildenhall et al, 2020) and 3D Gaussian Splatting (Kerbl et al, 2023), which show remarkable reconstruction qualities for large-scale 3D scenes. Prior arts mainly focused on decomposing the large-scale scene into smaller parts, *i.e.*, divide-and-conquer. NeRF-based methods (Tancik et al, 2022; Turki et al, 2022; Zhenxing and Xu, 2022) divide large-scale scenes into multiple regions and optimize a neural radiance field for each region. Similarly, Gaussian-based methods (Lin et al, 2024; Kerbl et al, 2024; Liu et al, 2024b) also reconstruct a large-scale scene in a divide-and-conquer manner and achieve real-time rendering. We propose methods for the two representations, FPRF is for NeRF-based methods, FPGS is for Gaussian-based methods.

The large-scale scene comprising multiple radiance fields poses a challenge for style transfer in terms of efficiency. Stylizing a large-scale scene with per-style optimization methods (Zhang et al, 2023b; Li and Pan, 2024) is impractical, which requires extensive time to optimize every radiance field in the large-scale scene whenever new style images are given. We overcome this limitation by transferring style of radiance fields in a feed-forward manner. Our models are trained separately

with an efficient training stage compared to previous feed-forward methods (Chen et al, 2024; Liu et al, 2023), and generalized to any scenes and any style reference images.

**Photorealistic style transfer of 3D scenes.** The 3D scene style transfer task aims to stylize 3D scenes according to the style of the given reference image while preserving multi-view consistency. Recent works (Chiang et al, 2022; Huang et al, 2022; Fan et al, 2022; Nguyen-Phuoc et al, 2022; Chen et al, 2024; Zhang et al, 2022; Liu et al, 2023; Zhang et al, 2023b; Li and Pan, 2024) combine 3D neural radiance fields with style transfer methods. Among them, UPST-NeRF (Chen et al, 2024), Instant-NeRF-Stylization (Li and Pan, 2024) and LipRF (Zhang et al, 2023b) tackled PST on 3D neural radiance fields. UPST-NeRF constructs a stylizable 3D scene with a hypernetwork, which is trained on stylized multi-view images of a single scene. They can stylize a trained scene in a feed-forward manner with arbitrary style. However, the model requires additional time-consuming (>10 hrs.) per-scene optimization after scene reconstruction. Instant-NeRF-Stylization trains the content and style sub-branches and executes AdaIN (Huang and Belongie, 2017) to perform 3D style transfer. LipRF stylizes the reconstructed 3D scene with multi-view stylized images with 2D PST methods, by leveraging Lipschitz network (Virmaux and Scaman, 2018). However, Instant-NeRF-Stylization and LipRF require a time-consuming iterative optimization process for every unseen reference style image. One of the artistic style transfer methods, StyleRF (Liu et al, 2023) leverages a 3D feature field to stylize 3D scenes with an artistic style. They achieve feed-forward style transfer with distilled 3D feature field, however, it also requires time-consuming per-scene training stages which spends much more time than reconstruction (>5hrs.) even for a small scene. Recently, StyleGaussian (Liu et al, 2024a) proposes style transfer of radiance fields represented by 3D Gaussians, which achieves multi-view consistent style transfer and real-time rendering. However, StyleGaussian focuses on artistic style transfer without considering semantic correspondence between the scene and the reference image, and also requires an additional time-consuming training process (>5hrs.).

Our method, FPGS, mitigates aforementioned inefficiency by constructing stylizable 3D radiance field represented by 3D Gaussians with a faster training stage spending about 24 mins. Also, we distinctively focus on photorealistic stylization with the capability of referring to multi-style references, while StyleGaussian focuses only on artistic stylization with a single reference. Table 1 compares the distinctiveness of our method with the prior works.

### 3 Preliminary

#### 3.1 Neural Radiance Fields (NeRF)

A radiance field  $f$  is a continuous 5D function that maps any 3D point  $\mathbf{x} \in \mathbb{R}^3$  and viewing direction  $\mathbf{d} \in \mathbb{S}^2$  to a volume density  $\sigma \in \mathbb{R}^+$  and a color  $\mathbf{c} \in \mathbb{R}^3$ . Neural Radiance Fields (NeRF) (Mildenhall et al, 2020) use MLP to represent radiance fields, and the scene representation can be rendered and optimized via differentiable volume rendering. A pixel’s color  $\hat{\mathbf{C}}(\mathbf{r})$  of given ray  $\mathbf{r}$  is computed by accumulating the color  $\hat{\mathbf{c}}_i$  and density  $\sigma_i$  of  $N_{\mathbf{r}}$  sampled 3D points  $\mathbf{x}_i$  along the ray  $\mathbf{r}$  as:

$$\hat{\mathbf{C}}(\mathbf{r}) = \sum_{i=1}^{N_{\mathbf{r}}} \hat{\mathbf{c}}_i \alpha_i^{\text{NF}} \prod_{j=1}^{i-1} (1 - \alpha_j^{\text{NF}}) \quad (1)$$

$$\alpha_j^{\text{NF}} = 1 - \exp(-\sigma_j \delta_j),$$

where  $\delta_i$  denotes the distance between the sampled point  $\mathbf{x}_i$  and the next sample point,  $\alpha_j^{\text{NF}}$  is the absorption by  $\mathbf{x}_j$ , and  $\prod_{j=1}^{i-1} (1 - \alpha_j^{\text{NF}})$  denotes the transmittance to the point  $\mathbf{x}_i$ . NeRF is optimized by minimizing the rendered pixel values and ground truth pixel values from training images. Recently, grid-based approaches (Chen et al, 2022; Fridovich-Keil et al, 2023; Müller et al, 2022; Jun-Seong et al, 2024) utilize efficient grid backbones to accelerate training and rendering speed. We adopt K-planes (Fridovich-Keil et al, 2023) as the neural radiance field representation of FPRF due to its efficiency.

#### 3.2 3D Gaussian Splatting (3DGS)

Different from NeRF, 3D Gaussian Splatting (3DGS) (Kerbl et al, 2023) represents a radiance field with explicit 3D Gaussian primitives. In detail, 3D Gaussians are parameterized as:  $\mathcal{G} = \{\mathbf{g}_i = (\mathbf{p}_i, \boldsymbol{\Sigma}_i, \sigma_i, \mathbf{K}_i)\}_{i=1, \dots, P}$ , where  $P$  is the number of 3D Gaussians, a mean  $\mathbf{p}_i \in \mathbb{R}^3$  denotes the position of a 3D Gaussian  $\mathbf{g}_i$  and a covariance  $\boldsymbol{\Sigma}_i \in \mathbb{R}^{3 \times 3}$  denotes the scale and shape of  $\mathbf{g}_i$ . An opacity  $\sigma_i \in \mathbb{R}^+$  and spherical harmonics (SH) coefficients  $\mathbf{K}_i \in \mathbb{R}^{3 \times 16}$  specify the geometry and appearance of the scene. A color  $\hat{\mathbf{c}}_i$  of a 3D Gaussian  $\hat{\mathbf{g}}_i$  can be decomposed into a view-independent base color  $\hat{\mathbf{c}}_i^{\text{diff}}$  and a view-dependent color  $\hat{\mathbf{c}}_i^{\text{spec}}$  which are obtained via SH representation as:

$$\hat{\mathbf{c}}_i = \hat{\mathbf{c}}_i^{\text{diff}} + \hat{\mathbf{c}}_i^{\text{spec}}, \quad (2)$$

$$\hat{\mathbf{c}}_i^{\text{diff}} = \mathbf{K}_i^{[:,1:2]} C_{\text{SH}}, \quad \hat{\mathbf{c}}_i^{\text{spec}} = \mathbf{K}_i^{[:,2:17]} \Gamma_{\text{SH}}(\mathbf{d}).$$

A matrix  $\mathbf{K}_i^{[:,n:m]}$  comprises the columns of  $\mathbf{K}_i$  from  $n$ th up to  $m$ th column,  $C_{\text{SH}}$  is the pre-defined view-independent constant, and  $\Gamma_{\text{SH}}$  is the basis function which generates a basis  $\Gamma_{\text{SH}}(\mathbf{d}) \in \mathbb{R}^{15}$  depending on a

view direction  $\mathbf{d}$ . Similar to Eq. (1), a pixel’s color  $\hat{\mathbf{C}}(\mathbf{r})$  of a given ray  $\mathbf{r}$  is computed as:

$$\hat{\mathbf{C}}(\mathbf{r}) = \sum_{\mathbf{g}_i \in \mathcal{G}_{\mathbf{r}}} \hat{\mathbf{c}}_i \alpha_i^{\text{GS}} \prod_{j=1}^{i-1} (1 - \alpha_j^{\text{GS}}), \quad (3)$$

where  $\mathcal{G}_{\mathbf{r}}$  is the ordered set of overlapped 3D Gaussians on a ray  $\mathbf{r}$  and  $\alpha_i^{\text{GS}}$  is obtained by projecting the 3D Gaussian  $\mathbf{g}_i$  to 2D Gaussian and multiplying the density of the 2D Gaussian with  $\sigma_i$ . The 3D Gaussians  $\mathcal{G}$  are optimized by minimizing a photometric loss between the rendered images and training images. 3DGS provides a tile-based splatting solution achieving real-time rendering of the scene, which is notably faster than prior neural volume rendering methods. We leverage this property of 3DGS to achieve real-time rendering after the scene stylization.

### 4 Feed-Forward Photorealistic Style Transfer of Neural Radiance Field - FPRF

In this section, we propose FPRF, a feed-forward Photorealistic Style Transfer (PST) method of neural radiance fields, as a reference method of feed-forward stylizable radiance field, FPRF. Then, we extend to Gaussian-based one, FPGS, in Sec. 5, which is an advanced PST method for Gaussian splatting. We first propose a single-stage training of the stylizable radiance field using AdaIN (Sec. 4.1). We further describe the scene semantic field to stylize large-scale scenes with multiple reference images (Sec. 4.2).

#### 4.1 Large-scale Stylizable Radiance Field

Our goal is to construct a large-scale radiance field that can be stylized with reference images in a feed-forward manner while achieving photorealistic results after stylization. For the feed-forward style transfer, we employ adaptive instance normalization (AdaIN) (Huang and Belongie, 2017), which is an efficient style reference-based style transfer method, where it linearly transforms the content image’s feature statistics with a given style image’s feature as:

$$\text{AdaIN}(\mathbf{F}(c), \mathbf{F}(s)) = \sigma_s \frac{(\mathbf{F}(c) - \boldsymbol{\mu}_c)}{\sigma_c} + \boldsymbol{\mu}_s, \quad (4)$$

where  $\mathbf{F}(c)$  and  $\mathbf{F}(s)$  are semantic features extracted from content and style images by a pre-trained feature encoder, *e.g.*, VGGNet (Simonyan and Zisserman, 2014). The feature statistics  $\boldsymbol{\mu}_*$  and  $\sigma_*$  over spatial axes are the mean and standard deviation of the extracted features, respectively. The AdaIN layer is favorable for



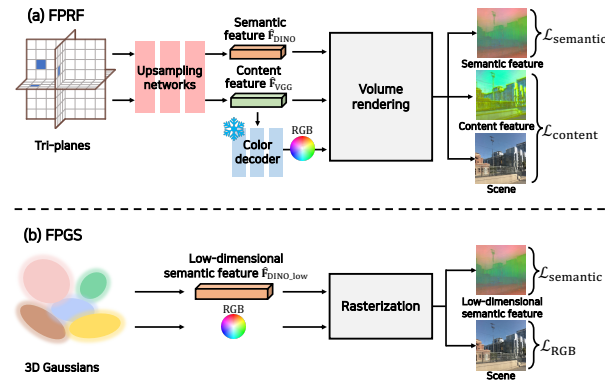


Fig. 1: **Training stage.** Given the scene images, our methods learn the stylizable radiance field. **[Top]** FPRF reconstructs the scene semantic field and the scene content field by distilling high-dimensional semantic and content features extracted from the training images. **[Bottom]** Different from FPRF, FPGS reconstructs the scene content field with RGB loss only and learns the scene semantic field with compressed low-dimensional semantic features.

enabling feed-forward style transfer. This property is particularly useful when dealing with large-scale 3D scenes and enables a fast and seamless stylization. To leverage AdaIN’s statistic-based style transfer to the 3D scene, we propose reconstructing a stylizable 3D neural radiance field by distilling the 2D features onto the field, called the stylizable radiance field.

**Stylizable radiance field.** To build a multi-view consistent stylizable feature field, we apply multi-view bootstrapping (Simon et al, 2017) to our domain. We distill high-dimensional features obtained from 2D input images into neural feature fields that models the 3D scene. To represent a large-scale 3D scene, we extend K-planes (Fridovich-Keil et al, 2023) with the block-composition manner (Tancik et al, 2022), which is used for embedding volumetric scene geometry, radiance, and semantic features. Specifically, we design the stylizable radiance field with two tri-plane grids: scene *content* field and scene *semantic* field. The scene semantic field embeds semantic features of a scene, which will be discussed later in Sec. 4.2. The scene content field is responsible for embedding accurate scene geometry and appearance-related content decoder features. Given a 3D scene point position  $\mathbf{x}=[x, y, z]$  and a ray direction vector  $\mathbf{d}$  as inputs, our scene content field outputs the density, and content feature of the query 3D point (see Fig. 1).

The original NeRF computes a pixel’s RGB color  $\hat{\mathbf{C}}(\mathbf{r})$  with Eq. (1). Correspondingly, we train to render high-dimensional features of 3D points in the scene to a

pixel value  $\hat{\mathbf{F}}(\mathbf{r})$  by accumulating features  $\hat{\mathbf{f}}_i$  along the ray  $\mathbf{r}$ :

$$\hat{\mathbf{F}}(\mathbf{r}) = \sum_{i=1}^{N_r} \hat{\mathbf{f}}_i \alpha_i^{\text{NF}} \prod_{j=1}^{i-1} (1 - \alpha_j^{\text{NF}}), \quad (5)$$

where we substitute the high-dimensional feature  $\hat{\mathbf{f}}$  for the color  $\hat{\mathbf{c}}$  of Eq. (1). We distill 2D image features to a 3D scene by minimizing the error between the volume-rendered features and the 2D features extracted from input images, which we call feature distillation. To distill highly-detailed features into the 3D scenes, we use the refined 2D features by Guided Filtering (He et al, 2012) before distillation. We employ upsampling MLP networks to match the dimension of features from the tri-plane to the dimension of distilled high-dimensional features.

#### Generalizable pre-trained MLP color decoder.

The output of the scene content field requires a separate decoder to decode the stylized feature into the color. One may naïvely train such a decoder scene-by-scene, which suffers from a limited generalization to unseen colors and requires additional training for stylization. In the following, we present a scene-agnostic and pre-trained MLP color decoder  $D_{\text{VGG}}$  compatible with AdaIN. Specifically,  $D_{\text{VGG}}$  transforms the distilled VGG features into colors. To perform style transfer in a feed-forward manner, we pre-train  $D_{\text{VGG}}$  with the diverse set of content images (Lin et al, 2014) and style images (Nichol, 2016), which enable  $D_{\text{VGG}}$  to be generalized to arbitrary style reference images. For that, we employ content loss  $\mathcal{L}_c$  and the style loss  $\mathcal{L}_s$ , similar to (Huang and Belongie, 2017):  $\mathcal{L}_{D_{\text{VGG}}} = \mathcal{L}_c + \lambda_s \mathcal{L}_s$ . The main differences with the training process of AdaIN are threefold. First, we use the MLP architecture instead of the CNN upsampling layer inducing multi-view inconsistency. Also, we employ features from the shallower layer, ReLU2.1 of VGGNet, that contain richer color information. Furthermore, the features from the input content images are upsampled to the pixel resolution for per-pixel decoding. When training the stylizable radiance field, we fix the pre-trained  $D_{\text{VGG}}$  so that it preserves the knowledge about the distribution of VGG features from diverse images, which induces compatibility with AdaIN.

**Training scene content field.** We train the scene content field by optimizing the tri-plane grid features and MLP. In detail, for the input 3D point  $\mathbf{x}_i$  and the view direction vector  $\mathbf{d}$ , the grid features of the scene content field are decoded into the 3D point density  $\sigma_i$  and the content feature (see the green grid and MLP in Fig. 1). We further decode the content feature into RGB values using the pre-trained  $D_{\text{VGG}}$ . At the end,

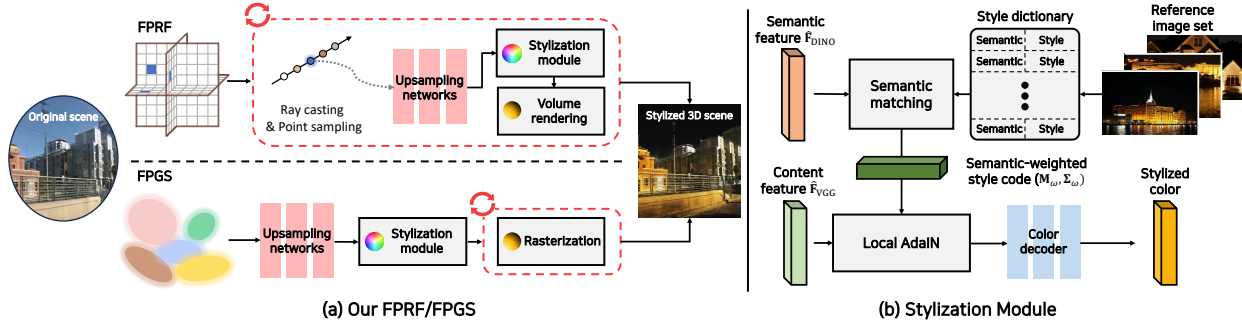


Fig. 2: **Stylization stage.** (a) The stylization module and rendering process of FPRF are intertwined so that the time-consuming stylization process is required for each new view rendering. In contrast, FPGS stylizes whole 3D Gaussians at once before rendering, and the stylized scene can be rendered in real-time. (b) Given the optimized stylizable radiance field and the set of arbitrary reference images, we stylize the large-scale 3D scene via our novel semantic-aware local AdaIN. We compose a style dictionary consisting of local semantic codes and local style codes pairs extracted from the clustered reference images. Using semantic features from the stylizable radiance as a query, we find the corresponding local semantic features and retrieve the paired local style codes. Using the retrieved semantic-style code pairs, we perform semantic matching and local AdaIN, then finally render the stylized colors.

we render the content feature  $\hat{\mathbf{F}}_{VGG}^{NF}(\mathbf{r})$  and the scene color  $\hat{\mathbf{C}}(\mathbf{r})$  via volume rendering (Eq. (5)).

To perform AdaIN, we guide the content feature map  $\hat{\mathbf{F}}_{VGG}^{NF}$  to follow VGG feature distribution by feature distillation. We distill the ground truth features  $\mathbf{F}_{VGG}(\mathbf{I})$  obtained from input images  $\mathcal{I} = \{\mathbf{I}^i\}_{i=1\dots C}$  via pre-trained VGGNet, by minimizing the error between  $\mathbf{F}_{VGG}(\mathbf{I})$  and the volume rendered features  $\hat{\mathbf{F}}_{VGG}^{NF}(\mathbf{r})$ . Note that the ground truth VGG feature maps are up-sampled to pixel resolution with guided filtering. Since the scene content field needs to reconstruct the accurate scene geometry and appearance, we compute the photometric loss for the volume rendered color  $\hat{\mathbf{C}}(\mathbf{r})$ . Also, typical regularization losses  $\mathcal{L}_{reg}$  are employed (Fridovich-Keil et al, 2023). The total loss function for training the scene content field is as below:

$$\mathcal{L}_{content} = \sum_{\mathbf{r} \in \mathcal{R}} \|\hat{\mathbf{F}}_{VGG}^{NF}(\mathbf{r}) - \mathbf{F}_{VGG}(\mathbf{I}, \mathbf{r})\|_2^2 + \lambda_{RGB} \sum_{\mathbf{r} \in \mathcal{R}} \|\hat{\mathbf{C}}(\mathbf{r}) - \mathbf{C}(\mathbf{I}, \mathbf{r})\|_2^2 + \lambda_{reg} \mathcal{L}_{reg}, \quad (6)$$

where  $\mathcal{R}$  is the set of sampled rays in each training batch, and  $\mathbf{F}_{VGG}(\mathbf{I}, \mathbf{r})$  and  $\mathbf{C}(\mathbf{I}, \mathbf{r})$  denote ground truth VGG features and RGB values of the pixels correspond to the ray  $\mathbf{r} \in \mathcal{R}$ .

Note that we keep  $D_{VGG}$  frozen after its pre-train stage, *i.e.*,  $D_{VGG}$  is fixed during the training stage of the scene content field. This differs from StyleRF (Liu et al, 2023), which needs to fine-tune a CNN-based decoder for each scene.

**Feed-forward stylization using the scene content field.** After training the scene content field, we can

perform PST with an arbitrary style image in the stylization stage. In other words, we train the scene content field once and perform PST on a trained radiance field in a feed-forwards manner, *i.e.*, without per-style or per-scene optimization.

Given a reference image  $\mathbf{I}_s$ , we start with extracting the VGG feature  $\mathbf{F}_{VGG}(\mathbf{I}_s)$ . Then we stylize 3D content features  $\hat{\mathbf{F}}_{VGG}^{NF}$  as  $\sigma_s \frac{(\hat{\mathbf{F}}_{VGG}^{NF} - \mu_c)}{\sigma_c} + \mu_s$ , where  $\mu_s$  and  $\sigma_s$  are the mean and standard deviation of  $\mathbf{F}_{VGG}(\mathbf{I}_s)$ . By decoding the stylized content features to RGB values using the pre-trained color decoder  $D_{VGG}$ , we can render a stylized 3D scene. The mean and standard deviation of  $\hat{\mathbf{F}}_{VGG}^{NF}(\mathbf{r})$  are kept tracked with the moving average during training (Liu et al, 2023).

#### 4.2 Multi-reference image 3D Scene PST via Semantic matching and Local AdaIN

With the trained scene content field and AdaIN, we can efficiently transfer the style of images to the 3D radiance field. However, it often fails to produce satisfactory results when it comes to large-scale 3D scenes: AdaIN only allows a single reference image which often cannot cover all components in the large-scale scene. To overcome this limitation, we propose a multi-reference based 3D scene PST by semantically matching the radiance field and multiple reference images. As shown in the Fig. 2-(b), the process involves two steps. First, we compose a style dictionary containing local semantic-style code pairs obtained from semantically clustered reference images. Then, we perform a semantic-aware style transfer by

leveraging the semantic correspondence between the 3D scene and each element of the composed style dictionary.

**Reference image clustering.** To stylize a 3D scene with multiple style reference images, we consider a set of reference images,  $\mathcal{I}_s = \{\mathbf{I}_s^i\}_{i=1, \dots, N}$ , where  $N$  denotes the number of reference images we use. We compose a compact style dictionary  $\mathcal{D}$  with the reference images by clustering them with similar styles and semantics. We first extract semantic feature maps to cluster the reference images according to semantic similarity. We employ DINO (Caron et al, 2021) as a semantic feature encoder, which can be generalized to various domains by being trained on large-scale datasets in a self-supervised manner. We then apply K-means clustering to the extracted semantic feature map  $\mathbf{F}_{\text{DINO}}(\mathbf{I}_s^i)$ , and obtain semantically correlated  $M$  number of clusters  $\mathcal{S} = \{\mathbf{S}^{ij}\}_{i=1, \dots, N}^M$  from each reference image  $\mathbb{I}_s^i$ .

We obtain local style codes from the clusters by extracting another feature map  $\mathbf{F}_{\text{VGG}}(\mathbf{I}_s^i)$  from each reference image with VGGNet (Simonyan and Zisserman, 2014). Then we obtain the local style code, mean  $\boldsymbol{\mu}_s^{ij}$  and standard deviation  $\boldsymbol{\sigma}_s^{ij}$ , from VGG features  $\mathbf{F}_{\text{VGG}}(\mathbf{I}_s^{ij}) \in \mathbf{S}^{ij}$  assigned to each cluster. The centroid  $\hat{\mathbf{F}}_{\text{DINO}}(\mathbf{I}_s^{ij})$  of each clustered semantic feature and the assigned local style code  $(\boldsymbol{\mu}_s^{ij}, \boldsymbol{\sigma}_s^{ij})$  compose a key-value pair for the style dictionary  $\mathcal{D}$ , as  $\mathcal{D} = \{\hat{\mathbf{F}}_{\text{DINO}}(\mathbf{I}_s^{ij}) : (\boldsymbol{\mu}_s^{ij}, \boldsymbol{\sigma}_s^{ij})\}_{i=1, \dots, N}^M$ . With this compact style dictionary, we can efficiently perform local style transfer using multiple reference images by semantically matching the clusters with the 3D scene.

**Scene semantic field.** To semantically match the 3D scene and the reference image clusters, we design and learn an auxiliary 3D feature grid, called scene semantic field. The scene semantic field contains semantic features of the 3D scene (Kobayashi et al, 2022; Tschernetzki et al, 2022; Kerr et al, 2023). To distill semantic features to the scene semantic field, we optimize the tri-plane features and MLP (orange grid and MLP in Fig. 1) by minimizing the error between rendered features  $\hat{\mathbf{F}}_{\text{DINO}}(\mathbf{r})$  and features extracted from the input images by DINO as follows:

$$\mathcal{L}_{\text{semantic}} = \sum_{\mathbf{r} \in \mathcal{R}} \|\hat{\mathbf{F}}_{\text{DINO}}^{\text{NF}}(\mathbf{r}) - \mathbf{F}_{\text{DINO}}(\mathbf{I}, \mathbf{r})\|_1, \quad (7)$$

where  $\mathbf{F}_{\text{DINO}}(\mathbf{I}, \mathbf{r})$  denotes ground truth DINO features matched to ray  $\mathbf{r}$ . The density  $\sigma$  from the scene content field is used for volume rendering, and  $\mathcal{L}_{\text{semantic}}$  does not affect the learning of the density. We do not query view direction as input, in order to preserve multi-view semantic consistency. Also, for constructing a fine-grained scene semantic field, the ground truth DINO feature maps are refined by guided filtering (He et al, 2012). The guided filtering enables consistent distillation of

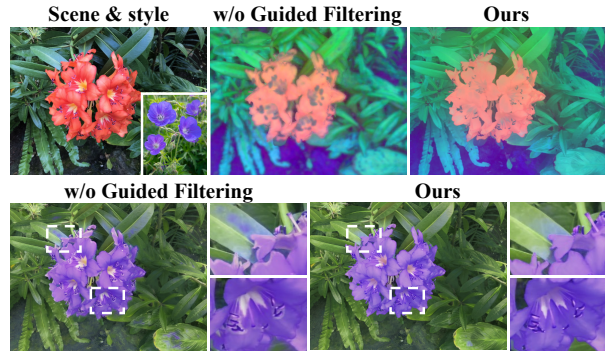


Fig. 3: **Effects of guided filtering on semantic features.** [Top] Given the trained 3D scene and the reference image (left), we visualize the learned stylizable radiance field using PCA without (mid) / with (right) guided filtering. The learned semantic features are much sharper when guided filtering is applied. [Bottom] The stylizable radiance field shows degraded stylization results if learned without guided filtering, *e.g.*, blurry boundaries (left), higher stylization quality when learned with guided filtering (right).

semantic features, resulting in clean and photorealistic stylized outputs (see Fig. 3).

#### Semantic correspondence matching & Local AdaIN.

When rendering the stylized 3D scene, we use two semantic feature matrices for computing semantic correspondence between the 3D scene and the elements of the style dictionary  $\mathcal{D}$ . The first one is  $\hat{\mathbf{F}}_{\text{DINO}}^{\text{NF}} \in \mathbb{R}^{K \times C_D}$ , obtained from the scene semantic field, where  $K$  is the number of queried 3D points, and  $C_D$  is the channel size of the semantic feature, *i.e.*, DINO feature. The other one is  $\bar{\mathbf{F}}_{\text{DINO}}(\mathbf{I}_s) \in \mathbb{R}^{T \times C_D}$  comprising keys  $\bar{\mathbf{F}}_{\text{DINO}}(\mathbf{I}_s^{ij})$  of  $\mathcal{D}$ , which are the centroids of the  $T$  clusters. Given semantic feature matrices,  $\hat{\mathbf{F}}_{\text{DINO}}^{\text{NF}}$  and  $\bar{\mathbf{F}}_{\text{DINO}}(\mathbf{I}_s)$ , we compute a cross-correlation matrix  $\mathbf{R}$  as:

$$\mathbf{R} = \hat{\mathbf{F}}_{\text{DINO}}^{\text{NF}} \bar{\mathbf{F}}_{\text{DINO}}(\mathbf{I}_s)^\top. \quad (8)$$

To map local styles of the reference images according to  $\mathbf{R}$ , we compose two matrices,  $\mathbf{M}(\mathbf{F}_{\text{VGG}}(\mathbf{I}_s)) \in \mathbb{R}^{T \times C_V}$  and  $\boldsymbol{\Sigma}(\mathbf{F}_{\text{VGG}}(\mathbf{I}_s)) \in \mathbb{R}^{T \times C_V}$ , comprising the local style codes  $(\boldsymbol{\mu}_s^{ij}, \boldsymbol{\sigma}_s^{ij})$  from the style dictionary, where  $C_V$  denotes the channel size of VGG feature map. We compute the matrix form of semantic-weighted style codes  $(\mathbf{M}_w, \boldsymbol{\Sigma}_w)$  as follows:

$$\mathbf{M}_w = \mathbf{R}^{\text{S}} \mathbf{M}(\mathbf{F}_{\text{VGG}}(\mathbf{I}_s)), \quad \boldsymbol{\Sigma}_w = \mathbf{R}^{\text{S}} \boldsymbol{\Sigma}(\mathbf{F}_{\text{VGG}}(\mathbf{I}_s)), \quad (9)$$

where  $\mathbf{R}^{\text{S}} = \text{Softmax}(\mathbf{R})$  demonstrates style attention assigned to the queried 3D point features. The semantic-weighted style code  $(\mathbf{M}_w, \boldsymbol{\Sigma}_w)$  is assigned to each 3D point according to the style attention. We feed

these semantic-weighted style codes to AdaIN layer as  $\Sigma_w \frac{(\hat{\mathbf{F}}_{\text{VGG}}^{\text{NF}} - \mu_c)}{\sigma_c} + \mathbf{M}_w$ , followed by the decoder  $D_{\text{VGG}}$  decoding the stylized features into colors. Finally, we perform volumetric rendering to obtain the final stylized scene. Note that this semantic-weighted local AdaIN preserves the multi-view stylized color consistency by directly measuring semantic correspondence between reference images and features on the scene semantic fields (Kobayashi et al, 2022).

We found that computing  $\mathbf{M}_w$  and  $\Sigma_w$  on the feature map resolution without clustering (Gunawan et al, 2023) is inefficient for the volume rendering framework where iterative rendering of rays is inevitable. Our clustering enables efficient style transfer, especially for cases using multiple reference images. We highlight that photorealistic scene stylization results are obtained when  $\sim 10$  clusters are used. It is worth noting that the clustering process takes no more than 1 *sec.* for each reference image. Using the small number of clusters allows us to avoid iterative concatenation of high-dimensional reference image features and effectively reduces the computational cost of matrix multiplication.

## 5 Feed-Forward Photorealistic Style Transfer of 3D Gaussians - FPGS

In this section, we propose FPGS, a feed-forward photorealistic style transfer method of scenes represented by 3D Gaussians (Kerbl et al, 2023). FPGS is an extended version of FPRF, which achieves real-time rendering by redesigning the style transfer process of FPRF and leveraging Gaussian Splatting. In Sec. 5.1, we propose a novel feature field representation by using an RGB to high-dimensional feature mapping network. In Sec. 5.2, we elaborate on algorithmic enhancements of FPGS from FPRF, in Sec. 5.3, we propose an iterative style transfer method enhancing local style transfer quality.

### 5.1 Feature fields with 3D Gaussians

In FPGS, we represent a target scene with explicit 3D Gaussians containing high-dimensional content features and semantic features, which correspond to the scene content field and the scene semantic field of FPRF. Similar to Eq. (5), high-dimensional features on 3D Gaussians can be rendered by modifying Eq. (3) as:

$$\hat{\mathbf{F}}(\mathbf{r}) = \sum_{\mathbf{g}_i \in \mathcal{G}_r} \hat{\mathbf{f}}_i \alpha_i^{\text{GS}} \prod_{j=1}^{i-1} (1 - \alpha_j^{\text{GS}}), \quad (10)$$

where  $\hat{\mathbf{f}}_i$  is the high-dimensional feature assigned to each 3D Gaussian  $\mathbf{g}_i$ . We can train feature fields represented

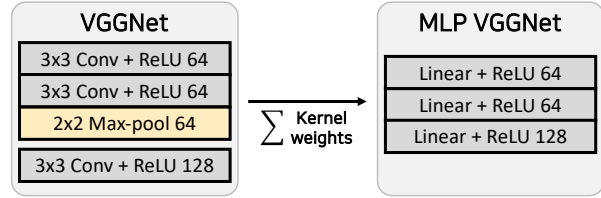


Fig. 4: **Architecture of the MLP VGGNet.** We convert the original VGGNet to MLP VGGNet by mimicking computation of CNN in homogeneous region. The output of the MLP VGGNet approximates the original VGG features and enables style transfer with AdaIN by mapping RGB values to VGG features.

with 3D Gaussians by minimizing error between rendered  $\hat{\mathbf{F}}(\mathbf{r})$  and 2D image features extracted from input images. However, directly assigning high-dimensional features to 3D Gaussians is memory intensive and rendering high-dimensional features is time-consuming (Qin et al, 2024; Liu et al, 2024a; Zhou et al, 2024; Jun-Seong et al, 2025).

**Scene content field.** To address these inefficiencies, we circumvent the feature distillation process of the scene content field. Different from FPRF which reconstructs the scene content field by distilling features extracted from VGGNet, FPGS first trains the scene content field with RGB loss  $\mathcal{L}_{\text{RGB}}$  only, as shown in the Fig. 1. Then we employ a mapping network which transforms the RGB values of the 3D Gaussians into high-dimensional features which follow VGG feature distribution. By leveraging the mapping network as an upsampling network, we can implement the scene content field without feature distillation, which requires memory-intensive assignments of high-dimensional features on 3D Gaussians. However, we cannot utilize the original VGGNet as a mapping network, whose 2D CNN architecture is not compatible with mapping the RGB values of Gaussians in 3D space.

To alleviate this issue, we propose an MLP VGGNet, which maps a single RGB value of each Gaussian to a VGG feature by interpreting the computation of VGGNet’s CNN architecture with an MLP. To map a single RGB value to a VGG feature, we can consider a case we predict a VGG feature map from an image containing only a single RGB value. In this case, the same input value is multiplied with each weight of convolution kernels and summed. As a result, the CNN architecture of VGGNet activates as an MLP, which comprises the same number of neurons as the number of convolution kernels in VGGNet. The  $j$ th weight in  $i$ th layer of the



MLP is represented as:

$$w_{ij}^{\text{MLP}} = \sum_{k=1}^{S_{ij}} w_{ijk}^{\text{VGG}}, \quad (11)$$

where  $S_{ij}$  is the kernel size and  $w_{ijk}^{\text{VGG}}$  is the kernel weight of  $j$ th convolution kernel in  $i$ th layer of VGGNet. Motivated by this intuitive case example, we convert VGGNet to a novel MLP network called MLP VGGNet. As shown in Fig. 4, we sum all weights of each kernel in the VGGNet up to the ReLU2.1 layer, and construct an MLP with the summed weights. Pooling layers of VGGNet are removed while ReLU layers are maintained after the weight-distillation process. Using the obtained MLP VGGNet, we represent the scene content field by predicting VGG features with the all RGB values obtained from the pre-optimized 3D Gaussians. The content features from the MLP VGGNet are compatible with the decoder  $D_{\text{VGG}}$  trained with the original VGGNet, which enables style transfer with 3D Gaussians using AdaIN. The MLP VGGNet enables the construction of the scene content field without high-dimensional feature distillation process during scene optimization.

This weight-distilled MLP VGGNet effectively predicts the original VGG feature from a single RGB value. The original VGGNet activates identically to the MLP VGGNet when the input image contains only a single RGB value, which means the receptive field of the target pixel is a homogeneous area. In our method, we approximate features from the shallow ReLU2.1 layer, which has relatively small receptive fields. The smaller receptive fields hold higher homogeneity, which allows the MLP VGGNet to accurately approximate the original VGG features from the shallow layers. Note that the proposed converting method does not require any additional training or a fine-tuning process to obtain the MLP VGGNet, so that we can leverage pre-trained VGGNet instantly.

**Scene semantic field.** We cannot employ the same weight-distilled MLP for representing the scene semantic field since semantic encoders require large receptive fields to acquire semantic information from the entire image. Inspired by Langsplat (Qin et al, 2024), we train a scene-specific semantic feature autoencoder for compression, which maps high-dimensional features to low-dimensional latent space, and upsamples again to the high-dimensional features. Before scene reconstruction, we extract semantic feature maps  $\mathbf{F}_{\text{DINO}}(\mathbf{I})$  from the training images  $\mathcal{I}$  with DINO, and train the autoencoder

with:

$$\begin{aligned} \mathcal{L}_{\text{AE}} = & \sum_{\mathbf{I} \in \mathcal{I}} \|D_{\text{AE}}(E_{\text{AE}}(\mathbf{F}_{\text{DINO}}(\mathbf{I}))) - \mathbf{F}_{\text{DINO}}(\mathbf{I})\|_2^2 \\ & + \lambda_{\text{cos}} \sum_{\mathbf{I} \in \mathcal{I}} (1 - \text{sim}(D_{\text{AE}}(E_{\text{AE}}(\mathbf{F}_{\text{DINO}}(\mathbf{I}))), \mathbf{F}_{\text{DINO}}(\mathbf{I}))), \end{aligned} \quad (12)$$

where  $E_{\text{AE}}$  and  $D_{\text{AE}}$  denote the encoder and decoder of the autoencoder, and  $\text{sim}(\cdot, \cdot)$  represents the similarity measure, which in this case is cosine similarity. With the trained encoder, we encode high-dimensional feature maps  $\mathbf{F}_{\text{DINO}}(\mathbf{I})$  to low-dimensional feature maps, and distill them to 3D Gaussians with Eq. (10) during the scene optimization. After optimization, the stylizable 3D Gaussians can be parameterized as:  $\mathcal{G} = \{\mathbf{g}_i = (\mathbf{p}_i, \boldsymbol{\Sigma}_i, \sigma_i, \mathbf{K}_i, \hat{\mathbf{f}}_i^{\text{GS}})\}_{i=1, \dots, P}$ , where  $\hat{\mathbf{f}}_i^{\text{GS}}$  is a distilled low-dimensional semantic feature. In stylization process, we decode  $\hat{\mathbf{f}}_i^{\text{GS}}$  to a high-dimensional semantic feature  $(\hat{\mathbf{f}}_{\text{DINO}}^{\text{GS}})_i$  and use it for semantic matching. The decoder  $D_{\text{AE}}$  and the MLP VGGNet activate as upsampling networks which map low-dimensional semantic features and RGB values to high-dimensional features, enabling reconstruction of high-dimensional feature fields with 3D Gaussians.

## 5.2 Holistic scene style transfer and real-time rendering

In contrast to FPRF, FPGS utilizes an explicit representation for scene reconstruction and editing. FPRF uses an implicit representation to reconstruct feature fields, which requires query point sampling and a point-wise inference process to obtain high-dimensional features from 3D space. To stylize the scene using the queried high-dimensional features, the stylization process is needed for every new view rendering, which requires extensive computational cost and slows down the rendering speed.

To overcome this limitation, as shown in Fig. 2, we decouple the stylization process and the rendering process of FPGS by leveraging explicit nature of 3D Gaussians, which enables real-time rendering. We first decode the low-dimensional semantic features contained in all  $P$  3D Gaussians to the high-dimensional semantic features  $\hat{\mathbf{F}}_{\text{DINO}}^{\text{GS}} \in \mathbb{R}^{P \times C_D}$ , using  $D_{\text{AE}}$ . The decoded explicit 3D features from Gaussians enable semantic matching between the entire scene and reference images  $\mathbf{I}_s$  using modified Eq. (8) and Eq. (9) as:

$$\begin{aligned} \mathbf{R} &= \hat{\mathbf{F}}_{\text{DINO}}^{\text{GS}} \bar{\mathbf{F}}_{\text{DINO}}(\mathbf{I}_s)^\top \\ \mathbf{M}_w &= \mathbf{R}^s \mathbf{M}(\mathbf{F}_{\text{VGG}}(\mathbf{I}_s)), \quad \boldsymbol{\Sigma}_w = \mathbf{R}^s \boldsymbol{\Sigma}(\mathbf{F}_{\text{VGG}}(\mathbf{I}_s)). \end{aligned} \quad (13)$$

The semantic-weighted style code  $(\mathbf{M}_w, \boldsymbol{\Sigma}_w)$  is assigned to all 3D Gaussians, which contains semantically corresponding styles from the reference images  $\mathbf{I}_s$ . To stylize

the whole 3D Gaussians with the obtained style code, we acquire the scene content features  $\hat{\mathbf{F}}_{\text{VGG}}^{\text{GS}} \in \mathbb{R}^{P \times C_V}$ , by encoding base colors  $\mathbf{C}^{\text{diff}} = [\mathbf{c}_1^{\text{diff}}, \mathbf{c}_2^{\text{diff}}, \dots, \mathbf{c}_P^{\text{diff}}]^\top \in \mathbb{R}^{P \times 3}$  of 3D Gaussians (see Sec. 3.2) with the MLP VG-GNet. Then we stylize the 3D Gaussians using AdaIN as:

$$\begin{aligned} (\hat{\mathbf{F}}_{\text{VGG}}^{\text{GS}})' &= \boldsymbol{\Sigma}_w \frac{(\hat{\mathbf{F}}_{\text{VGG}}^{\text{GS}} - \boldsymbol{\mu}_c)}{\boldsymbol{\sigma}_c} + \mathbf{M}_w \\ (\mathbf{C}^{\text{diff}})' &= D_{\text{VGG}}((\hat{\mathbf{F}}_{\text{VGG}}^{\text{GS}})'), \end{aligned} \quad (14)$$

where  $\boldsymbol{\mu}_c$  and  $\boldsymbol{\sigma}_c$  are the mean and standard deviation of  $\hat{\mathbf{F}}_{\text{VGG}}^{\text{GS}}$ , and  $(\mathbf{C}^{\text{diff}})' \in \mathbb{R}^{P \times 3}$  denotes the stylized base color of 3D Gaussians. With the stylized base color, we update SH coefficients  $\mathbf{K}_i$  of each 3D Gaussian  $\mathbf{g}_i$  using Eq. (2) as:  $(\mathbf{K}_i^{[:,1:2]})' = (\mathbf{c}_i^{\text{diff}})' / C_{\text{SH}}$ . This process stylizes the entire scene at once in a feed-forward manner. After we obtain the stylized Gaussians, we can render the stylized scene in real-time without rendering the high-dimensional features.

### 5.3 Iterative style transfer

To enhance local style transfer quality, we propose an iterative style transfer method. By decoupling the style transfer and the rendering process, we can perform additional style transfer process without compromising rendering speed. After obtaining semantic-weighted style code  $(\mathbf{M}_w, \boldsymbol{\Sigma}_w)$  by semantic correspondence matching, we perform local AdaIN iteratively as:

$$\begin{aligned} (\hat{\mathbf{F}}_{\text{VGG}}^{\text{GS}})^i &= E_{\text{VGG}}((\mathbf{C}^{\text{diff}})^i) \\ (\hat{\mathbf{F}}_{\text{VGG}}^{\text{GS}})^{i+1} &= \boldsymbol{\Sigma}_w \frac{((\hat{\mathbf{F}}_{\text{VGG}}^{\text{GS}})^i - \boldsymbol{\mu}_c^i)}{\boldsymbol{\sigma}_c^i} + \mathbf{M}_w \\ (\mathbf{C}^{\text{diff}})^{i+1} &= D_{\text{VGG}}((\hat{\mathbf{F}}_{\text{VGG}}^{\text{GS}})^{i+1}), \end{aligned} \quad (15)$$

where  $(\hat{\mathbf{C}}^{\text{diff}})^i$  and  $(\hat{\mathbf{F}}_{\text{VGG}}^{\text{GS}})^i$  denotes the diffuse colors and content features at  $i$ th iteration,  $\boldsymbol{\mu}_i$  and  $\boldsymbol{\sigma}_i$  are the mean and standard deviation of  $(\hat{\mathbf{F}}_{\text{VGG}}^{\text{GS}})^i$ . In each style transfer process, the content features are normalized by the global style code  $(\boldsymbol{\mu}_c^i, \boldsymbol{\sigma}_c^i)$  and transformed with the semantic-weighted local style code  $(\mathbf{M}_w, \boldsymbol{\Sigma}_w)$ . By updating the style of the scene iteratively, we can enhance the locality of the stylization, which transfers the local style to the semantically corresponding regions of the 3D scene more precisely (see Fig. 10).

## 6 Experiments

**Datasets.** We evaluate our methods on two public datasets, the LLFF (Mildenhall et al, 2019) dataset for small scenes and the San Fran Cisco Mission Bay

dataset (Tancik et al, 2022) for large-scale scenes. The LLFF dataset includes 8 real forward facing scenes and the San Fran Cisco Mission Bay dataset is a city scene dataset consisting of about 12,000 images recorded by 12 cameras. For 4D scene experiments, we use KITTI-360 (Liao et al, 2022) comprising 9 urban driving sequences, and we also use Tanks and Temples (Knapitsch et al, 2017) containing unbounded scenes for additional experiments.

### 6.1 Qualitative results

**PST on small-scale scenes.** We compare our model with three competing 3D PST methods; Instant-NeRF-Stylization (Li and Pan, 2024), UPST-NeRF (Chen et al, 2024), LipRF (Zhang et al, 2023b), and recent feed-forward 3D *artistic* style transfer methods, StyleRF (Liu et al, 2023) and StyleGaussian (Liu et al, 2024a). As shown in Fig. 5, our methods transfer the diverse colors of the style reference image well while maintaining the texture fidelity of the original scene. While UPST-NeRF effectively retains the structure of the scene, the stylized scene’s color style differs from the reference. LipRF stabilizes the artifacts caused by 2D PST methods (Yoo et al, 2019; Wu et al, 2022) with the Lipschitz network, but it tends to oversmooth the scene and loses the color diversity of the reference image. Instant-NeRF-Stylization and StyleRF fail to obtain photorealistic results. Note that UPST-NeRF, StyleRF, and StyleGaussian require time-consuming per-scene optimization after scene reconstruction for style transfer, LipRF and Instant-NeRF-Stylization need per-style optimization. On the contrary, our methods achieve feed-forward PST after an efficient training process. To compare our methods with LipRF which is not open-sourced, we brought the qualitative results from LipRF paper. In other words, the scenes in Fig. 5 are not cherry-picked.

**PST on large-scale scenes.** We compare our methods against two competing 2D PST methods, CCPL (Wu et al, 2022) and PhotoWCT<sup>2</sup> (Chiu and Gurari, 2022), since we propose the first method aiming for large-scale 3D radiance field PST, no 3D PST method supports large-scale datasets. Figure 6 shows a qualitative comparison on large-scale scenes. We observe that 2D PST methods fail to preserve multi-view color consistency under wide-range view changes, *e.g.*, they obtain different colors of the same building and sky as the viewpoints change. Also, they stylize images without understanding of semantic relation between the content image and the reference image. Instead, our methods elaborately transfer styles by reflecting the semantic correspondence between scene and reference images. Semantic matching



Fig. 5: **PST quality comparison on the LLFF dataset (Mildenhall et al, 2019)**. Compared to the competing 3D PST methods, our methods stylize the radiance field in a photorealistic manner by transferring the diverse color of the reference image while preserving the original images’ naturalness and vividness. **Instant** denotes Instant-NeRF-Stylization (Li and Pan, 2024) method, and **WCT<sup>2</sup> + LipRF** denotes LipRF (Zhang et al, 2023b) based on the 2D PST method, **WCT<sup>2</sup>** (Yoo et al, 2019)

Table 2: Multi-view consistency comparison. [Top] Comparison with feed-forward 3D style transfer methods. [Bottom] Comparison with 2D PST methods. Short denotes short baseline and long denotes wide baseline of camera settings.

LLFF dataset				
	StyleGaussian	UPST-NeRF	FPRF (Ours)	FPGS (Ours)
Short ( $\downarrow$ )	0.0264	0.0234	0.0231	0.0214
Long ( $\downarrow$ )	0.0440	0.0384	0.0399	0.0349
San Fran Cisco Mission Bay dataset				
	CCPL	PhotoWCT <sup>2</sup>	FPRF (Ours)	FPGS (Ours)
Short ( $\downarrow$ )	0.0528	0.0525	0.0395	0.0335
Long ( $\downarrow$ )	0.0775	0.0755	0.0563	0.0502

can also preserve multi-view consistency by directly measuring semantic correspondence between the reference images and the 3D scene semantic field directly.

More qualitative results from diverse scenes are given in our supplementary material, which shows the generalizability of our methods.

## 6.2 Quantitative results

**Multi-view consistency.** We report multi-view consistency of our method by following previous 3D style transfer works (Virmaux and Scaman, 2018; Chen et al, 2024; Liu et al, 2023). Multi-view consistency denotes a

Table 3: Running time comparison with feed-forward 3D style transfer methods on the LLFF dataset Mildenhall et al (2019). FPGS requires less training time than the other methods and achieves real-time rendering.

	Train (min)			Render (sec)
	pre-/post-optimization	scene recon.	total	
StyleRF	301	40	341	16.62
StyleGaussian	347	11	358	0.004
UPST-NeRF	650	6	656	1.357
FPRF (Ours)	0	61	61	3.812
FPGS (Ours)	4	20	24	0.004

Table 4: User Study on the quality of PST with 3D PST methods. We evaluate on the LLFF Mildenhall et al (2019) dataset and the rating is of scale 1-5. **Instant** denotes Instant-NeRF-Stylization Li and Pan (2024) method.

	Instant	UPST-NeRF	FPRF (Ours)	FPGS (Ours)
PST quality ( $\uparrow$ )	1.305	3.279	3.576	3.936

feature that a shared geometry wherein a scene observed from various angles can consistently explain multiple views of the scene. We evaluate the multi-view consistency of 3D stylization by computing the multi-view error via image warping. Specifically, for the two rendered images  $\mathbf{I}_d$  and  $\mathbf{I}_{d'}$  from two different views  $\mathbf{d}$  and



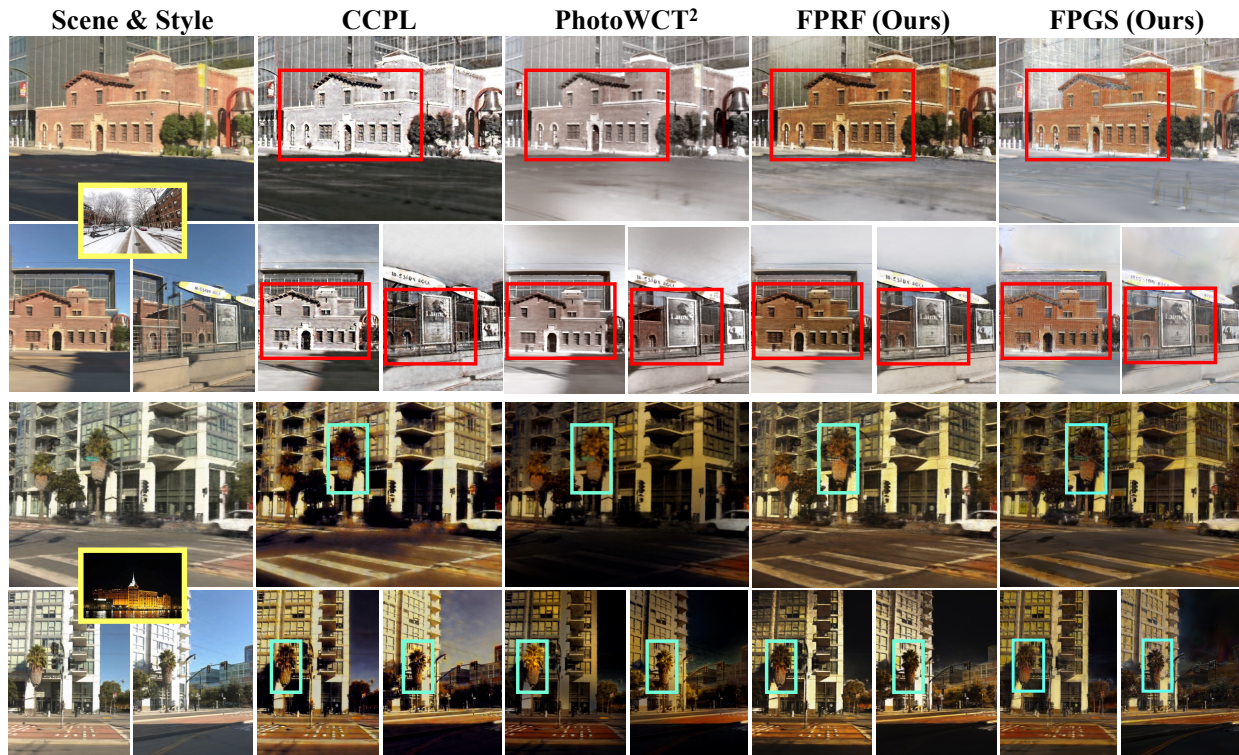


Fig. 6: **Multi-view appearance consistency on the San Francisco Mission Bay dataset (Tancik et al, 2022).** Our methods preserve multi-view appearance consistency even in extreme viewpoint change, while 2D PST methods (Wu et al, 2022; Chiu and Gurari, 2022) produce inconsistent colors of the same building as the viewpoint changes.

$\mathbf{d}'$ , the multi-view error is computed as:

$$E_{\text{warp}}(\mathbf{I}_{\mathbf{d}}, \mathbf{I}_{\mathbf{d}'}) = \text{RMSE}(\mathbf{I}_{\mathbf{d}}, W(\mathbf{I}_{\mathbf{d}'}); \mathbf{B}_{\mathbf{d}'\mathbf{d}}), \quad (16)$$

where  $W$  warps  $\mathbf{I}_{\mathbf{d}'}$  to  $\mathbf{I}_{\mathbf{d}}$  and  $\mathbf{B}_{\mathbf{d}'\mathbf{d}}$  is the binary mask of valid pixels warped from the view  $\mathbf{d}'$  to  $\mathbf{d}$ . The warping function  $W$  and binary mask  $\mathbf{B}_{\mathbf{d}'\mathbf{d}}$  are measured from the rendered original scene images, with an off-the-shelf optical flow method (Teed and Deng, 2020). The non-valid pixels are masked out and are not considered.

We compare our method with feed-forward 3D style transfer methods (Liu et al, 2024a; Chen et al, 2024) on the 8 scenes of the LLFF dataset (Mildenhall et al, 2019), and with 2D PST methods (Wu et al, 2022; Chiu and Gurari, 2022) on the 4 blocks from the San Francisco Mission Bay dataset (Tancik et al, 2022). We use 20 style images from the PST dataset (Luan et al, 2017) for evaluation. We report warp errors in both short-/long-term settings denoting short-/wide-baselines of cameras. Table 2 shows the superior multi-view consistency of FPGS compared to prior methods. Note that our methods effectively preserve multi-view consistency by computing semantic correspondence with features on the 3D space directly, not on the rendered features. Our

scene semantic field contains view-independent semantic features in the 3D space, which ensures that each 3D point is always stylized with the assigned same local style, regardless of view directions.

**Running time.** Table 3 shows FPGS significantly outperforms other methods in terms of training and rendering time. As training time, we measure the total time the model required to become ready for feed-forward style transfer with arbitrary styles, which includes scene reconstruction time and per-scene pre-/post-optimization time. Note that scene reconstruction time highly depends on the efficiency of the off-the-shelf baseline scene representation which can be substituted. In other words, the key process deciding the training time of feed-forward 3D style transfer model is per-scene post-optimization, which is required to the previous methods (Chen et al, 2024; Liu et al, 2023, 2024a) to obtain generalizability for arbitrary styles. Our methods effectively circumvent this per-scene post optimization by using pre-trained MLP color decoder enabling style transfer using arbitrary style, without per-scene fine-tuning. FPGS only requires a short pre-optimization to





Fig. 7: PST on the KITTI-360 dataset (Liao et al, 2022) containing 4D urban scenes. FPGS can be applied to 4D large-scale scenes with time/multi-view consistency.

train a semantic feature autoencoder for compressing the semantic features on 3D Gaussians.

FPGS also achieves real-time rendering by decoupling stylization and rendering process. In contrast, NeRF-based methods (Chen et al, 2024; Liu et al, 2023; Kim et al, 2024) intertwine the stylization and rendering processes, which slows down the rendering speed.

**User study.** We conduct a user study to evaluate the perceptual quality of stylization. We evaluate our methods on the LLFF (Mildenhall et al, 2019) dataset by asking 20 volunteers to score (1-5) the stylization quality of 3D PST. As shown in Table 4, FPGS obtained the best score in terms of PST quality.

### 6.3 Applications

**4D scene style transfer.** We can apply FPGS for stylizing 4D scenes represented by time-variant Gaussians. To stylize 4D scenes, we reconstruct stylizable 4D scenes based on a 4D Gaussian modeling method, VEGS (Hwang et al, 2024). VEGS represents a 4D Gaussian  $\mathbf{g}_i^t$  with a time-variant mean  $\boldsymbol{\mu}_i^t = \mathbf{T}_i^t \boldsymbol{\mu}_i$  and a covariance  $\boldsymbol{\Sigma}_i^t = \mathbf{R}_i^t \boldsymbol{\Sigma}_i \mathbf{R}_i^{t\top}$ , where  $\mathbf{T}_i^t$  is a transform matrix of  $\mathbf{g}_i^t$  at timestep  $t$ , and  $\mathbf{R}_i^t$  is the rotation matrix of  $\mathbf{T}_i^t$ . We modify the training process of VEGS to jointly optimize colors and semantic features on the 4D Gaussians. Figure 7 shows the rendered images of stylized 4D autonomous driving scenes from the KITTI-360 dataset (Liao et al, 2022), with different timesteps and viewpoints. As can be seen, the stylized scenes preserve time/multi-view consistency after extreme time/view-

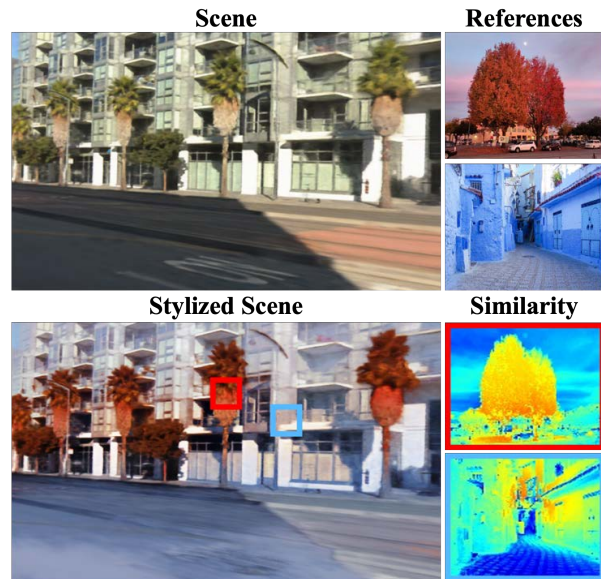


Fig. 8: Multi-reference style transfer. FPGS stylizes the 3D radiance field with multiple reference images. Each heatmap shows the similarity between the rendered semantic features from a highlighted patch and the reference image. Our model comprehends the semantic relationship of a large-scale 3D scene and matches the scene with the reference images.



Fig. 9: Scribble-based style transfer. We render images from the optimized scene and draw scribbles on the images. Our methods can transfer the scribbled colors on the images to the corresponding regions of the scene photorealistically.

point change. Since we simply embeds semantic features on the time-variant Gaussians, our method inherits time/multi-view consistency from the based 4D reconstruction model.



Fig. 10: **Ablation studies.** “w/o Semantic matching” stylizes the model without semantic matching and local AdaIN, *i.e.*, global style transfer. “w/o Iterative style transfer” performs style transfer with one iteration.

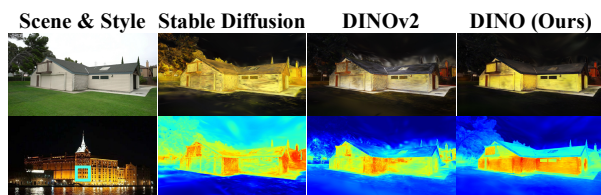


Fig. 11: **Style transfer results with diverse semantic image encoders.** We construct scene semantic fields by distilling image features extracted from different image foundation models. Each heatmap shows the similarity between the features extracted from the highlighted patch of the reference image and the semantic features rendered from the scene semantic field.

**Multi-reference style transfer.** Figure 8 shows the style transfer results using multiple reference images. Our methods effectively select suitable styles from multiple references with the scene semantic field. Semantic similarity is computed by multiplying features from the scene semantic field with DINO (Caron et al, 2021) feature maps extracted from the reference images. The similarity map clearly shows that our methods comprehend the semantic relationship between scenes and reference images.

**Scribble-based style transfer.** Figure 9 shows that our methods can stylize the scene with scribbles. We slightly modified the semantic matching and local AdaIN process, to match the scribbled colors on the rendered images to semantically corresponding region of the original scene (see details in supplementary material). The scribbled colors are transferred to the 3D/4D scenes photorealistically, and the results are consistent after extreme changes of viewpoints and timesteps.

#### 6.4 Additional experiments

**Ablation studies.** As can be seen in Fig. 10, semantic

matching improves perceptual style similarity between the stylized scene and the reference image, by using semantic-weighted local style code for stylization. Iterative style transfer enhances the effect of local style transfer by iteratively stylizing the scene with the obtained semantic-weighted local style code.

**Semantic encoder comparison.** We investigate the effect of semantic encoder selection on the style transfer quality, by training the scene semantic fields with features from the different image foundation models. We test our method with Stable Diffusion v1-5 (Rombach et al, 2022) and DINOv2 (Oquab et al, 2023), which can perform dense semantic correspondence matching with feature distance (Zhang et al, 2023a). To utilize Stable Diffusion as a feature encoder, we extract image features from the intermediate layers of denoising U-Net, by following (Zhang et al, 2023a). As shown in Fig. 11, the semantic correspondence matching and style transfer quality depends on the semantic features from the foundation models. We find DINO (Caron et al, 2021) shows the most perceptually satisfactory stylized results and utilize it as our semantic encoder.

## 7 Conclusion

In this paper, we present FPGS, a novel 3D photorealistic style transfer (PST) method for radiance fields represented by 3D Gaussians. FPGS allows PST in a feed-forward manner by leveraging AdaIN, without sacrificing real-time rendering speed inherited from 3D Gaussian representation. FPGS also supports multi-reference style transfer enabling stylization of large-scale 3D scenes which consist of diverse components.

The current limitation is that the semantic matching performance of our model is bounded by the capability of the semantic image encoder, DINO. Nonetheless, since our model can utilize any semantic encoder for constructing the scene semantic field, the performance of our model stands to benefit from the emergence of more advanced models.

**Data availability statements.** All data supporting the findings of this study are available online. The LLFF dataset can be downloaded from <https://github.com/Fyusion/LLFF>. The San Fran Cisco Mission Bay dataset can be downloaded from <https://waymo.com/research/block-nerf/>. The Tank and Temples dataset can be downloaded from <https://www.tanksandtemples.org/>. The KITTI-360 dataset can be downloaded from <https://www.cvlibs.net/datasets/kitti-360/>.



## References

- Agarwal S, Snavely N, Simon I, Seitz SM, Szeliski R (2009) Building rome in a day. In: *Int. Conf. Comput. Vis.*
- Barron JT, Mildenhall B, Tancik M, Hedman P, Martin-Brualla R, Srinivasan PP (2021) Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In: *Int. Conf. Comput. Vis.*, pp 5855–5864
- Caron M, Touvron H, Misra I, Jégou H, Mairal J, Bojanowski P, Joulin A (2021) Emerging properties in self-supervised vision transformers. In: *Int. Conf. Comput. Vis.*
- Chen A, Xu Z, Geiger A, Yu J, Su H (2022) Tensorf: Tensorial radiance fields. In: *Eur. Conf. Comput. Vis.*, Springer, pp 333–350
- Chen Y, Yuan Q, Li Z, Liu Y, Wang W, Xie C, Wen X, Yu Q (2024) Upst-nerf: Universal photorealistic style transfer of neural radiance fields for 3d scene. *IEEE Transactions on Visualization and Computer Graphics*
- Chiang PZ, Tsai MS, Tseng HY, Lai WS, Chiu WC (2022) Stylizing 3d scene via implicit representation and hypernetwork. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp 1475–1484
- Chiu TY, Gurari D (2022) Photowt2: Compact autoencoder for photorealistic style transfer resulting from blockwise training and skip connections of high-frequency residuals. In: *IEEE Winter Conf. on Applications of Computer Vision (WACV)*
- Cordts M, Omran M, Ramos S, Rehfeld T, Enzweiler M, Benenson R, Franke U, Roth S, Schiele B (2016) The cityscapes dataset for semantic urban scene understanding. In: *IEEE Conf. Comput. Vis. Pattern Recog.*
- Fan Z, Jiang Y, Wang P, Gong X, Xu D, Wang Z (2022) Unified implicit neural stylization. In: *European Conference on Computer Vision*, Springer, pp 636–654
- Fridovich-Keil S, Yu A, Tancik M, Chen Q, Recht B, Kanazawa A (2022) Plenoxels: Radiance fields without neural networks. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp 5501–5510
- Fridovich-Keil S, Meanti G, Warburg FR, Recht B, Kanazawa A (2023) K-planes: Explicit radiance fields in space, time, and appearance. In: *IEEE Conf. Comput. Vis. Pattern Recog.*
- Früh C, Zakhov A (2004) An automated method for large-scale, ground-based city model acquisition. *International Journal of Computer Vision* 60:5–24
- Geiger A, Lenz P, Urtasun R (2012) Are we ready for autonomous driving? the kitti vision benchmark suite. In: *IEEE Conf. Comput. Vis. Pattern Recog.*
- Gunawan A, Kim SY, Sim H, Lee JH, Kim M (2023) Modernizing old photos using multiple references via photorealistic style transfer. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp 12,460–12,469
- He K, Sun J, Tang X (2012) Guided image filtering. *IEEE Trans Pattern Anal Mach Intell* 35(6):1397–1409
- Huang X, Belongie S (2017) Arbitrary style transfer in real-time with adaptive instance normalization. In: *Int. Conf. Comput. Vis.*
- Huang YH, He Y, Yuan YJ, Lai YK, Gao L (2022) Stylized-nerf: 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*, pp 18,342–18,352
- Hwang S, Kim MJ, Kang T, Kang J, Choo J (2024) Vegs: View extrapolation of urban scenes in 3d gaussian splatting using learned priors. *arXiv preprint arXiv:240702945*
- Jun-Seong K, Yu-Ji K, Ye-Bin M, Oh TH (2022) Hdr-plenoxels: Self-calibrating high dynamic range radiance fields. In: *European Conference on Computer Vision*, Springer, pp 384–401
- Jun-Seong K, Kim M, Kim G, Oh TH, Kim JH (2024) Factorized multi-resolution hashgrid for efficient neural radiance fields: Execution on edge-devices. *IEEE Robotics and Automation Letters*
- Jun-Seong K, Kim G, Yu-Ji K, Wang YCF, Choe J, Oh TH (2025) Dr. splat: Directly referring 3d gaussian splatting via direct language embedding registration. *arXiv preprint arXiv:250216652*
- Kerbl B, Kopanas G, Leimkühler T, Drettakis G (2023) 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics* 42(4), URL <https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/>
- Kerbl B, Meuleman A, Kopanas G, Wimmer M, Lanvin A, Drettakis G (2024) A hierarchical 3d gaussian representation for real-time rendering of very large datasets. *ACM Transactions on Graphics (TOG)* 43(4):1–15
- Kerr J, Kim CM, Goldberg K, Kanazawa A, Tancik M (2023) Lurf: Language embedded radiance fields. In: *Int. Conf. Comput. Vis.*
- Kim G, Youwang K, Oh TH (2024) FPRF: Feed-forward photorealistic style transfer of large-scale 3D neural radiance fields. In: *AAAI*
- Knapitsch A, Park J, Zhou QY, Koltun V (2017) Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)* 36(4):1–13
- Kobayashi S, Matsumoto E, Sitzmann V (2022) Decomposing nerf for editing via feature field distillation. In: *Adv. Neural Inform. Process. Syst.*
- Li S, Pan Y (2024) Instant photorealistic neural radiance fields stylization. In: *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, pp 2980–2984
- Liao Y, Xie J, Geiger A (2022) Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45(3):3292–3310
- Lin J, Li Z, Tang X, Liu J, Liu S, Liu J, Lu Y, Wu X, Xu S, Yan Y, et al (2024) Vastgaussian: Vast 3d gaussians for large scene reconstruction. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp 5166–5175
- Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014) Microsoft coco: Common objects in context. In: *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V 13*, Springer, pp 740–755
- Liu K, Zhan F, Chen Y, Zhang J, Yu Y, El Saddik A, Lu S, Xing EP (2023) Stylerf: Zero-shot 3d style transfer of neural radiance fields. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp 8338–8348
- Liu K, Zhan F, Xu M, Theobalt C, Shao L, Lu S (2024a) Style-gaussian: Instant 3d style transfer with gaussian splatting. *arXiv preprint arXiv:240307807*
- Liu Y, Guan H, Luo C, Fan L, Peng J, Zhang Z (2024b) City-gaussian: Real-time high-quality large-scale scene rendering with gaussians. *arXiv preprint arXiv:240401133*
- Luan F, Paris S, Shechtman E, Bala K (2017) Deep photo style transfer. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 4990–4998
- Mildenhall B, Srinivasan PP, Ortiz-Cayon R, Kalantari NK, Ramamoorthi R, Ng R, Kar A (2019) Local light field fusion: Practical view synthesis with prescriptive sampling

- guidelines. *ACM Transactions on Graphics (TOG)* 38(4):1–14
- Mildenhall B, Srinivasan PP, Tancik M, Barron JT, Ramamoorthi R, Ng R (2020) Nerf: Representing scenes as neural radiance fields for view synthesis. In: *Eur. Conf. Comput. Vis.*
- Müller T, Evans A, Schied C, Keller A (2022) Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)* 41(4):1–15
- Nguyen-Phuoc T, Liu F, Xiao L (2022) Snerf: stylized neural implicit representations for 3d scenes. *arXiv preprint arXiv:220702363*
- Nichol K (2016) Painter by numbers, wikiart, 2016. URL <https://www.kaggle.com/c/painter-by-numbers/overview>
- Oquab M, Darcet T, Moutakanni T, Vo H, Szafraniec M, Khalidov V, Fernandez P, Haziza D, Massa F, El-Nouby A, et al (2023) Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:230407193*
- Pollefeys M, Nistér D, Frahm JM, Akbarzadeh A, Mordohai P, Clipp B, Engels C, Gallup D, Kim SJ, Merrell P, et al (2008) Detailed real-time urban 3d reconstruction from video. *International Journal of Computer Vision* 78:143–167
- Qin M, Li W, Zhou J, Wang H, Pfister H (2024) Langsplat: 3d language gaussian splatting. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp 20,051–20,060
- Rombach R, Blattmann A, Lorenz D, Esser P, Ommer B (2022) High-resolution image synthesis with latent diffusion models. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp 10,684–10,695
- Simon T, Joo H, Matthews I, Sheikh Y (2017) Hand keypoint detection in single images using multiview bootstrapping. In: *IEEE Conf. Comput. Vis. Pattern Recog.*
- Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:14091556*
- Snavely N, Seitz SM, Szeliski R (2006) Photo tourism: Exploring photo collections in 3d. In: *SIGGRAPH Conference Proceedings*
- Tancik M, Casser V, Yan X, Pradhan S, Mildenhall B, Srinivasan P, Barron JT, Kretzschmar H (2022) Block-NeRF: Scalable large scene neural view synthesis. In: *IEEE Conf. Comput. Vis. Pattern Recog.*
- Teed Z, Deng J (2020) Raft: Recurrent all-pairs field transforms for optical flow. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II* 16, Springer, pp 402–419
- Tschernezki V, Laina I, Larlus D, Vedaldi A (2022) Neural feature fusion fields: 3d distillation of self-supervised 2d image representations. In: *International Conference on 3D Vision (3DV)*, IEEE, pp 443–453
- Turki H, Ramanan D, Satyanarayanan M (2022) Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs. In: *IEEE Conf. Comput. Vis. Pattern Recog.*, pp 12,922–12,931
- Virmaux A, Scaman K (2018) Lipschitz regularity of deep neural networks: analysis and efficient estimation. *Advances in Neural Information Processing Systems* 31
- Wu Z, Zhu Z, Du J, Bai X (2022) Ccpl: contrastive coherence preserving loss for versatile style transfer. In: *European Conference on Computer Vision*, Springer, pp 189–206
- Yoo J, Uh Y, Chun S, Kang B, Ha JW (2019) Photorealistic style transfer via wavelet transforms. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp 9036–9045
- Zhang J, Herrmann C, Hur J, Cabrera LP, Jampani V, Sun D, Yang MH (2023a) A tale of two features: Stable diffusion complements dino for zero-shot semantic correspondence. *arXiv preprint arXiv:230515347*
- Zhang K, Kolkin N, Bi S, Luan F, Xu Z, Shechtman E, Snavely N (2022) Arf: Artistic radiance fields. In: *European Conference on Computer Vision*, Springer, pp 717–733
- Zhang Z, Liu Y, Han C, Pan Y, Guo T, Yao T (2023b) Transforming radiance field with lipschitz network for photorealistic 3d scene stylization. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp 20,712–20,721
- Zhenxing M, Xu D (2022) Switch-nerf: Learning scene decomposition with mixture of experts for large-scale neural radiance fields. In: *Int. Conf. Learn. Represent.*
- Zhou S, Chang H, Jiang S, Fan Z, Zhu Z, Xu D, Chari P, You S, Wang Z, Kadambi A (2024) Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp 21,676–21,685
- Zhu S, Zhang R, Zhou L, Shen T, Fang T, Tan P, Quan L (2018) Very large-scale global sfm by distributed motion averaging. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 4568–4577



# — Supplementary Material —

## FPGS



arXiv:2503.09635v1 [cs.GR] 11 Mar 2025

### APPENDIX A EXPERIMENTAL DETAILS

#### .1 Multi-view consistency comparison

By following previous works (Huang et al, 2022; Liu et al, 2023; Chen et al, 2024; Hedlin et al, 2023) we use multi-view error via image warping (see Sec. VI-B) as a metric for quantifying multi-view consistency of stylized scenes. With this metric, the rendered original scenes should be same for a fair comparison, since multi-view error of the stylized scene is proportional to multi-view error of the original scene. However, the original scenes of different methods cannot be same and multi-view errors of the original scenes depend on the capabilities for view-dependent color of comparing methods using different radiance field modelings.

To minimize the multi-view error difference of original scenes rendered from the comparing methods, we disable view-dependent color of each method. For the NeRF-based (Mildenhall et al, 2020) methods, UPST-NeRF (Chen et al, 2024) and FPRF (Kim et al, 2024), we train and render the scene with a single dummy view direction, which ensures view-independent radiance field reconstruction, *i.e.*, Lambertian model. For the 3D Gaussian-based (Kerbl et al, 2023) method, FPGS, we train and render the scene using spherical harmonics of degree 0, which can only represent view-independent radiance field.

We render  $K$  images  $\{\mathbf{I}_i\}_{i=1,2,\dots,K}$  with sampled sequential viewpoints from a continuous camera trajectory and select nearby image pairs with short and long baseline settings to compute warped image errors. We predict optical flows and binary masks with the rendered original image pairs using RAFT (Teed and Deng, 2020), and stylize the images with 20 style images from the PST dataset (Luan et al, 2017) for evaluation.

**Large-scale scene.** We compare our methods with two competing 2D PST methods (Wu et al, 2022; Chiu and Gurari, 2022) on the 4 blocks from the San Fran Cisco Mission Bay dataset (Tancik et al, 2022). We sample 80 image pairs from 4 camera trajectories for each block, which comprise  $(\mathbf{I}_i, \mathbf{I}_{i+1})$  as short baseline pairs and  $(\mathbf{I}_i, \mathbf{I}_{i+3})$  as long baseline pairs. To compare with 2D methods, we first render the original scenes with FPGS and stylize the rendered images with 2D PST methods.

**Small-scale scene.** We evaluate our methods with two competing 3D feed-forward methods (Liu et al, 2023; Chen et al, 2024) on the 8 scenes from the LLFF dataset (Mildenhall

et al, 2019). We sample 24 image pairs from a camera trajectory for each scene, which comprise  $(\mathbf{I}_i, \mathbf{I}_{i+1})$  as short baseline pairs and  $(\mathbf{I}_i, \mathbf{I}_{i+5})$  as long baseline pairs.

#### .2 Running time comparison

We compare training time and rendering time of our methods with feed-forward 3D style transfer methods, on the LLFF dataset (Mildenhall et al, 2019) (see Tab. IV). To train StyleRF (Liu et al, 2023) and UPST-NeRF (Chen et al, 2024), we follow the official implementation. We train FPRF and FPGS with 30,000 iterations for each scene, and train the semantic autoencoder of FPGS for 5 epochs with the training images. For a  $1008 \times 756$  resolution scene, we train and render comparing methods on a single NVIDIA RTX A6000 gpu, and obtain the average running time of all 8 scenes.

#### .3 Scribble-based style transfer

Our methods can stylize the 3D scene with a rendered image and scribbles on the image, as shown in Fig. 9. The rendered image holds strong semantic correspondence with the projected region from the 3D scene. This strong semantic correspondence enables transferring of scribbled colors to the matched region in the 3D scene. We first compute semantic correspondence between the rendered image and the scene semantic field, and transfer the style of the scribbled image to 3D scene with the obtained semantic correspondence.

For that, we compose a modified style dictionary  $\mathcal{D}$  with the clustered semantic features  $\mathbf{f}_{\text{DINO}}(\mathbf{I}_{\text{ren}}^j)$  extracted from the rendered image  $\mathbf{I}_{\text{ren}}$  and the style codes from the scribbled image  $\mathbf{I}_{\text{scr}}$ , as  $\mathcal{D} = \{\mathbf{f}_{\text{DINO}}(\mathbf{I}_{\text{ren}}^j) : (\mu_{\text{scr}}^j, \sigma_{\text{scr}}^j)\}_{j=1,\dots,M}$ . A centroid of DINO feature  $\mathbf{f}_{\text{DINO}}(\mathbf{I}_{\text{ren}}^j)$  is averaged from each cluster. A style code  $(\mu_{\text{scr}}^j, \sigma_{\text{scr}}^j)$  is a pair of the mean and standard deviation of the VGG features extracted from each cluster of  $\mathbf{I}_{\text{scr}}$ , which have the same clustered region with  $\mathbf{I}_{\text{ren}}$ . Then we compute semantic correspondence and a semantic-weighted style code using Eq. (13) with the modified dictionary  $\mathcal{D}$ . As a result, we obtain the semantic-weighted style code  $(\mathbf{M}_w, \Sigma_w)$  being assigned to 3D Gaussians, which contains semantically corresponding style codes from the scribbled image  $\mathbf{I}_{\text{scr}}$ . Then we can obtain the stylized result using Eq. (14), which transfer the style of scribbled image to the scene, by considering semantic correspondence between the rendered image and the 3D scene.

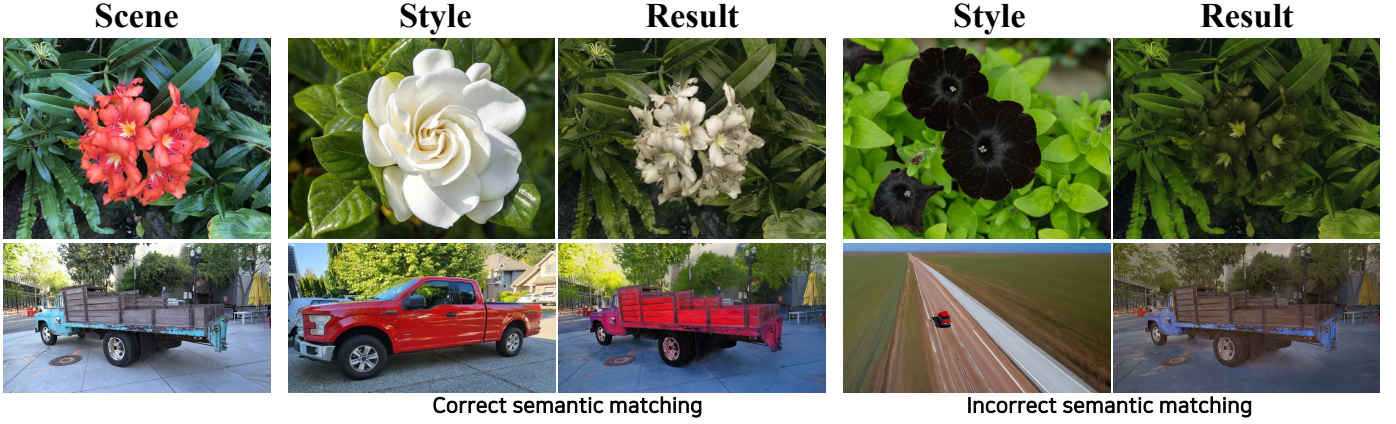


Fig. S1. Stylization results with correct semantic matchings and incorrect semantic matchings. [Left] Original 3D scene. [Mid] Stylization results with correct semantic matchings. [Right] Failure cases of our methods with incorrect semantic matchings in extreme cases.

## APPENDIX B IMPLEMENTATION DETAILS

**Generalizable pre-trained MLP color decoder.** We propose a pre-trained MLP color decoder which enables feed-forward style transfer with new arbitrary styles (see Sec. IV-A). We compose the decoder with a sequence of 2 linear layers whose output channels are 128 and 3, followed by a sigmoid function to output an RGB value. The color decoder is trained on the MS COCO (Lin et al, 2014) dataset and the wikiart (Nichol, 2016) dataset as a content and a style image dataset. We train the network for 50,000 iterations with the learning rate 0.0001.

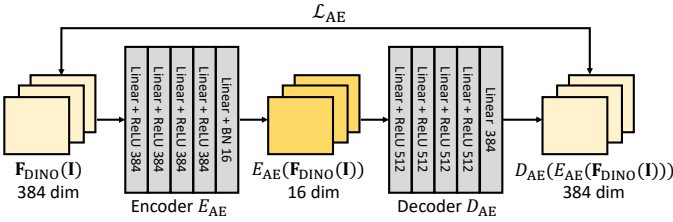


Fig. S2. Training pipeline of the semantic feature autoencoder.  $F_{\text{DINO}}(I)$  denotes the feature maps extracted from the training images  $I$  with DINO and  $\mathcal{L}_{\text{AE}}$  denotes the reconstruction loss (see Eq. (12)).

**Semantic feature autoencoder.** We employ a scene-specific semantic feature autoencoder compressing high-dimensional semantic features to low-dimensional features, which enables semantic feature field represented by 3D Gaussians (see Sec. V). For the DINO (Caron et al, 2021) features whose channel size is 384, we set the channel of low-dimensional features to 16. Figure S2 shows the detailed architecture and the training process of the semantic feature autoencoder. We train the autoencoder for 5 epochs with semantic features extracted from training images of each scene. We set the learning rate to 0.0001.

## APPENDIX C ADDITIONAL RESULTS

**Failure cases.** As mentioned in Sec. VII, our semantic matching performance is bounded by the semantic image

encoder, DINO (Caron et al, 2021). Figure S1 shows failure cases of our method, which are related to the capability of DINO. The first row shows a success case and a failure case due to an out of distribution sample, a black flower. DINO cannot correctly predict that the black flower in the reference image and the red flower in the original scene are in the same category. *i.e.*, have semantic correspondence. The second row also shows that DINO works well when the truck in the reference image is big enough, however, fails to match the extremely small truck on the road and the truck in the original scene.

**Qualitative results.** Figure S3 shows comparisons with UPST-NeRF (Chen et al, 2024) on the LLFF dataset (Mildenhall et al, 2019). Figure S4 shows qualitative results on the San Fran Cisco Mission Bay dataset (Tancik et al, 2022), the Tank and Temples dataset (Knapitsch et al, 2017), and the Mip-NeRF 360 dataset (Barron et al, 2022), which contain unbounded scenes. Figure S5 shows qualitative results on the DyNeRF dataset (Li et al, 2022) and the KITTI-360 dataset (Liao et al, 2022) dataset, which are 4D scene datasets.

## REFERENCES

- Barron JT, Mildenhall B, Verbin D, Srinivasan PP, Hedman P (2022) Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 5470–5479
- Caron M, Touvron H, Misra I, Jégou H, Mairal J, Bojanowski P, Joulin A (2021) Emerging properties in self-supervised vision transformers. In: Int. Conf. Comput. Vis.
- Chen Y, Yuan Q, Li Z, Liu Y, Wang W, Xie C, Wen X, Yu Q (2024) Upst-nerf: Universal photorealistic style transfer of neural radiance fields for 3d scene. IEEE Transactions on Visualization and Computer Graphics
- Chiu TY, Gurari D (2022) Photowct2: Compact autoencoder for photorealistic style transfer resulting from blockwise training and skip connections of high-frequency residuals. In: IEEE Winter Conf. on Applications of Computer Vision (WACV)
- Hedlin E, Sharma G, Mahajan S, Isack H, Kar A, Tagliasacchi A, Yi KM (2023) Unsupervised semantic correspondence using stable diffusion. arXiv preprint arXiv:230515581



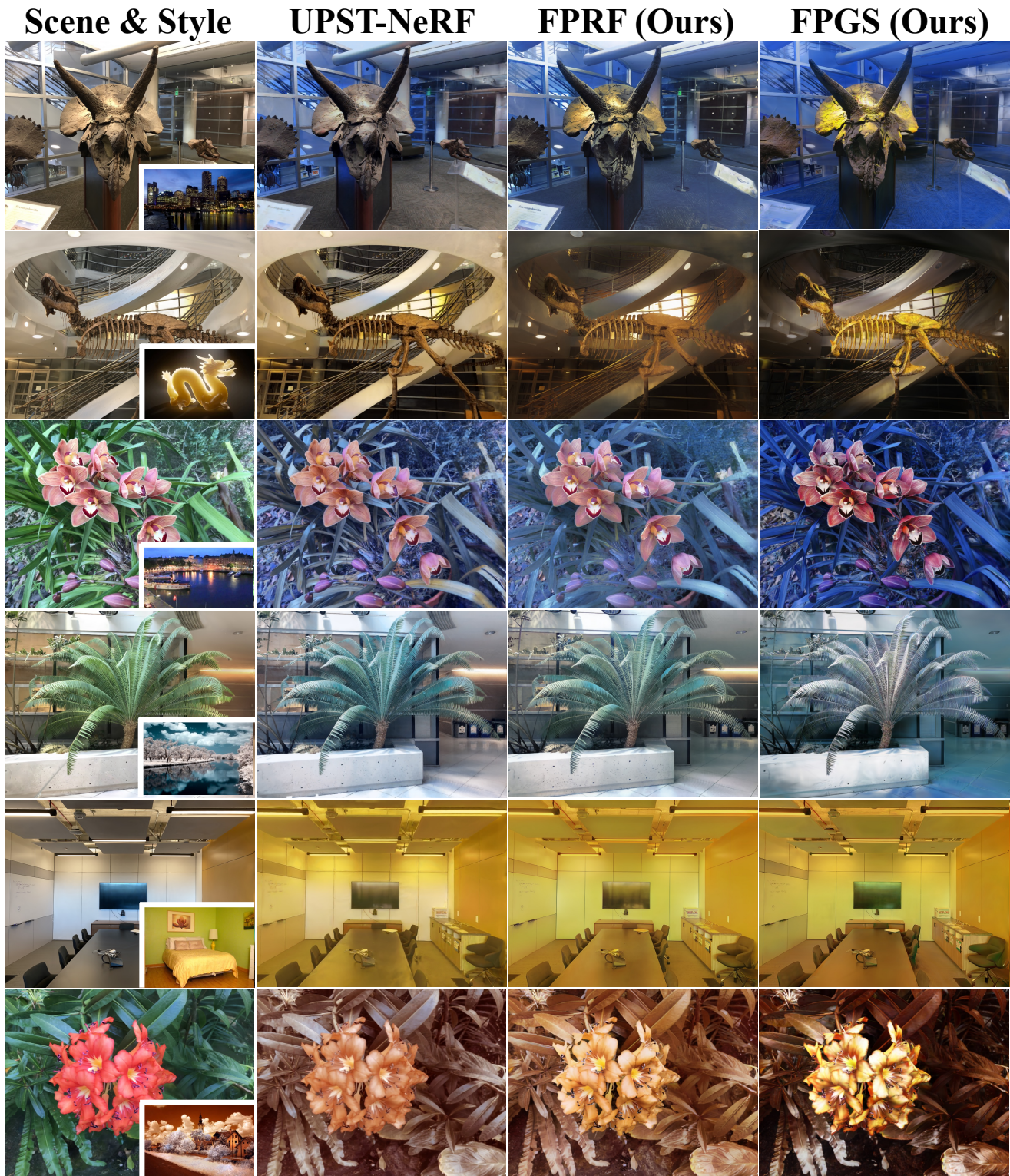


Fig. S3. Additional qualitative results on LLFF dataset (Mildenhall et al, 2019). Compared to the UPST-NeRF (Chen et al, 2024), our methods accurately reflect the diverse color of the reference image.



## Scene & Style

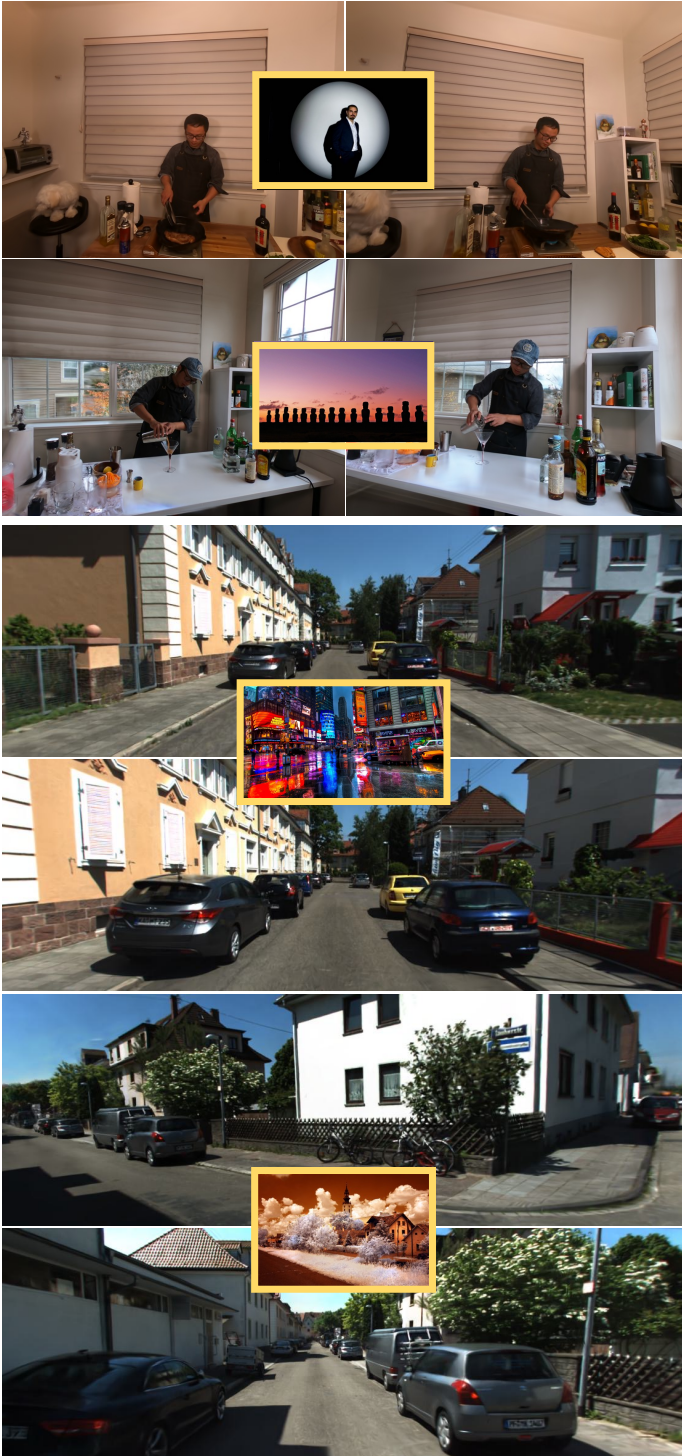
## Stylized scene



Fig. S4. **Additional qualitative results on unbounded scenes.** Style transfer results on the San Francisco Mission Bay dataset (Tancik et al, 2022) (top 2), on the Tank and Temples dataset (Knapitsch et al, 2017) (mid 2), and on the Mip-NeRF 360 dataset (Barron et al, 2022) (bottom 2).



## Scene & Style



## Stylized scene



Fig. S5. **Additional qualitative results on 4D scenes** Style transfer results on the DyNeRF dataset (Li et al, 2022) (top) and on the KITTI-360 dataset (Liao et al, 2022) (bottom).

- Huang YH, He Y, Yuan YJ, Lai YK, Gao L (2022) 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, pp 18,342–18,352
- Kerbl B, Kopanas G, Leimkühler T, Drettakis G (2023) 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics* 42(4), URL <https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/>
- Kim G, Youwang K, Oh TH (2024) FPRF: Feed-forward photorealistic style transfer of large-scale 3D neural radiance fields. In: AAAI
- Knapitsch A, Park J, Zhou QY, Koltun V (2017) Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)* 36(4):1–13
- Li T, Slavcheva M, Zollhoefer M, Green S, Lassner C, Kim C, Schmidt T, Lovegrove S, Goesele M, Newcombe R, et al (2022) Neural 3d video synthesis from multi-view video. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 5521–5531
- Liao Y, Xie J, Geiger A (2022) Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45(3):3292–3310
- Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014) Microsoft coco: Common objects in context. In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V 13*, Springer, pp 740–755
- Liu K, Zhan F, Chen Y, Zhang J, Yu Y, El Saddik A, Lu S, Xing EP (2023) Stylerf: Zero-shot 3d style transfer of neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 8338–8348
- Luan F, Paris S, Shechtman E, Bala K (2017) Deep photo style transfer. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4990–4998
- Mildenhall B, Srinivasan PP, Ortiz-Cayon R, Kalantari NK, Ramamoorthi R, Ng R, Kar A (2019) Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)* 38(4):1–14
- Mildenhall B, Srinivasan PP, Tancik M, Barron JT, Ramamoorthi R, Ng R (2020) Nerf: Representing scenes as neural radiance fields for view synthesis. In: *Eur. Conf. Comput. Vis.*
- Nichol K (2016) Painter by numbers, wikiart, 2016. URL <https://www.kaggle.com/c/painter-by-numbers/overview>
- Tancik M, Casser V, Yan X, Pradhan S, Mildenhall B, Srinivasan P, Barron JT, Kretzschmar H (2022) Block-NeRF: Scalable large scene neural view synthesis. In: *IEEE Conf. Comput. Vis. Pattern Recog.*
- Teed Z, Deng J (2020) Raft: Recurrent all-pairs field transforms for optical flow. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, Springer, pp 402–419
- Wu Z, Zhu Z, Du J, Bai X (2022) Ccpl: contrastive coherence preserving loss for versatile style transfer. In: *European Conference on Computer Vision*, Springer, pp 189–206