A^{3}GS: Arbitrary Artistic Style into Arbitrary 3D Gaussian Splatting

Supplementary Material

1. Video Demonstration

A video demonstration of the proposed approach is provided as supplementary material, providing a visual presentation of the methods and results discussed in this article.

2. Training Environment

The experiments were conducted on a system equipped with NVIDIA RTX A6000 GPU with 48GB of memory. Both the autoencoder and stylizer models were trained using a dual-GPU setup. For the training process, the following hyperparameters were employed: a learning rate of 5e-5, a batch size of 8, and the optimizer used was Adam. The autoencoder was trained for 30 epochs, while the stylizer was trained for 10 epochs. The training time for completing the entire process was approximately 22 hours.

3. Details of 3D Graph Convolution Network

3D Graph Convolution Network (3D-GCN) [6] is designed to extract local 3D features from point clouds. For any instance of point clouds $P = \{p_n\}_{n=1}^{N}$ which contains N points $p_n \in \mathbb{R}^3$, we define the receptive field of point p_n with size M as:

$$R_n^M = \{p_n, p_m \mid \forall p_m \in \mathcal{N}(p_n, M)\} \tag{1}$$

where $\mathcal{N}(p_n, M)$ represents M nearest neighbors for point p_n based on the spatial distance. It is important to note that each point is associated with a D-dimensional feature vector, denoted by $f(p_n)$. Similar to convolution kernels that extract features on grid structures in 2D CNN, a novel convolution kernel with learnable shape and weights is introduced for unstructured data like point clouds, which is denoted as:

$$K^S = \{k_C, k_s\}_{s=1}^{S}, \quad k_s \in \mathbb{R}^3 \tag{2}$$

Specifically, $k_C = (0, 0, 0)$ is the center of the kernel, trainable parameters k_1 to k_S denote the associated supports and each kernel point k is associated with a D-dimensional weight vector $w(k) \in \mathbb{R}^D$. With the above definitions, the convolution operation is calculated as:

$$f'(p_n) = \langle f(p_n), w(k_C)\rangle + \sum_{s=1}^{S} \max_{m\in(1,M)} \{\mathcal{S}(p_m, k_s)\} \tag{3}$$

$$\mathcal{S}(p_m, k_s) = \langle f(p_m), w(k_s)\rangle \frac{\langle d_{m,n}, k_s\rangle}{\|d_{m,n}\|\|k_s\|} \tag{4}$$

where $\langle\cdot\rangle$ denotes the inner product operation and $d_{m,n}$ denotes the directional vector from p_n to p_m.

Furthermore, a pooling operation is introduced to allow the model to capture more complex patterns at higher levels while reducing computational cost. This pooling operation uses max pooling in the channel to aggregate the features of each point within its receptive field, followed by downsampling of the point cloud at a sampling rate r.

4. Loss function for Stylization Stage

In this section, we introduce the losses used during the stylization stage. The content loss $\mathcal{L}_c$ is defined as,

$$\mathcal{L}_c = \|\phi_i(c_i) - \phi_i(c_o)\|_2, \tag{5}$$

where a is the input content image, and b is the generated stylized image. In addition, each ϕ_i denotes a layer in VGG-19 [8] used to compute the style loss. In our experiments we use $relu1_1$, $relu2_1$, $relu3_1$, $relu4_1$ layers with equal weights.

Furthermore, As our stylizer transfers the mean and standard deviation of the style features, our style loss is defined to match these statistics. Specifically, the style loss $\mathcal{L}_s$ is defined as,

$$\mathcal{L}_s = \sum_{i=1}^{L} \|\mu(\phi_i(c_o)) - \mu(\phi_i(s))\|_2 + \sum_{i=1}^{L} \|\sigma(\phi_i(c_o)) - \sigma(\phi_i(s))\|_2. \tag{6}$$

This style loss has also been explored by [3, 5]

5. Details of User Study

To evaluate the perceptual quality of our style transfer method, we conducted a subjective user study. Each participant was presented with the original 3D scene, the style, and stylized results from multiple methods displayed side by side in a random order. Participants rated each result on three criterias: style consistency, content preservation, and visual naturalness, with scores ranging from 0 to 5. Specifically, style consistency measures how closely the result matches the features of the target style. Content preservation evaluates whether the structure and details of the original scene are retained. Visual naturalness evaluates the absence of artifacts and overall perceptual realism. The final scores are averaged among all participants.

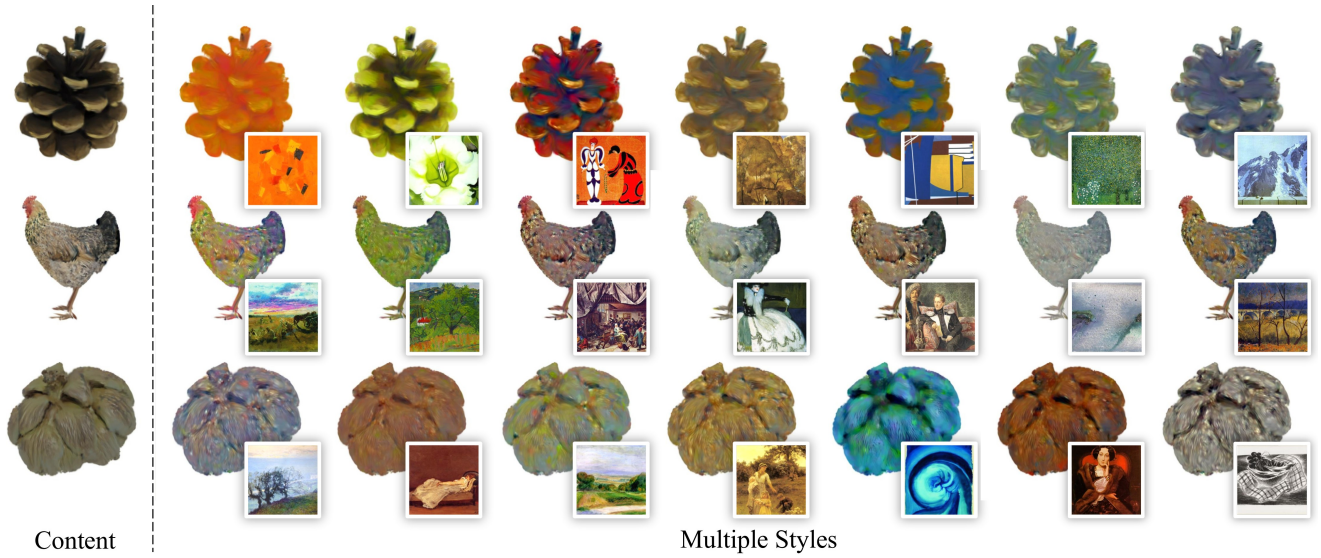

Figure 1. **Results of Objects.** Style transfer results over dataset constructed from Objaverse [1, 2] using TriplaneGaussian [11].

Figure 2. **Additional comparison results.** Comparison with state-of-the-art 3DGS stylization method StyleSplat [4].

6. Additional Experiments

6.1. Object-level Results

To further demonstrate the versatility of our method, we present additional object-level style transfer results, as shown in Figure 1. The test objects used in this evaluation are selected from the test split of our generated dataset. Results show that even for objects with complex textures, our model is capable of transferring brushstroke-level styles from the reference images. This demonstrates the model's ability to capture fine-grained style patterns while preserving the structural integrity of the target object.

6.2. Additional Comparison Results

Comparisons with StyleSplat. We have added more qualitative results in Figure 2. StyleSplat requires conducting additional training optimization for each new style scene, which takes 5 minutes. The additional optimization enhances the visual quality of StyleSplat's results to some extent, but it tends to *"smooth out"* fine geometric details. In contrast, our approach only requires a zero-shot forward inference for each new scene and style in 10s and achieves more faithful style transfer, exhibiting clearer brushstroke patterns and better color consistency with the reference styles, while preserving the content structure. Notably, our approach demonstrates stronger adaptability across diverse style domains, especially in capturing the stylistic nuances in Style2.

Comparisons with NeRF. We also present the qualitative comparison results with the NeRF-based method, StyleRF [7], as shown in Figure 3. Compared to the NeRF-based ap-

proach, our method not only achieves real-time online style transfer but also produces clearer and more distinct stylization results.

6.3. Larger Content Loss

In the main text, we evaluated different feature extraction methods within the autoencoder architecture to assess their effectiveness, including MLP and a simple GCN without learnable kernel shapes. Results initially indicated that the 3D-GCN better preserves 3D structures and generates more convincing stylization compared to the other two methods. However, this conclusion may be affected by potential confounding factors, as the degraded structural quality in baseline methods could potentially be mitigated by increasing the content loss during training. To address this concern, we conducted additional experiments (see Figure 4) with a higher content loss to examine whether structural preservation holds across methods. The results show that even with an increased content loss, the alternative feature extraction methods still fail to effectively preserve scene-level structural details. In contrast, our method consistently maintains superior 3D structural integrity while producing visually compelling styles.

6.4. Exploring Per-layer AdaIN

Deep CNN networks are capable of extracting richer semantic content and global features from images at deeper layers, as observed in previous works[9]. Inspired by this, we apply AdaIN at the bottleneck and then propagate the style feature through the GCN, with the aim of achieving more global consistent stylization. However, in other types of tasks[10], performing AdaIN at each layer has been shown to produce better results. We added experiments on applying AdaIN at each layer to further investigate its benefits.

Based on the original network architecture, we augment each convolutional layer with the stylizer module, which includes both AdaIN and MLPs for feature space alignment. Specifically, we introduce three AdaIN stylizers with shared weights, each designed to align features of different dimensions with VGG-style features of corresponding dimensions, as shown in Figure 5. The results presented in Figure 6 indicate that the network does not benefit from per-layer AdaIN. On the contrary, it leads to significant loss of scene details and incorrect style matching. Moreover, the introduction of per-layer AdaIN substantially introduces considerable parameter overhead and hinders training convergence.

6.5. Additional Ablation Results

Additionally, we provide further ablation study results on feature extraction methods, as shown in Figure 7.

References

[1] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. *arXiv preprint arXiv:2212.08051*, 2022. 2

[2] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, Eli VanderBilt, Aniruddha Kembhavi, Carl Vondrick, Georgia Gkioxari, Kiana Ehsani, Ludwig Schmidt, and Ali Farhadi. Objaverse-xl: A universe of 10m+ 3d objects. *arXiv preprint arXiv:2307.05663*, 2023. 2

[3] Xun Huang and Serge J. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. *CoRR*, abs/1703.06868, 2017. 1

[4] Sahil Jain, Avik Kuthiala, Prabhdeep Singh Sethi, and Prakanshul Saxena. Stylesplat: 3d object style transfer with gaussian splatting, 2024. 2

[5] Yanghao Li, Naiyan Wang, Jiaying Liu, and Xiaodi Hou. Demystifying neural style transfer. *arXiv preprint arXiv:1701.01036*, 2017. 1

[6] Zhi-Hao Lin, Sheng-Yu Huang, and Yu-Chiang Frank Wang. Convolution in the cloud: Learning deformable kernels in 3d graph convolution networks for point cloud analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1

[7] Kunhao Liu, Fangneng Zhan, Yiwen Chen, Jiahui Zhang, Yingchen Yu, Abdulmotaleb El Saddik, Shijian Lu, and Eric Xing. Stylerf: Zero-shot 3d style transfer of neural radiance fields. 2023. 2

[8] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 1

[9] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013. 3

[10] Yunfan Zhang, Hao Wang, Guosheng Lin, Vun Chan Hua Nicholas, Zhiqi Shen, and Chunyan Miao. Starnet: Style-aware 3d point cloud generation, 2023. 3

[11] Zi-Xin Zou, Zhipeng Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Yan-Pei Cao, and Song-Hai Zhang. Triplane meets gaussian splatting: Fast and generalizable single-view 3d reconstruction with transformers. *arXiv preprint arXiv:2312.09147*, 2023. 2

Figure 3. **Comparison results with NeRF based work.**

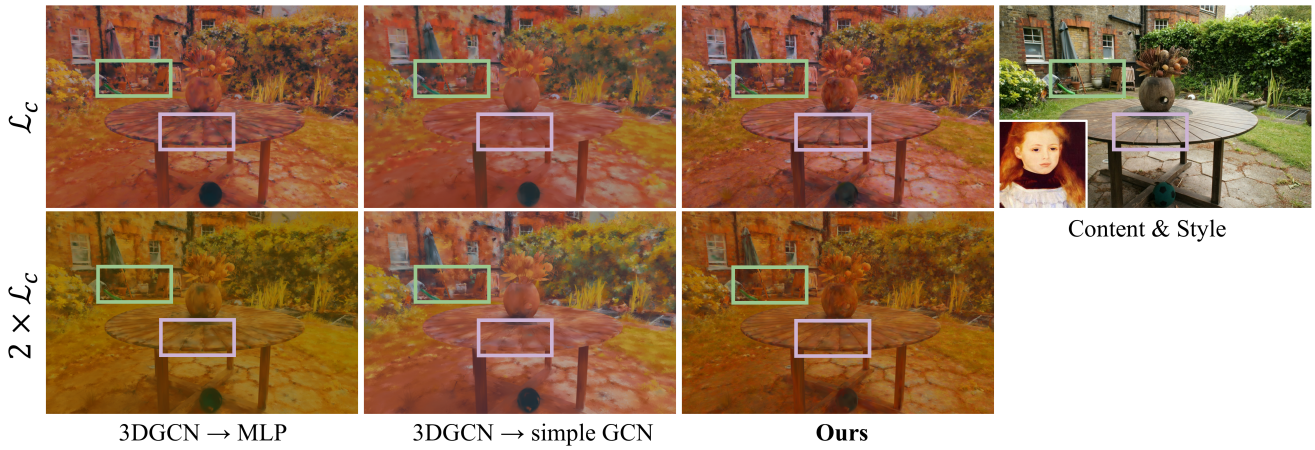

Figure 4. **Larger Content Loss.** We double the content loss during training for three different network architectures. The results show that simply increasing the content loss cannot eliminate the loss of scene details caused by MLP or simple GCN.

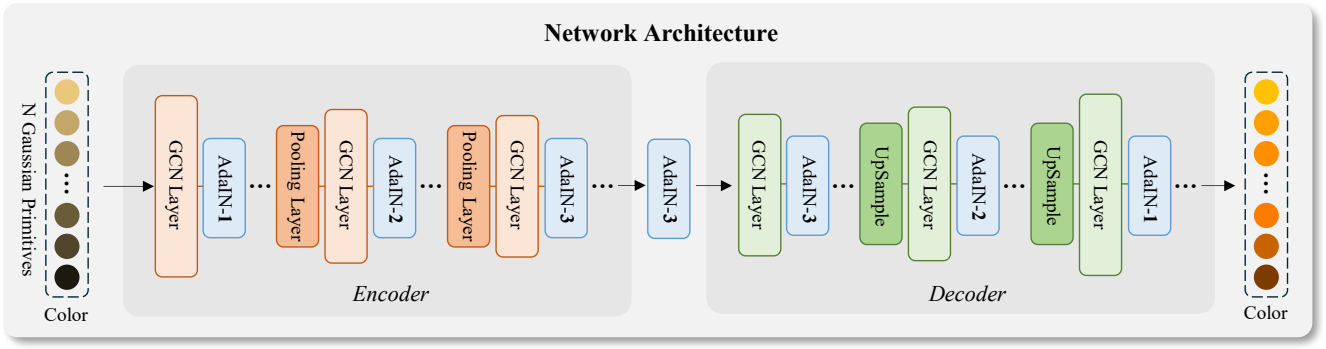

Figure 5. **Network Architecture with Per-layer AdaIN.**

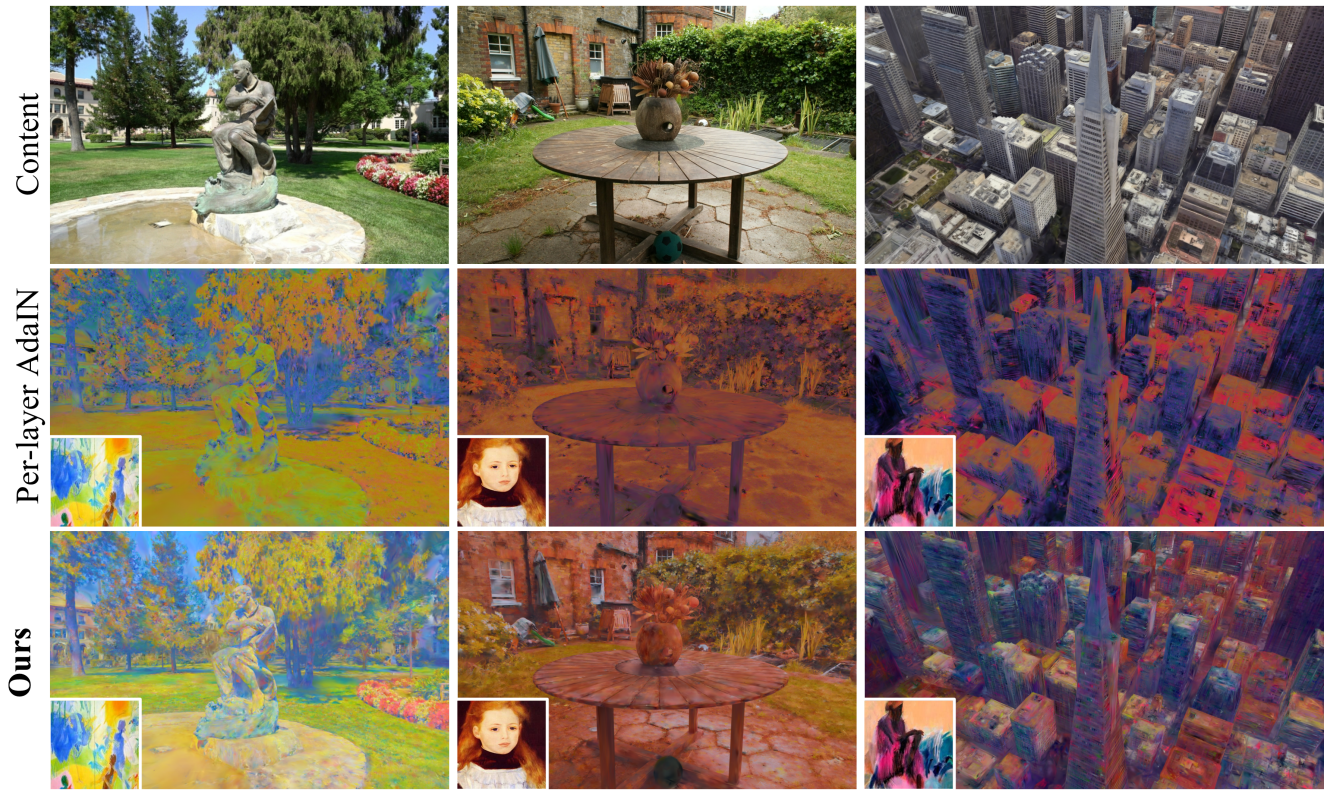

Figure 6. **Results of Per-layer AdaIN.**

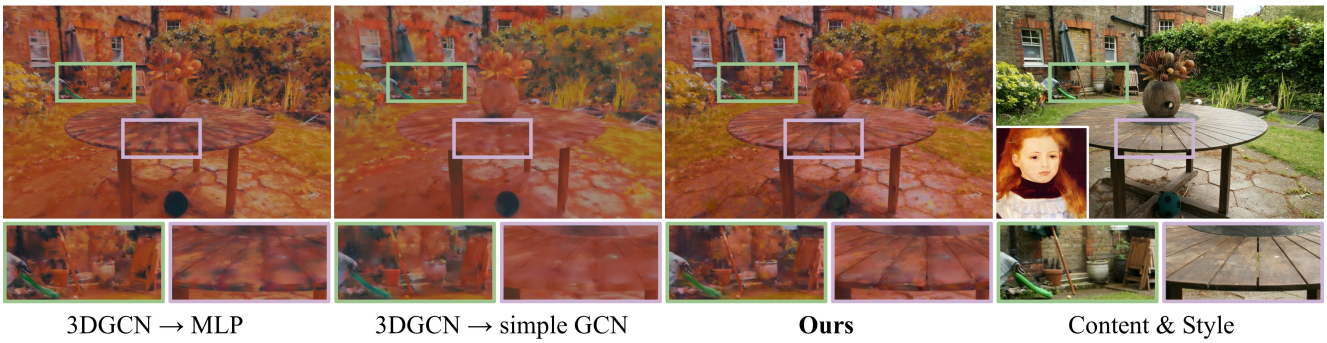

Figure 7. **Additional Ablation Results** about the feature extraction methods in autoencoder structure.