3D Scene Painting via Semantic Image Synthesis
— *Supplementary Material* —

Jaebong Jeong Janghun Jo Sunghyun Cho Jaesik Park
POSTECH GSAI & CSE
{jbjeong, jhjo432, s.cho, jaesik.park}@postech.ac.kr

This supplementary material provides more qualitative results, details of the network architecture, and some failure cases that could not be included in the main paper due to the limit of space. Especially, we provide a video that shows the visual consistency and the advantages of our scene painting framework. We also provide the code, data, and instructions needed to reproduce the experimental results.

A. Qualitative Results

A.1. Video

The supplementary video presents a comparison between OASIS [5] and our method. The results of OASIS in the video are generated by applying OASIS to semantic label maps in a frame-wise manner. The results of OASIS show temporal inconsistency, and distorted structures as it does not consider underlying 3D structures. Our results show consistent and geometrically accurately synthesized frames.

A.2. Scene Style Manipulation

As mentioned in the main paper, our approach readily controls the style of a scene by changing the style vector $\boldsymbol{z}$. We can change the style in the test time, so it does not require the re-training of the network. In Figure 1, Figure 2, and Figure 3, we show examples of scene style manipulation of a living room, an office, and a kitchen scene.

We also compare the colored scene using output images from our scene painting network with a colored scene using pseudo images from OASIS [5]. In Figure 3, the colored scenes using pseudo images show messy textures.

A.3. Scene Editing

In Figure 4, Figure 5, and Figure 6, we present an application of our approach to scene editing. Note that our approach assigns texture colors for each mesh in a 3D scene, and it is just the same with the ordinal 3D scene editing case for the manipulation. Compared with neural rendering techniques, this feature allows users to modify scene color and configuration more easily. The example images were rendered from the edited scene with textured meshes generated by our method.

B. Network Architecture

In Table 1 in the main paper, we compare the performances of different network architectures for the scene painting network: a MLP and a MLP+CNN. In this section, we present their detailed architectures.

In Figure 7, we show the network architecture of the MLP. From a given 3D scene with semantically labeled objects, we render a 2D semantic label map and a 3D world coordinate map, which is normalized into $[-1, 1]$. We compute the positional encoding from the coordinate map. Then, we concatenate the positional encoded coordinate map and the label map to generate an input feature map for the generator. A style vector is a 64-dimensional vector sampled from the standard normal distribution. We feed the 64-dimensional style vector into the style mapping network $\mathcal{W}$ as done in StyleGANv2 [2] and get a 256-dimensional transformed style vector. The MLP generator is implemented with 1×1 convolution layers as it is equivalent to applying fully connected layers to each pixel independently. ModConv1x1 is a 1×1 convolution layer with weight modulation using a 256-dimensional transformed style vector such as StyleGANv2 [2]. We use the leaky ReLU [3] activation function. We use seven ModConv1x1 layers. Conv1x1 is a vanilla 1×1 convolution layer. After the last convolution layer, we use a hyperbolic tangent function to generate an RGB image.

Previous methods based on implicit representations [1,4] adopt CNNs to improve their performance. In this study, we also examine whether adopting a CNN can improve the generation quality of our framework. In Figure 8, we show the network architecture of the MLP+CNN. We add a Conv3x3 block to the MLP architecture, which consist of Conv3x3 ResBlocks and Affine layers. A Conv$N$xN ResBlock consists of two conv layers and a Leaky ReLU layer with a skip conneciton. An Affine layer computes Affine parame-

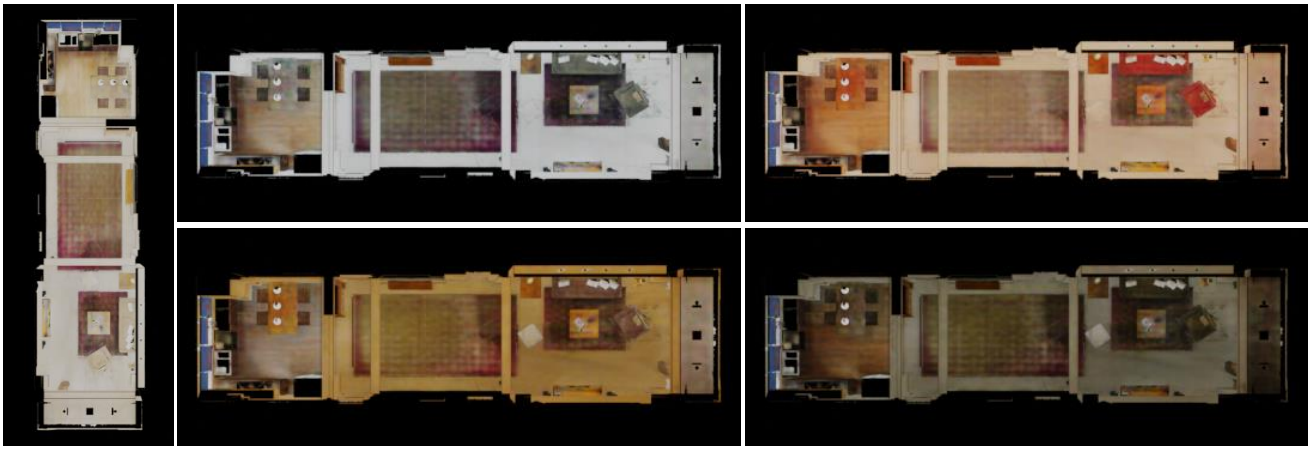

Figure 1. Living room scene examples of style manipulation using our scene painting network.

Figure 2. Office scene examples of style manipulation using our scene painting network.

Figure 3. Kitchen scene examples of style manipulation using our scene painting network. The first row shows colored scenes using generated images from our scene painting network. The second row shows colored scenes using pseudo images from OASIS [5].

ters from a transformed style vector, and applies the parameters to an input feature map.

C. Failure Cases

Our scene painting network generates a color value of a 3D coordinate on a mesh surface. This approach introduces a few limitations. First, our method cannot handle transpar-

Figure 4. Scene editing example of a bedroom scene. We paint the scene using our scene painting network. We add some cups, some pillows, and change the color of the floor and the table.

Figure 5. Scene editing example of a living room scene. We paint the scene using our scene painting network. We add some stools and a chair, and resize a stool.

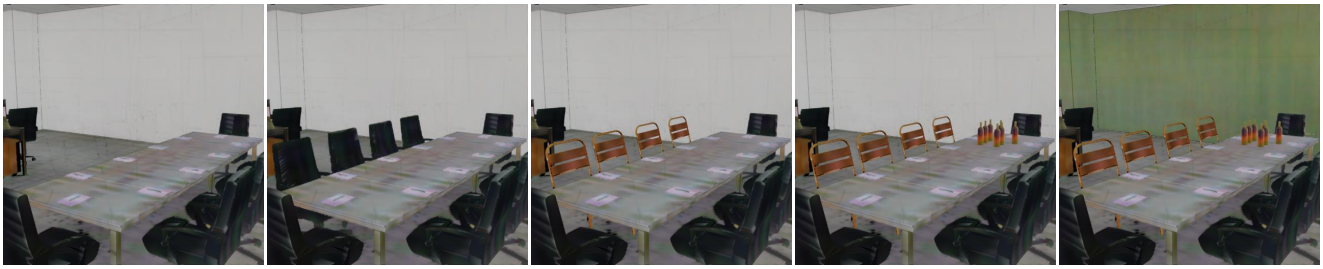

Figure 6. Scene editing example of a office scene. We paint the scene using our scene painting network. We add some chairs, replace the chairs, put some bottles, and change the color of the wall.

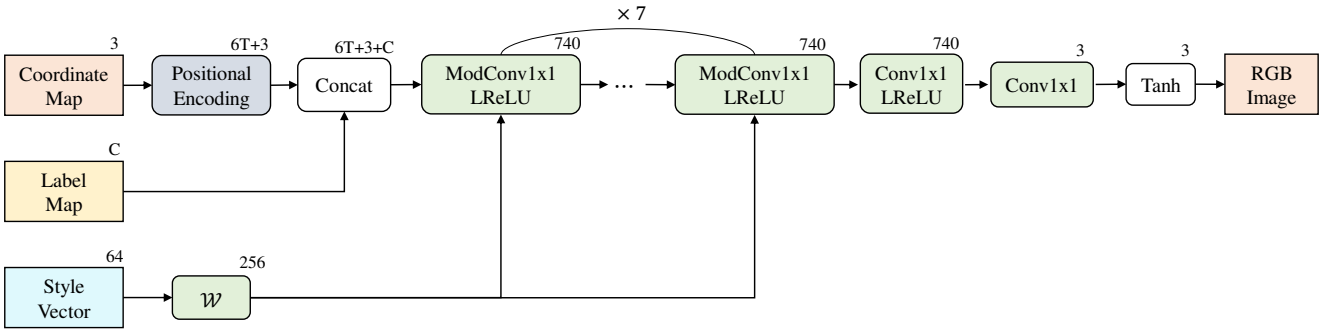

Figure 7. A flowchart of the MLP architecture. The inputs are a coordinate map, a label map, and a style vector. The number above each box shows the number of output channels. Concat means concatenation of the input tensors along the channel dimension. ModConv1x1 is a 1×1 convolution layer with weight modulation using the 256-dimensional transformed style vector such as StyleGANv2 [2]. LReLU is a leaky ReLU function [3]. Conv1x1 is a vanilla 1×1 convolution layer. Tanh is a hyperbolic tangent function. The output is a generated RGB image.

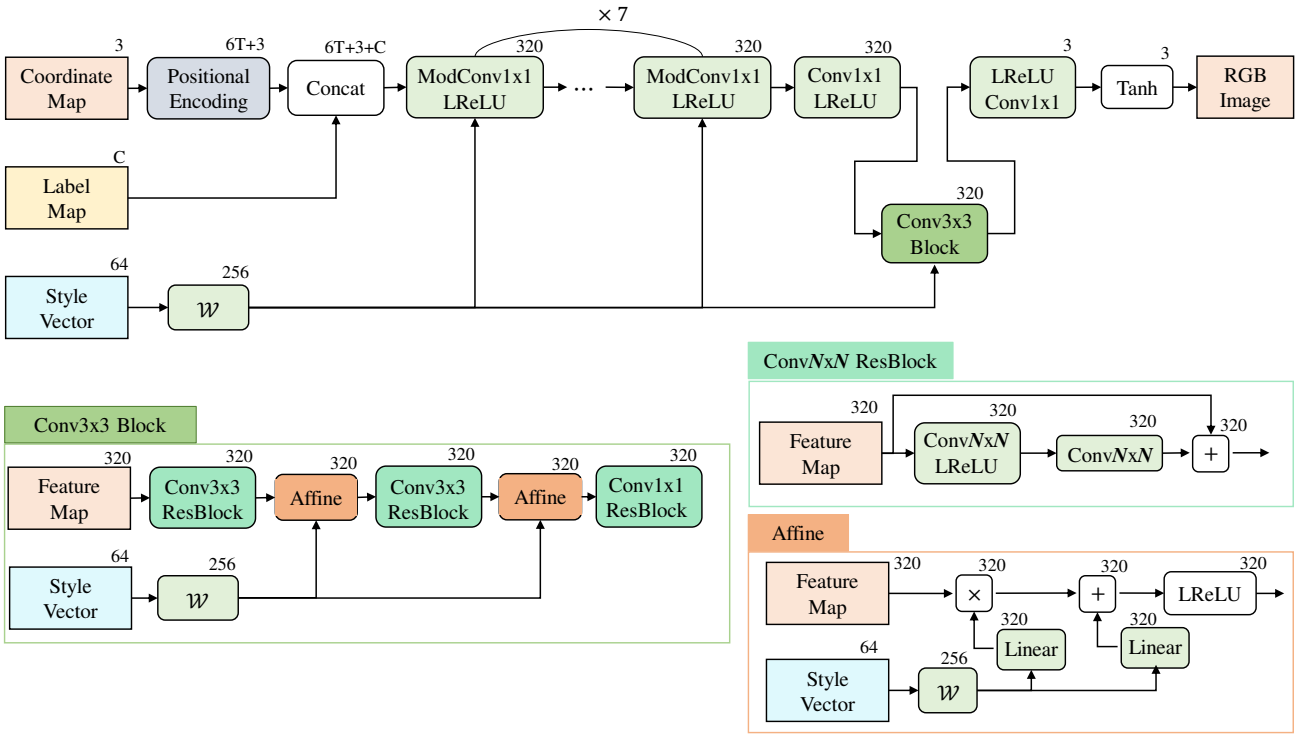

Figure 8. A flowchart of the MLP+CNN architecture. We add a Conv3x3 Block to the MLP architecture. The inputs are a coordinate map, a label map, and a style vector. The number above each box shows the number of output channels. Concat means concatenation of the input tensors along the channel dimension. ModConv1x1 is a 1×1 convolution layer with weight modulation using the 256-dimensional transformed style vector such as StyleGANv2 [2]. LReLU is a leaky ReLU function [3]. Conv1x1 is a vanilla 1×1 convolution layer. Tanh is a hyperbolic tangent function. The output is a generated RGB image.

Figure 9. Failure case of transparent objects. The first row shows our results and the second row shows pseudo images from OASIS [5]. The images in the same row have different camera viewpoints.

ent objects such as windows but handles them as opaque objects, as shown in Figure 9. Second, our method cannot effectively handle view-dependent lighting effects such as specular lighting and reflection, as shown by the bed and the

Figure 10. Failure case of lighting. The first row shows our results and the second row shows pseudo images from OASIS [5]. The images in the same row have different camera viewpoints.

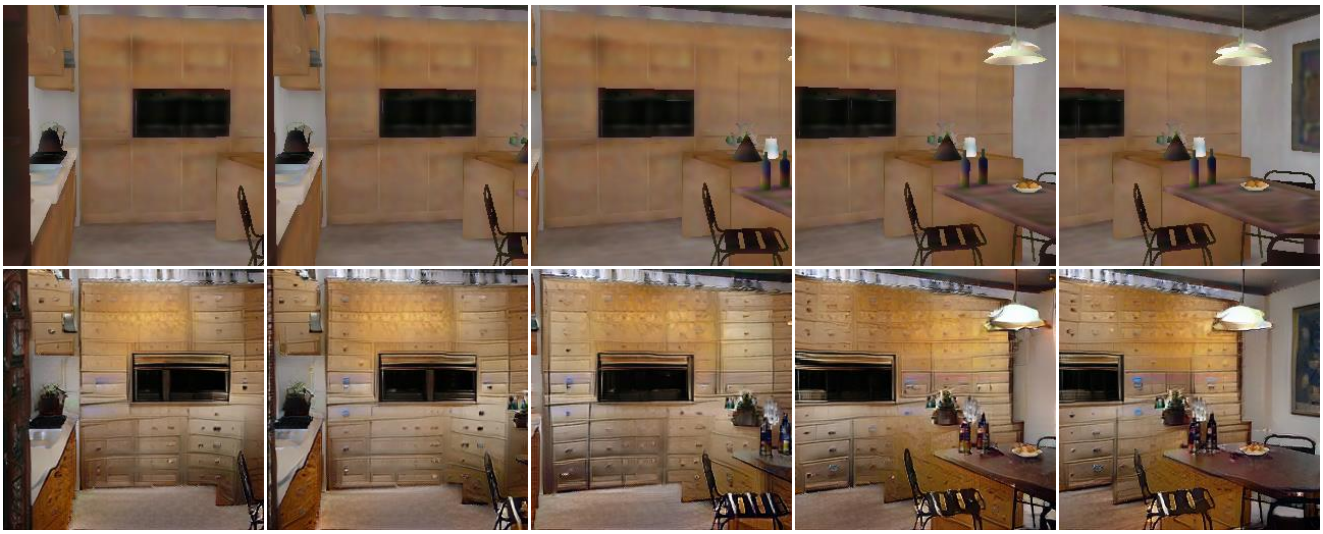

Figure 11. Failure case of blurred drawers. The first row shows our results and the second row shows pseudo images from OASIS [5]. The images in the same row have different camera viewpoints.

floor in Figure 10. Finally, our method aggregates pseudo images independently synthesized by a 2D semantic image synthesis method, and this may introduce blurry artifacts. In Figure 11 and Figure 12, while the drawers and paintings in the pseudo images synthesized by OASIS [5] have complex textures, the drawers and paintings in our results look blurry without textures.

D. Number of training frames

Table 1 show the effect of different numbers of training frames on the image quality. Fewer training frames indicate a smaller amount of supervision for each 3D coordinate. Some unobserved regions may not have supervision. This leads to quality degradation (1−4th rows of Table 1).

Table 1. The quantitative evaluation.

Method	# of frames	Time	mIoU (↑)	FID (↓)
Generated	50	2h	0.179	194.158
Generated	100	4h	0.195	185.081
Generated	400	18h	0.219	162.211
Generated	800	30h	**0.241**	**151.401**
Averaging	800	36m	0.214	180.436

Figure 12. Failure case of blurred paintings. The first row shows our results and the second row shows pseudo images from OASIS [5]. The images in the same row have different camera viewpoints.

References

[1] Zekun Hao, Arun Mallya, Serge Belongie, and Ming-Yu Liu. Gancraft: Unsupervised 3d neural rendering of minecraft worlds. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 14072–14082, October 2021. 1

[2] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8107–8116, 2020. 1, 3, 4

[3] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 2013. 1, 3, 4

[4] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11448–11459, 2021. 1

[5] Edgar Schönfeld, Vadim Sushko, Dan Zhang, Juergen Gall, Bernt Schiele, and Anna Khoreva. You only need adversarial supervision for semantic image synthesis. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. 1, 2, 4, 5, 6