3D Scene Painting via Semantic Image Synthesis

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

Abstract

We propose a novel approach to 3D scene painting using a configurable 3D scene layout. Our approach takes a 3D scene with semantic class labels as input and trains a 3D scene painting network that synthesizes color values for the input 3D scene. We exploit an off-the-shelf 2D semantic image synthesis method to teach the 3D painting network without explicit color supervision. Experiments show that our approach produces images with geometrically correct structures and supports scene manipulation, such as the change of viewpoint, object poses, and painting style. Our approach provides rich controllability to synthesized images in the aspect of 3D geometry.

1. Introduction

Creating realistic 3D scenes becomes crucial due to the increasing demands of unobserved content for virtual realism. However, it is regarded as a challenging problem because many components depend on human labor [5, 7, 10, 24] or manual capture of real scenes [45]. In particular, painting a scene is hard for a human, and it takes extra effort if we want to change the style after the creation. There are some attempts to focus on automatic 3D scene painting to resolve the issues. For instance, given a 3D scene and reference image, they find a texture from a texture database, which has a similar color distribution to the reference image [54]. However, making a large-scale texture set is a burden due to the variety of scene geometry.

We propose an automatic painting approach for 3D scene creation in this work. For 3D scene painting, our approach learns a 3D scene painting network that takes a 2D semantic map and 3D coordinate map as input and produces a 2D image with realistic RGB colors. Training a 3D scene painting network, on the other hand, requires numerous colored 3D scenes with semantic labels for supervision, which are hard to acquire. To overcome this, we propose to utilize techniques already developed for 2D image synthesis. Specifically, given a 3D scene with object-wise semantic labels, we render 2D maps of 3D coordinates and semantic labels. The rendered maps are then fed to a realistic image synthesis module to synthesize 2D RGB images, which are used as pseudo-ground-truth labels for training our 3D scene painting network to produce realistic colors. As our network is conditioned on 3D coordinates, it produces consistent colors under the change of viewpoint.

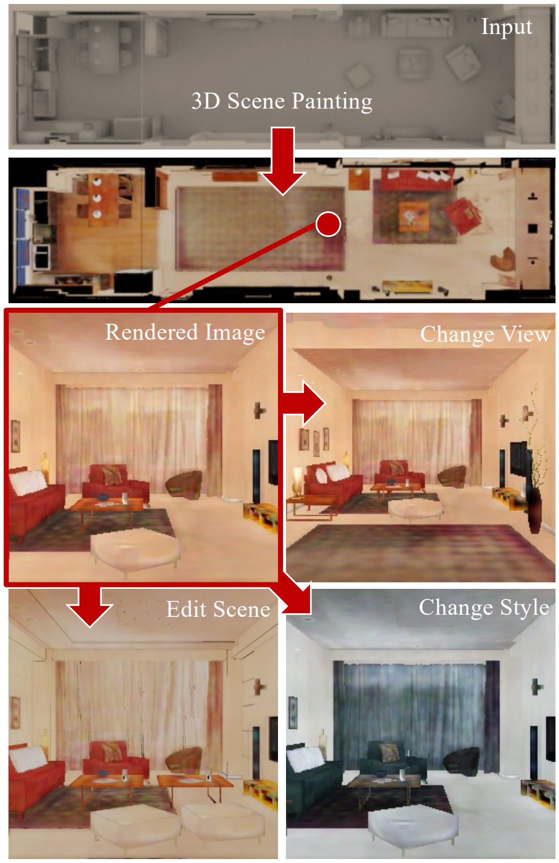


Figure 1. The colored 3D scenes using the proposed method. Our framework learns to paint a scene given 3D geometry and semantic label map. The images are rendered from colored 3D scenes and provide a way to change viewpoints, scene style manipulation, and scene editing.

[*] Part of this work was done while the first author was a research intern at Microsoft Research Asia.

The proposed method has a few merits compared to previous works on 3D scene painting or texture mapping. First, our approach can generate a quality texture of a scene with the aid of a generative adversarial loss. Second, our approach can change the style of a scene by simply manipulating a style vector. Third, our approach allows a user detailed control of scene layouts, and change of the viewpoint or the positions of objects for image rendering.

The proposed scene painting network can be regarded as a geometrically conditioned image synthesis from an image synthesis perspective. Concurrent image synthesis approaches utilize a rough guide, such as image class [2, 21, 31, 34, 51], semantic labels [18, 38, 41, 47], attributes [43], poses [3], voxelized scenes [14], or viewing-directions [32]. On the other hand, our approach opens a new research direction for cases when 3D scenes are provided.

We apply our approach to various indoor scenes. Our qualitative and quantitative experimental results verify that our method is highly controllable and produces high-quality colored scenes and images.

Our contributions can be summarized as follows:

- We propose a novel approach for automatic 3D scene painting, which is based on a novel 3D scene painting network that produces realistic RGB colors from 3D coordinates and semantic class labels.
- Our approach can learn 3D scene painting without ground-truth colored 3D scenes by combining off-the-shelf 2D semantic image synthesis and 2D renderings of 3D semantic labels and coordinates.
- Our approach allows detailed control over scene layouts and change of the viewpoint and object poses.

2. Related Work

Semantic image synthesis. Since the emergence of generative adversarial networks (GANs) [11] and conditional GANs (cGANs) [18], a number of approaches have been proposed that utilize 2D semantic label maps to control the image synthesis process. Wang *et al.* [47] proposed a coarse-to-fine generator and a multi-scale discriminator to achieve high-resolution image synthesis. Park *et al.* [38] proposed spatially-adaptive normalization (SPADE) layers. Schönfeld *et al.* [41] utilize a semantic segmentation network as a discriminator. Zhu *et al.* [55] proposed a group convolution-based network. Ntavelis *et al.* [33] proposed a method for image editing using semantic labels. While these approaches show astonishing results, these approaches are limited to the synthesis of 2D images that are neither multi-view consistent nor conditioned by 3D geometries. On the other hand, we tackle the problem of realistic 3D scene coloring, where we aim to produce multi-view consistent images with realistic colors for a given 3D scene.

Material suggestion. Material suggestion methods [6, 19, 54] aim to automatically assign texture maps to input 3D meshes by searching an external database. They require a large-scale database of textured 3D models [19, 54] or images with 3D material annotations [6], but both of which are expensive to acquire. We aim to automatically generate a realistic 3D scene coloring with a deep generative model.

Image synthesis for 3D scenes. There have been various attempts to incorporate 3D information for image generation, such as novel view synthesis, texture synthesis, and 3D-aware generative models. Novel view synthesis approaches aim at generating images of novel viewpoints from a single input image or multiple images [36, 49, 53]. However, it is hard to synthesize an image that is largely deviated from the original viewing directions or to allow the manipulation of objects, such as the adjustment of objects' poses.

Most previous texture synthesis approaches for 3D scenes aim at generating textures of a single 3D object. Grigorev *et al.* [12] and Huang *et al.* [17] proposed image synthesis methods conditioned on the arbitrary viewpoint for an object. Henderson *et al.* [15] proposed a method for synthesizing a textured 3D mesh. Martin-Brualla *et al.* [28] presented a compact representation for reconstructing thin 3D structures by combining a coarse shape collection with their learned textures. Oechsle *et al.* [35] and Schwarz *et al.* [42] proposed a texture field and a radiance field, respectively, both of which is a mapping function from a 3D point to a color value. Unfortunately, these methods focus on a single 3D object, and it is not trivial to extend them to handle 3D scenes having multiple objects of different classes. Recently, Liao *et al.* [25] introduced a generative model for textures of multiple 3D objects. However, their approach is limited to a small number of simple objects due to the complexity of the approach. Liu *et al.* [26] proposed a pipeline that uses camera trajectories and generates an outdoor natural image sequence of an infinite length. However, as their approach relies on depth prediction, handling indoor scenes with heavy occlusions and various thin structures gets hard.

Recently, GANcraft [14] introduces an approach that generates geometrically consistent and realistic outdoor images. Similarly to ours, GANcraft trains a 3D-aware GAN using pseudo-ground-truth data and synthesizes images conditioned by 3D geometries. However, GANcraft uses the voxel representation, which is limited in representing fine geometric details. Moreover, to synthesize each image, it relies on frame-by-frame image synthesis procedure that needs to render a voxel-based feature map to synthesize an image for each viewpoint. In contrast, our approach is designed for indoor scenes with fine geometric details, and assigns a color value for each 3D coordinate in a scene, so it does not require frame-by-frame synthesis. In addition, since our approach assigns a color value for each 3D coordinate, it can be directly used for scene manipulation.

Implicit representation. Another relevant work to ours is implicit representation-based approaches. These approaches use implicit representations or continuous representations, a class of learnable functions that map a coordinate to a particular type of signal, e.g., color and voxel occupancy. Occupancy Networks [29] and DeepSDF [37] reconstruct a 3D model by introducing an implicit function that takes a 3D coordinate as input and predicts the 3D occupancy of that position. Oechsle *et al.* [35] learn a function that maps a 3D coordinate to color. NeRF [30] takes a 3D coordinate and a viewing direction as input and predicts a novel-view image. Schwarz *et al.* [42] designed a generative model based on NeRF. Anokhin *et al.* [1] propose an image generator that independently calculates the color value at each pixel given a random vector and a 2D coordinate of that pixel. Sitzmann *et al.* [44] showed that periodic activation functions could improve the representational performance. Oechsle *et al.* [35] learn a function that maps a 3D coordinate to a color value. Peng *et al.* [39] reconstructs a whole scene using a convolutional neural network that predicts occupancy. Encoding the coordinates is known to yield successful results for the implicit mapping from a coordinate to the desired output. NeRF [30] found that the sinusoidal positional encoding improves the representation power. Other approaches [1, 46] show that mapping Fourier features [40] enables to learn high-frequency functions.

These advances greatly inspire our approach. To effectively learn consistent color information of an input 3D scene from independently generated pseudo images, our method is designed with an implicit function that maps a 3D coordinate to an RGB color. To enhance the image quality, we adopt positional encoding. Nevertheless, our method allows scene-level image synthesis and object manipulation for complex 3D scenes in contrast to prior work.

3. Method

This section introduces a problem definition for 3D scene painting and our pipeline that benefits from conditional image synthesis. Figure 2 shows an overview of our 3D scene painting framework.

3.1. Problem Definition

The objective of our approach is to colorize a 3D scene. Specifically, a 3D scene $\mathbb{S}$ consists of 3D meshes of objects $\mathbb{M}_i$ with their semantic class labels l_i, i.e., $\mathbb{S} = \{(\mathbb{M}_i, l_i) | 1 \leq i \leq n(\mathbb{S})\}$, where $n(\mathbb{S})$ is the number of mesh models in the scene. For a given 3D scene $\mathbb{S}$, our goal is to generate a realistic RGB color $\boldsymbol{c}_j \in \mathbb{R}^3$ for each point $\boldsymbol{p}_j$ on the surfaces of the meshes in $\mathbb{S}$. In addition, we incorporate a style vector $\mathbf{z}$ to give ability to manipulate the color distribution of the 3D scene.

3.2. Data Preparation

Learning to paint 3D scenes requires color supervision, while it is not easy to access an extensive collection of 3D scenes with realistic colors. Instead, to train our scene painting network without direct supervision, we synthesize pseudo-ground-truth labels using an off-the-shelf conditional image synthesis method based on semantic segmentation maps. In the following, we describe our training data generation process.

Label map rendering. Given a 3D scene $\mathbb{S}$, we sample a set of multiple viewpoints. Then, from each viewpoint, we render a depth map $\mathbf{D} \in \mathbb{R}^{H \times W}$ and a label map $\mathbf{L} \in \mathbb{R}^{H \times W}$, where H and W are the height and width of the map, respectively. We transform $\mathbf{L}$ to $\mathbf{L}' \in \mathbb{R}^{H \times W \times C}$ by converting the class label at each pixel to a one-hot vector, where C is the number of class labels. By using the camera parameters of the current viewpoint, we convert the depth map to a coordinate map $\mathbf{P} \in \mathbb{R}^{H \times W \times 3}$ whose pixel values indicate the world coordinates in the 3D scene. In this manner, we render the input 3D scene from multiple viewpoints and acquire various pairs of a coordinate map and a label map that is denoted as $\{(\mathbf{P}_v, \mathbf{L}'_v) | 1 \leq v \leq V\}$, where V is the number of viewpoints.

Pseudo ground-truth generation. We then generate pseudo-ground-truth images $\{\mathbf{I}'\}$ from generated label maps $\{\mathbf{L}'\}$ using an off-the-shelf image synthesis network as a pseudo image generator $\mathcal{G}'$ where $\mathbf{I}' \in \mathbb{R}^{H \times W \times 3}$ For $\mathcal{G}'$, we use OASIS [41] trained on the ADE20K dataset [52], a state-of-the-art image synthesis network that produces a realistic 2D image from a semantic label map and a style vector $\boldsymbol{z}$. Exploiting $\mathcal{G}'$, we generate pseudo ground-truth images $\{\mathbf{I}'\}$ as $\mathbf{I}' = \mathcal{G}'(\mathbf{L}', \boldsymbol{z})$. By sampling random $\boldsymbol{z}$ from the normal distribution, we gather images of various styles for each scene. While the pseudo-ground-truth images $\mathbf{I}'$ are realistic-looking thanks to the advances in the 2D image synthesis, they are not multi-view consistent because they are generated independently. Nonetheless, our scene painting network can still learn multi-view consistent image generation thanks to its network architecture and learning strategy, which will be explained in the following.

3.3. Scene Painting Network

Our 3D scene painting network $\mathcal{G}$ generates colors for the points in a 3D scene for a given 3D coordinate map, a label map, and a style vector. Specifically, $\mathcal{G}$ performs 3D scene coloring as $\mathbf{I} = \mathcal{G}(\mathbf{X}, \mathcal{W}(\boldsymbol{z}))$ where $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$ is a resulting 2D RGB image, $\mathbf{X}$ is a concatenation of a positional encoding map $\gamma(\mathbf{P})$ and $\mathbf{L}'$, and $\mathcal{W}$ is a learnable style mapping network inspired by StyleGANv2 [22]. F is the sum of the number of channels in $\gamma(\mathbf{P})$ and $\mathbf{L}'$. We utilize sinusoidal function-based positional encoding $\gamma(\cdot)$ [30]. For each 3D world coordinate $\boldsymbol{x} = [x, y, z]$ in $\mathbf{P}$,

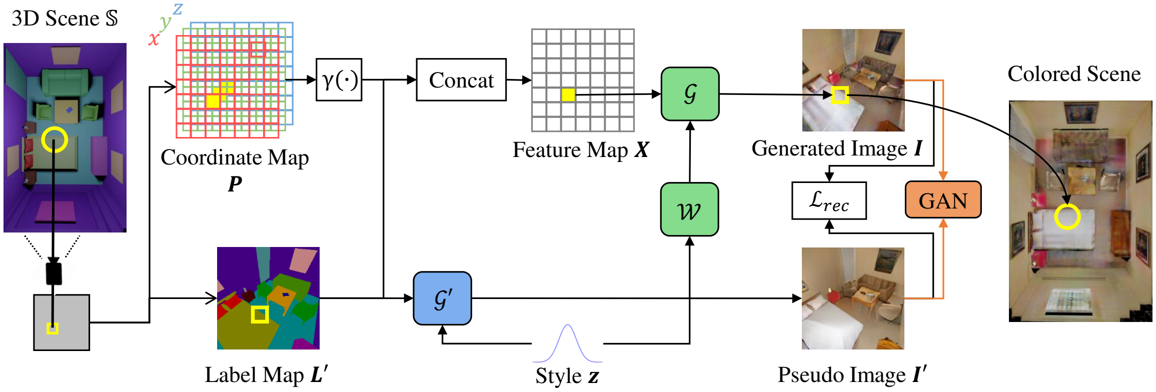

3D Scene $\mathbb{S}$
x
y
z
Coordinate Map $\boldsymbol{P}$
$\gamma(\cdot)$
Concat
Feature Map $\boldsymbol{X}$
$\mathcal{G}$
$\mathcal{W}$
Generated Image $\boldsymbol{I}$
$\mathcal{L}_{rec}$
GAN
Colored Scene
Label Map $\boldsymbol{L}'$
$\mathcal{G}'$
Style $\boldsymbol{z}$
Pseudo Image $\boldsymbol{I}'$

Figure 2. Overview of the proposed method. Given a 3D scene, we render a coordinate map $\mathbf{P}$ and a semantic label map $\mathbf{L}'$ from arbitrary viewpoints. Our scene painting network $\mathcal{G}$ is trained with pseudo images generated by $\mathcal{G}'$. $\mathcal{G}$ generates color from the feature map $\mathbf{X} = [\gamma(\mathbf{P}); \mathbf{L}']$, which is a concatenation of $\gamma(\mathbf{P})$ and $\mathbf{L}'$ in the channel dimension. $\gamma(\cdot)$ is a element-wise positional encoding function. $\mathcal{G}$ and $\mathcal{G}'$ share the style vector $\boldsymbol{z}$, and $\boldsymbol{z}$ passes the style mapping network $\mathcal{W}$. The generated image $\mathbf{I}$ is compared with the pseudo image $\mathbf{I}'$ by $\mathcal{L}_{rec}$. The segmentation-based discriminator classifies the authenticity of the generated images $\mathbf{I}$ and pseudo images $\mathbf{I}'$. Note that $\mathcal{G}$ generates color for *each 3D point* and $\mathcal{L}_{rec}$ is defined for *each 3D point*. After training, the generated colors are mapped to the 3D scene, and we get the colored scene.

which is normalized into $[-1, 1]$, $\gamma(\cdot)$[1] produces a $(6T+3)$-dimensional encoding vector where T is a hyperparameter.

The scene painting network $\mathcal{G}$ consists of nine fully-connected layers that independently assign a color to each coordinate. The layers use style adaptive weight modulation adopted from StyleGANv2 [22]. The detailed architecture is provided in the supplement. We train $\mathcal{G}$ with our pseudo-ground-truth images by optimizing a reconstruction loss $\mathcal{L}_{rec}$ and an adversarial loss $\mathcal{L}_{adv}$. We define the reconstruction loss $\mathcal{L}_{rec}$ as a combination of an L1, L2, and VGG perceptual loss [20], i.e.,:

$$\begin{aligned}\mathcal{L}_{rec} = \frac{1}{K}\sum_{b=1}^{B}\sum_{i=1}^{H\times W}\Big(&\mathbf{A}_i^b\left\|\mathbf{I}_i^b - \mathbf{I}_i'^b\right\|_1 + \\ &\lambda_{L2}\mathbf{A}_i^b\left\|\mathbf{I}_i^b - \mathbf{I}_i'^b\right\|_2\Big) + \mathcal{L}_{VGG}\end{aligned} \tag{1}$$

where $K = BHW$ is a normalization factor, B is the mini-batch size, and H and W are the height and width of a generated image, respectively. $\mathbf{I}^b$ and $\mathbf{I}'^b$ are the b-th generated image and pseudo image in the current mini-batch, and $\mathbf{I}_i^b$ indicates the i-th pixel of $\mathbf{I}^b$. $\mathbf{A}^b$ is the b-th adaptive weight map, which will be explained later. $\mathcal{L}_{VGG}$ is a perceptual loss [20] between $\mathbf{I}$ and $\mathbf{I}'$. λ_{L2} is a scalar weight and we use $\lambda_{L2} = 10$.

Pseudo images from different viewpoints may have different textures (Figure 3 (b)) and artifacts (Figure 7 (a)), which may eventually lead to artifacts in the final 3D scene coloring results. The reconstruction loss adopts the adaptive weight map to discard such inconsistent parts of pseudo images during training. The adaptive weight map $\mathbf{A} \in \mathbb{R}^{H\times W\times 3}$ is defined as:

$$\begin{aligned}\mathbf{A}_i &= \exp\left(-\frac{||\mathbf{I}_i' - \boldsymbol{\mu}(\mathbf{I}_i', \mathbf{L}_i)||_2^2}{\sigma}\right), \\ \boldsymbol{\mu}(\mathbf{I}_i', \mathbf{L}_i) &= \frac{\sum_{j=1}^{H\times W}\mathbf{I}_j'\mathbb{1}\left[\mathbf{L}_j = \mathbf{L}_i\right]}{\sum_{j=1}^{H\times W}\mathbb{1}\left[\mathbf{L}_j = \mathbf{L}_i\right]}\end{aligned} \tag{2}$$

where $\boldsymbol{\mu}(\mathbf{I}_i', \mathbf{L}_i) \in \mathbb{R}^3$ is a label-wise mean vector of pseudo image pixel values whose semantic class is $\mathbf{L}_i$. $\mathbb{1}$ is an indicator function. $\sigma = 0.1$ is a scalar that controls smoothness of the Gaussian function.

Minimizing the reconstruction loss $\mathcal{L}_{rec}$ tends to mix clashing colors from pseudo images of different viewpoints, and it makes $\mathcal{G}$ to produce blurry images as a trivial solution. To alleviate this, we adopt adversarial learning to guide $\mathcal{G}$ to synthesize more realistic-looking images with detailed textures. We adopt the semantic segmentation-based adversarial learning [41], where a discriminator learns to classify each pixel of an image into $(C+1)$ semantic classes. The additional class accounts for one fake class. Our framework uses a label map $\mathbf{L}'$ as ground truth for the discriminator.

The adversarial loss $\mathcal{L}_{adv}$ is defined as:

$$\mathcal{L}_{adv} = -\frac{1}{K}\sum_{b=1}^{B}\sum_{c=1}^{C}\alpha_c\sum_{i=1}^{H\times W}\mathbf{L}_{i;c}'^b\log\mathcal{D}\left(\mathbf{I}^b\right)_{i;c} \tag{3}$$

where α_c is a weight for each class to resolve the class imbalance problem [41], $\mathcal{D}$ is a semantic segmentation-based discriminator, and $\mathbf{L}_{i;c}'^b$ is the c-th element of the class label represented as a one-hot vector at the i-th pixel of the

[1] $\gamma(\boldsymbol{x}) = [\boldsymbol{x}^\intercal, \gamma_0(\boldsymbol{x}), \cdots, \gamma_{T-1}(\boldsymbol{x})]^\intercal$, where $\gamma_t(\boldsymbol{x}) = [\sin(2^t\pi x), \cos(2^t\pi x), \sin(2^t\pi y), \cos(2^t\pi y), \sin(2^t\pi z), \cos(2^t\pi z)]$

b-th label map, which is either 0 or 1. Similarly, $\mathcal{D}(\mathbf{I}^b)_{i;c}$ is the c-th element at the i-th pixel of the discriminator output $\mathcal{D}(\mathbf{I}^b) \in \mathbb{R}^{H\times W\times C}$. The discriminator $\mathcal{D}$ is trained using a loss defined as:

$$\begin{aligned}\mathcal{L}_{\mathcal{D}} = \lambda_{adv}\Big(&-\frac{1}{K}\sum_{b=1}^{B}\sum_{c=1}^{C}\alpha_c\sum_{i=1}^{H\times W}\mathbf{L}'^{b}_{i;c}\log\mathcal{D}\left(\mathbf{I}'^{b}\right)_{i;c} \\ &-\frac{1}{K}\sum_{b}^{B}\sum_{i=1}^{H\times W}\log\mathcal{D}\left(\mathbf{I}^{b}\right)_{i;C+1}\Big).\end{aligned} \tag{4}$$

The total loss for training $\mathcal{G}$ is then defined as: $\mathcal{L}_{\mathcal{G}} = \mathcal{L}_{rec} + \lambda_{adv}\mathcal{L}_{adv}$, where λ_{adv} is a scalar weight.

3.4. Texture Mapping

After training, we create a texture map of each mesh model $\mathbb{M}_i$ in the scene $\mathbb{S}$ using coordinate maps $\mathbf{P}_v$ and the corresponding generated images $\mathbf{I}_v$. Specifically, we retrieve 3D points appearing at each coordinate map and collect color values generated by the scene painting network. Then, using the texture coordinates corresponding to the 3D points, we reconstruct texture maps of meshes.

This step gives us *texture-mapped meshes* of the scene, so we can quickly render arbitrary viewpoints of the colored scenes, and the rearrangement of objects is straightforward.

4. Experiments

4.1. Implementation details

Network and training. We use the MLP architecture mentioned in Sec. 3.3 with the positional encoding parameter $T = 4$. We use leaky-ReLU [27] with a negative slope of 0.2 for the activation functions in our network. For training, we used the Adam [23] optimizer with $\beta_1 = 0, \beta_2 = 0.999$, and the learning rate of 1×10^{-4}. We set the batch size as 8. For the adversarial loss, we set $\lambda_{adv} = 0.1$. We train our scene painting network for 40,000 iterations.

Data preparation. For our experiments, we utilize 3D models and their arrangements from the SceneNet [13] dataset, as it provides complete object meshes, which enable scene editing, unlike other 3D datasets such as Replica [45], MatterPort3D [4], and ScanNet [9]. For qualitative experiments, we use four scenes in the SceneNet dataset: bedroom, kitchen, living room, and office, each of which provides 1.1k, 0.8k, 1k, and 1.8k training frames, respectively, and 49, 48, 64, and 64 test frames, respectively. We separately train the scene painting network for each scene. As SceneNet has no semantic labels on the objects, we assign each 3D model a class label using the 150 labels in the ADE20k [52] dataset[2]. We use Blender [8] to render label maps and depth maps from multiple viewpoints.

Toy scene. For quantitative experiments, including ablation studies, we create a toy example scene by modifying a bedroom scene of SceneNet [13]. The scene is of a room with objects of 15 classes.

4.2. Qualitative results

Image quality and view consistency. In Figure 3, we visually compare the quality and view consistency of our results with images generated by OASIS [41]. As shown in Figure 3 (b), OASIS produces unnatural textures, especially in the scene with complex geometry and large homogeneous regions. In contrast, our method produces geometrically valid and consistent images (Figure 3 (c)).

Scene style control. Our approach is readily able to control the style of a scene by changing the style vector $\boldsymbol{z}$. We can change the style during the test time, so it does not involve re-training the network. Figure 4 shows example scenes generated with different style vectors.

3D scene editing. Our approach directly assigns texture colors to each mesh in a 3D scene. Thus, once texture colors are assigned, a 3D scene can be easily edited using ordinary 3D operations in contrast to other 3D image synthesis techniques [14, 26]. Figure 5 shows such an example of 3D scene editing where we first synthesized texture maps for a 3D scene using our scene painting network, edited the 3D scene using Blender [8], and rendered new images with the synthesized textures.

4.3. Evaluation metrics

We evaluate our method using the following metrics. We focus on assessing the quality of generated images for quantitative evaluation because there is no available precise quantitative quality measure for colored 3D scenes to our best knowledge.

Frechet Inception Distance (FID). We measure FID [16] between the generated images and real images from the ADE20k dataset [52]. A smaller value indicates that the distribution of generated images is closer to that of real images.

Mean Intersection-over-Union (mIoU). Our method generates a 2D image from a rendered 2D semantic label map. To measure how semantically faithful a generated image is for a given input semantic label map, we measure the mIoU between the input semantic label map and the semantic segmentation result of a generated image predicted by a pre-trained semantic segmentation network [48]. A higher mIoU score indicates that the generated image is more faithful in terms of semantic labels.

View consistency (VC). We also measure the consistency of images generated by different viewpoints. To measure the view consistency among images generated by different viewpoints, we collect color values of 3D points from the images. Then, we compare the color values of 3D points

[2] Available on our project webpage: http://cvlab.postech.ac.kr/research/3DScenePainting.

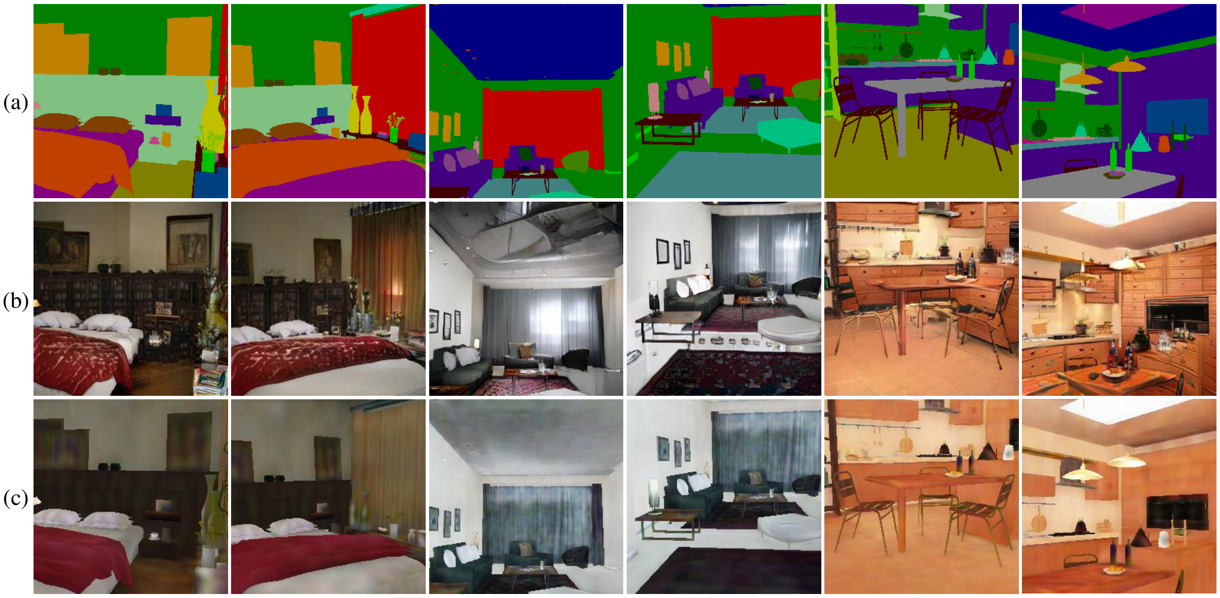


Figure 3. We compared images generated by our method and the images produced by OASIS [41] on the three scenes. For each scene, two images from two different views were compared. (a) Semantic label map, (b) Pseudo images produced by OASIS, and (c) Our results.

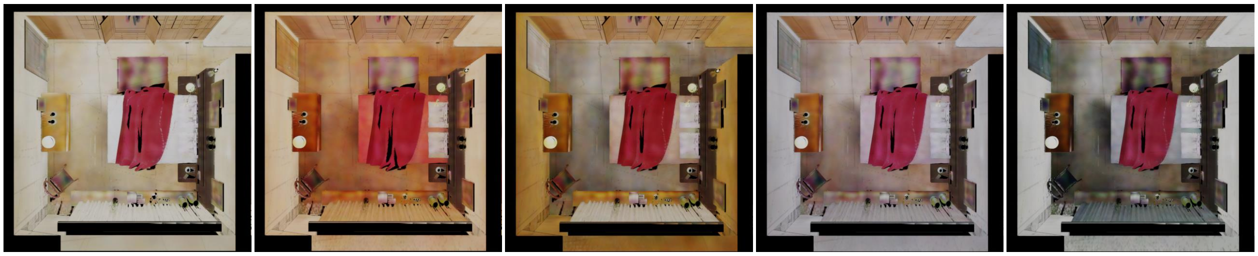


Figure 4. Examples of style manipulation using our scene painting network.

in a local neighborhood in the 3D space. Specifically, we define a local neighborhood $N_{\boldsymbol{p}}$ as a cell in the 3D space whose center is the 3D point $\boldsymbol{p}$, and define $\mathcal{C}(N_{\boldsymbol{p}})$ as a set of pixel colors in $\boldsymbol{p}$. Then, we define the VC metric as:

$$\frac{1}{n(\mathcal{N})} \sum_{N_{\boldsymbol{p}} \in \mathcal{N}} \max_{\boldsymbol{y} \in \mathcal{C}(N_{\boldsymbol{p}}), \boldsymbol{y'} \in \mathcal{C}(N_{\boldsymbol{p}})} |\boldsymbol{y} - \boldsymbol{y'}|_2 \tag{5}$$

where $\boldsymbol{y}$ and $\boldsymbol{y}'$ are color values in the neighborhood $N_{\boldsymbol{p}}$. $\mathcal{N} = \{N_{\boldsymbol{p}} | n(\mathcal{C}(N_{\boldsymbol{p}})) >= 2\}$ is the set of the local neighborhoods having two or more corresponding pixels. A lower VC score indicates higher view consistency.

4.4. Ablation Study

We conduct ablation studies on the scene painting network architecture, dimensions of positional encoding, and loss function. In this ablation study, we use the configuration described in the following as our baseline. We use the MLP architecture mentioned in Sec. 3.3 with positional encoding $T = 4$. We use the reconstruction loss $\mathcal{L}_{rec}$ without adaptive weight $\mathbf{A}$ and adversarial loss with pseudo images with $\lambda_{adv} = 1$. We utilize early stopping and the number of training iterations of 10,000 or 40,000 to avoid overfitting the discriminator. Unless otherwise mentioned, we use this setting in the quantitative experiments.

Architecture of the scene painting network. We conduct an experiment on two architectures. Previous methods based on implicit representations [14, 32] 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 Table 1, MLP+CNN is an MLP architecture followed by four 3×3 convolution layers. A residual connection is added to the output of each convolution layer. The number of training iterations is 10,000. The results indicate there exists a trade-off between the single image quality and view consistency. As architecture has more CNN layers, the receptive field gets larger, and the view consistency ge1qts worse. MLP+CNN achieves better mIoU and FID than MLP for the single image quality. However, we choose the MLP for the scene painting network because the change of colors of CNN layers is noticeable (Figure 6 (a), (b) ,(c)).

Dimension of positional encoding. We use positional encoding $\gamma(\cdot)$ for the input of $\mathcal{G}$, which has a hyperparameter

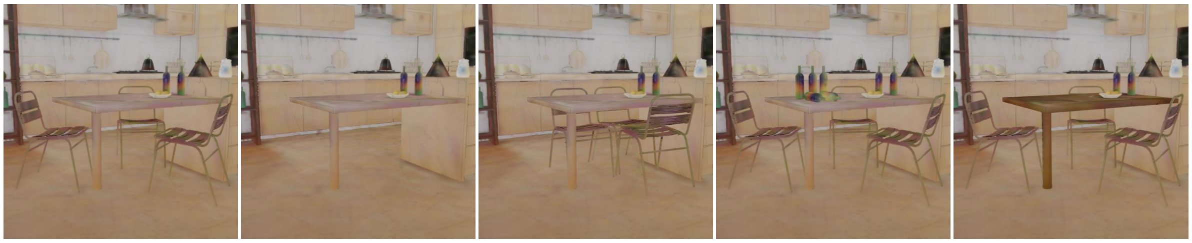


Figure 5. Colored meshes generated with our method, and scene editing examples. (From left to right) the original scene, removed chairs, rearranged chairs, more bottles on the table, and different style of the table.

Table 1. Comparison on scene painting network architectures.

Architecture	Measure		
	mIoU (↑)	FID (↓)	VC (↓)
MLP+CNN	0.461	111.69	37.51
MLP	0.391	132.93	**1.23**

Table 2. Evaluation on various T of positional encoding.

T	Training iter.	Measure		
		mIoU (↑)	FID (↓)	VC (↓)
0	10,000	0.290	174.39	**0.76**
2	10,000	0.396	151.23	0.92
4	10,000	0.391	132.93	1.23
4	40,000	0.427	**115.75**	3.14
6	10,000	0.386	155.02	3.99
10	10,000	0.381	179.77	11.58
10	40,000	**0.431**	148.33	14.36

T that can affect the quality of generated images. To study the effect of T on the generation quality, we test various values for T in Table 2. The scene painting network produces blurry images without positional encoding ($T = 0$). We found that a large T catches the high-frequency details in the pseudo images when the number of training iterations is large enough to reach the convergence of the reconstruction loss. However, when the number of training iterations is large, artifacts of pseudo images emerge in the generated images (Figure 7 (a), (d)). An extensive T causes a grid-like artifact in generated images when the number of training iterations is insufficient to converge the reconstruction loss (Figure 7 (b)). The mIoU values are similar except for $T = 0$ when the number of iterations is 10,000. The mIoU and FID improve when the networks are trained for 40,000 iterations, yet the resulting images have significant artifacts that are not shown in mIoU and FID (Figure 6 (d), (e)). The setting with the number of training iterations of 10,000 and $T = 4$ shows the best FID.

Loss function. We experiment to determine the best configuration for the loss functions. Table 3 shows the effects of $\mathcal{L}_{rec}$, adversarial loss $\mathcal{L}_{adv}$ with pseudo images (GAN-p), adaptive weight $\mathbf{A}$, and λ_{adv}. While the first row shows that employing only $\mathcal{L}_{rec}$ achieves the lowest VC, it is because using only $\mathcal{L}_{rec}$ results in blurry images (Figure 7 (b)). We achieve better mIoU and FID scores when combining L1, L2, VGG perceptual losses, and GAN-p. GAN-p enhances perceptual quality, as evidenced by improved mIoU when looking at the first and fourth rows. In this experiment, we also evaluate the adversarial learning with real indoor images from the ADE20k dataset [52], which is denoted as (GAN-r) in the table. The table shows that GAN-r does not improve the quality because the domain gap between the real images and labels from the 3D scene is enormous, making the training difficult. As mentioned earlier, a large number of iterations causes a GAN artifact, and we resolve this by balancing the λ_{adv}. In the sixth row, when $\lambda_{adv} = 0.1$, training progresses for significant iterations without artifact, and the result shows the better mIoU and FID compared with the fourth row. In the seventh row, when using adaptive weight $\mathbf{A}$, the result shows the better mIoU and FID compared with the fourth row. Finally, the last row shows the best configuration.

Table 3. Ablation study on loss functions. GAN-p is the adversarial loss with pseudo images, and GAN-r is the adversarial loss with real images (ADE20K [52]). When $\lambda_{adv} = 1$, # of iter. is 10k, and when $\lambda_{adv} = 0.1$, # of iter. is 40k.

Setting	λ_{adv}	use $\mathbf{A}$	Measure		
			mIoU (↑)	FID (↓)	VC (↓)
$\mathcal{L}_{rec}$	1.0		0.291	151.57	**0.44**
L1 + GAN-p	1.0		0.370	140.35	2.06
L1 + L2 + GAN-p	1.0		0.366	141.16	1.24
$\mathcal{L}_{rec}$+ GAN-p	1.0		0.391	132.93	1.23
$\mathcal{L}_{rec}$+ GAN-p + GAN-r	1.0		0.385	138.35	1.24
$\mathcal{L}_{rec}$+ GAN-p	0.1		0.394	124.30	1.08
$\mathcal{L}_{rec}$+ GAN-p	1.0	✓	0.393	129.69	1.35
$\mathcal{L}_{rec}$+ GAN-p	0.1	✓	$\underline{0.419}$	$\underline{116.33}$	$\underline{1.00}$
OASIS [41]	-	-	**0.5728**	**84.74**	57.863

We also show the measures of OASIS [41], which is our pseudo image generator (Figure 7 (a)). We generate pseudo images with the same labels used in the other experiments and evaluate the measures with those pseudo images. Although OASIS's perceptual quality is superior to ours, it is difficult for OASIS to create coherent images for different viewpoints. We also emphasize that our goal differs significantly from the image synthesis methods. It is unfair to compare our approach against previous ones solely based on conventional metrics.

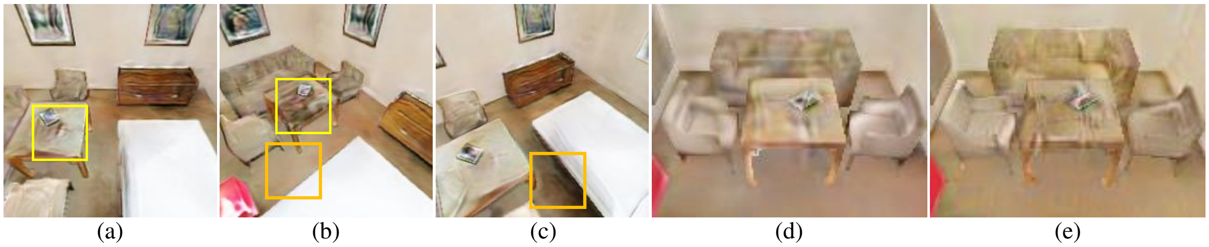


Figure 6. Images that are generated with different architectures. (a), (b) and (c) are generated by MLP + 3 × 3CNN architecture from different viewpoints. (a) and (b) show the color shift of the table, and (b) and (c) show the color shift of the floor. (d) shows the result of MLP architecture with 10k training iterations. (e) shows the result of MLP architecture with 40k iterations.

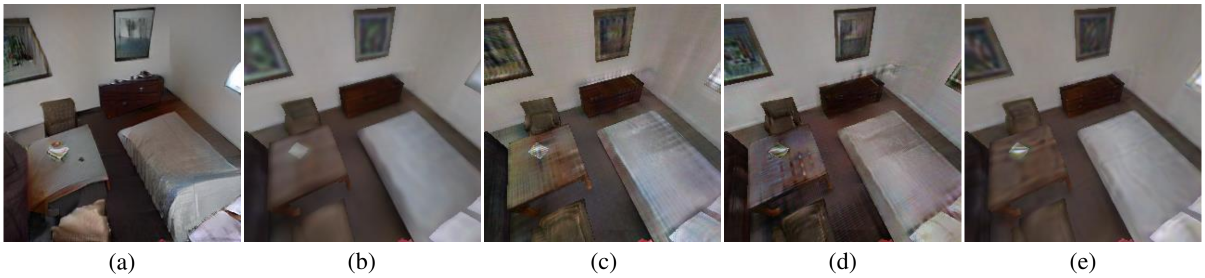
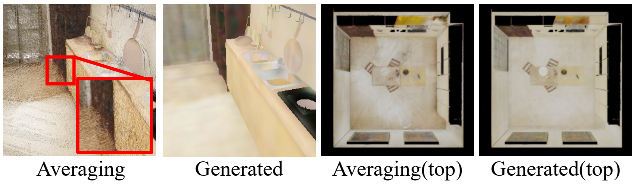


Figure 7. Examples of artifacts. (a) A pseudo image generated from OASIS [41]. (b) $T = 4$, 40k iter., and no adversarial loss. (c) $T = 10$ and 10k iter. (d) $T = 10$ and 40k iter. (e) $T = 4$, 40k iter., and adaptive weight $\mathbf{A}$, and $\lambda_{adv} = 0.1$.



Averaging Generated Averaging(top) Generated(top)

Figure 8. Comparison between the averaging approach and ours.

Comparison with averaging. An alternative to generating view-consistent images from pseudo-ground-truth images would be simply averaging colors from pseudo-ground-truth images for each 3D point. We compare our approach with the averaging approach. The results of the averaging approach show noisy results (Fig. 8). It is because averaging requires many training images without holes. On the other hand, our approach does not suffer from such a problem because it learns *a function* from a 3D coordinate to a color rather than simply memorizing color for each coordinate.

5. Conclusion

This paper proposed a novel 3D scene coloring approach synthesizing the color of configurable 3D scene layout and a training scheme that does not require direct supervision from colored 3D scenes. Given 3D coordinate and semantic label maps, our scene painting network synthesizes view-consistent colored scenes. Our method ensures the view consistency of synthesis, which is not addressed in the semantic image synthesis method. In addition, our method can be used to generate the color of a scene containing multiple objects, allowing users to modify scene color and configuration using 3D graphics tools. An interesting future direction would be to make the pipeline faster and support explainable adjustments of scene styles (make the scenes brighter, darker, or warmer).

Limitation Our approach has several limitations. While our approach can produce coherent images and be combined with any image generation approach, our results depend on the quality of the pseudo image generator. Our approach also shows limited diversity in generated images due to the limited diversity of a pre-trained OASIS model [41] as discussed by Yang et al. [50]. The example is shown in Fig. 4, where the bed cover is consistently red regardless of the different style vectors. The adaptive weight map assumes each class has a single color, which may lead to oversmoothing in some areas. Besides, our approach hardly generates transparent objects such as a window or view-dependent appearances. This limitation stems from the input representation. The network assigns one color to each 3D coordinate, so it cannot model the view-dependent lighting effect. Finally, our approach requires training of a new network for a new scene.

Acknowledgements This work was partly supported by IITP grants funded by the Korea government (MSIT) (2019-0-01906: Artificial Intelligence Graduate School Program (POSTECH) and 2020-0-01649: High-Potential Individuals Global Training Prog.) and by Ministry of Culture, Sports and Tourism (R2021040136: Development of meta-verse contents based on XR & AI)

References

[1] Ivan Anokhin, Kirill Demochkin, Taras Khakhulin, Gleb Sterkin, Victor Lempitsky, and Denis Korzhenkov. Image generators with conditionally-independent pixel synthesis. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14278–14287, 2021. 3

[2] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019. 2

[3] Caroline Chan, Shiry Ginosar, Tinghui Zhou, and Alexei A. Efros. Everybody dance now. *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 5932–5941, 2019. 2

[4] Angel X. Chang, Angela Dai, Thomas A. Funkhouser, Maciej Halber, Matthias Nießner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from RGB-D data in indoor environments. In *International Conference on 3D Vision (3DV)*, pages 667–676. IEEE Computer Society, 2017. 5

[5] Angel X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Qixing Huang, Zimo Li, S. Savarese, M. Savva, Shuran Song, H. Su, J. Xiao, L. Yi, and F. Yu. Shapenet: An information-rich 3d model repository. *arXiv*, abs/1512.03012, 2015. 1

[6] Kang Chen, Kun Xu, Yizhou Yu, Tian-Yi Wang, and Shi-Min Hu. Magic decorator: automatic material suggestion for indoor digital scenes. *ACM Trans. Graph.*, 34(6):232:1–232:11, 2015. 2

[7] Sungjoon Choi, Qian-Yi Zhou, Stephen D. Miller, and Vladlen Koltun. A large dataset of object scans. *arXiv*, abs/1602.02481, 2016. 1

[8] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 5

[9] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas A. Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2432–2443. IEEE Computer Society, 2017. 5

[10] Huan Fu, Rongfei Jia, Lin Gao, Mingming Gong, Binqiang Zhao, Stephen J. Maybank, and Dacheng Tao. 3d-future: 3d furniture shape with texture. *Int. J. Comput. Vis.*, 129(12):3313–3337, 2021. 1

[11] Ian J. Goodfellow, Jean Pouget-Abadie, M. Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014. 2

[12] Artur Grigorev, Artem Sevastopolsky, Alexander Vakhitov, and Victor Lempitsky. Coordinate-based texture inpainting for pose-guided human image generation. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[13] A. Handa, V. Pătrăucean, S. Stent, and R. Cipolla. Scenenet: An annotated model generator for indoor scene understanding. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5737–5743, 2016. 5

[14] 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. 2, 5, 6

[15] P. Henderson, Vagia Tsiminaki, and Christoph H. Lampert. Leveraging 2d data to learn textured 3d mesh generation. *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7495–7504, 2020. 2

[16] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017. 5

[17] J. Huang, J. Thies, A. Dai, A. Kundu, C. Jiang, L. J. Guibas, M. Nießner, and T. Funkhouser. Adversarial texture optimization from rgb-d scans. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1556–1565, 2020. 2

[18] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, 2017. 2

[19] Arjun Jain, Thorsten Thormählen, Tobias Ritschel, and Hans-Peter Seidel. Material memex: automatic material suggestions for 3d objects. *ACM Trans. Graph.*, 31(6):143:1–143:8, 2012. 2

[20] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 4

[21] Minguk Kang and Jaesik Park. Contragan: Contrastive learning for conditional image generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 21357–21369, 2020. 2

[22] 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. 3, 4

[23] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv*, abs/1412.6980, 2015. 5

[24] Marc Levoy, Kari Pulli, Brian Curless, Szymon M. Rusinkiewicz, David Koller, Lucas Pereira, Matt Ginzton, Sean E. Anderson, James Davis, Jeremy Ginsberg, Jonathan Shade, and Duane Fulk. The digital michelangelo project: 3d scanning of large statues. *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 2000. 1

[25] Y. Liao, K. Schwarz, L. Mescheder, and A. Geiger. Towards unsupervised learning of generative models for 3d controllable image synthesis. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5870–5879, 2020. 2

[26] Andrew Liu, Ameesh Makadia, Richard Tucker, Noah Snavely, Varun Jampani, and Angjoo Kanazawa. Infinite nature: Perpetual view generation of natural scenes from a single image. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 14438–14447. IEEE, 2021. 2, 5

[27] 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. 5

[28] Ricardo Martin-Brualla, R. Pandey, Sofien Bouaziz, M. Brown, and D. Goldman. Gelato: Generative latent textured objects. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2

[29] Lars M. Mescheder, Michael Oechsle, M. Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4455–4465, 2019. 3

[30] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, J. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 3

[31] Takeru Miyato and Masanori Koyama. cGANs with projection discriminator. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018. 2

[32] 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. 2, 6

[33] Evangelos Ntavelis, Andrés Romero, Iason Kastanis, Luc Van Gool, and Radu Timofte. Sesame: Semantic editing of scenes by adding, manipulating or erasing objects. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 394–411, Cham, 2020. 2

[34] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier GANs. In *Proceedings of the International Conference on Learning Representations (ICLR)*, volume 70 of *Proceedings of Machine Learning Research*, pages 2642–2651, 06–11 Aug 2017. 2

[35] Michael Oechsle, Lars M. Mescheder, M. Niemeyer, Thilo Strauss, and Andreas Geiger. Texture fields: Learning texture representations in function space. *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 4530–4539, 2019. 2, 3

[36] E. Park, J. Yang, E. Yumer, D. Ceylan, and A. C. Berg. Transformation-grounded image generation network for novel 3d view synthesis. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 702–711, 2017. 2

[37] Jeong Joon Park, P. Florence, J. Straub, Richard A. Newcombe, and S. Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 165–174, 2019. 3

[38] T. Park, Ming-Yu Liu, T. Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2332–2341, 2019. 2

[39] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 523–540, 2020. 3

[40] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2007. 3

[41] 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. 2, 3, 4, 5, 6, 7, 8

[42] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 20154–20166, 2020. 2, 3

[43] Wei Shen and Rujie Liu. Learning residual images for face attribute manipulation. *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1225–1233, 2017. 2

[44] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 7462–7473, 2020. 3

[45] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard A. Newcombe. The replica dataset: A digital replica of indoor spaces. *arXiv*, abs/1906.05797, 2019. 1, 5

[46] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3

[47] T. Wang, Ming-Yu Liu, Jun-Yan Zhu, A. Tao, J. Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8798–8807, 2018. 2

[48] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 432–448, 2018. 5

[49] X. Xu, Y. Chen, and J. Jia. View independent generative adversarial network for novel view synthesis. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 7790–7799, 2019. 2

[50] Dingdong Yang, Seunghoon Hong, Yunseok Jang, Tiangchen Zhao, and Honglak Lee. Diversity-sensitive conditional generative adversarial networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019. 8

[51] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pages 7354–7363, 09–15 Jun 2019. 2

[52] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5122–5130, 2017. 3, 5, 7

[53] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A. Efros. View synthesis by appearance flow. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 286–301, 2016. 2

[54] J. Zhu, Y. Guo, and H. Ma. A data-driven approach for furniture and indoor scene colorization. *IEEE Transactions on Visualization and Computer Graphics*, 24:2473–2486, 2018. 1, 2

[55] Z. Zhu, Z. Xu, A. You, and X. Bai. Semantically multimodal image synthesis. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5466–5475, 2020. 2