DisCoScene: Spatially Disentangled Generative Radiance Fields for Controllable 3D-aware Scene Synthesis

Yinghao Xu[1,2*]   Menglei Chai[2†]   Zifan Shi[3]   Sida Peng[4]

Ivan Skorokhodov[5,2*]   Aliaksandr Siarohin[2]   Ceyuan Yang[1]   Yujun Shen[1]

Hsin-Ying Lee[2]   Bolei Zhou[6]   Sergey Tulyakov[2]

[1]CUHK   [2]Snap Inc.   [3]HKUST   [4]ZJU   [5]KAUST   [6]UCLA

Abstract

Existing 3D-aware image synthesis approaches mainly focus on generating a single canonical object and show limited capacity in composing a complex scene containing a variety of objects. This work presents DisCoScene: a 3D-aware generative model for high-quality and controllable scene synthesis. The key ingredient of our method is a very abstract object-level representation (i.e., 3D bounding boxes without semantic annotation) as the scene layout prior, which is simple to obtain, general to describe various scene contents, and yet informative to disentangle objects and background. Moreover, it serves as an intuitive user control for scene editing. Based on such a prior, the proposed model spatially disentangles the whole scene into object-centric generative radiance fields by learning on only 2D images with the global-local discrimination. Our model obtains the generation fidelity and editing flexibility of individual objects while being able to efficiently compose objects and the background into a complete scene. We demonstrate state-of-the-art performance on many scene datasets, including the challenging Waymo outdoor dataset. Project page can be found here.

1. Introduction

3D-consistent image synthesis from single-view 2D data has become a trendy topic in generative modeling. Recent approaches like GRAF [39] and Pi-GAN [5] introduce 3D inductive bias by taking neural radiance fields [1, 27, 28, 35, 37] as the underlying representation, gaining the capability of geometry modeling and explicit camera control. Despite their success in synthesizing individual objects (*e.g.*, faces, cats, cars), they struggle on scene images that contain multiple objects with non-trivial layouts and complex backgrounds. The varying quantity and large

diversity of objects, along with the intricate spatial arrangement and mutual occlusions, bring enormous challenges, which exceed the capacity of the object-level generative models [4, 13, 15, 32, 44, 45, 59].

Recent efforts have been made towards 3D-aware scene synthesis. Despite the encouraging progress, there are still fundamental drawbacks. For example, Generative Scene Networks (GSN) [8] achieve large-scale scene synthesis by representing the scene as a grid of local radiance fields and training on 2D observations from continuous camera paths. However, object-level editing is not feasible due to spatial entanglement and the lack of explicit object definition. On the contrary, GIRAFFE [31] explicitly composites object-centric radiance fields [16, 33, 55, 62] to support object-level control. Yet, it works poorly on challenging datasets containing multiple objects and complex backgrounds due to the absence of proper spatial priors.

To achieve high-quality and controllable scene synthesis, the scene representation stands out as one critical design focus. A well-structured scene representation can scale up the generation capability and tackle the aforementioned challenges. Imagine, given an empty apartment and a furniture catalog, what does it take for a person to arrange the space? Would people prefer to walk around and throw things here and there, or instead figure out an overall layout and then attend to each location for the detailed selection? Obviously, a layout describing the arrangement of each furniture in the space substantially eases the scene composition process [17, 26, 57]. From this vantage point, here comes our primary motivation — an abstract object-oriented scene representation, namely a *layout prior*, could facilitate learning from challenging 2D data as a lightweight supervision signal during training and allow user interaction during inference. More specifically, to make such a prior easy to obtain and generalizable across different scenes, we define it as a set of object bounding boxes without semantic annotation, which describes the spatial composition of objects in the scene and supports intuitive object-level editing.

In this work, we present *DisCoScene*, a novel 3D-aware generative model for complex scenes. Our method allows for high-quality scene synthesis on challenging datasets and flexible user control of both the camera and scene objects. Driven by the aforementioned *layout prior*, our model spatially disentangles the scene into compositable radiance fields which are shared in the same object-centric generative model. To make the best use of the prior as a lightweight supervision during training, we propose global-local discrimination which attends to both the whole scene and individual objects to enforce spatial disentanglement between objects and against the background. Once the model is trained, users can generate and edit a scene by explicitly controlling the camera and the layout of objects' bounding boxes. In addition, we develop an efficient rendering pipeline tailored for the spatially-disentangled radiance fields, which significantly accelerates object rendering and scene composition for both training and inference stages.

Our method is evaluated on diverse datasets, including both indoor and outdoor scenes. Qualitative and quantitative results demonstrate that, compared to existing baselines, our method achieves state-of-the-art performance in terms of both generation quality and editing capability. Tab. 1 compares *DisCoScene* with relevant works. it is worth noting that, to the best of our knowledge, *DisCoScene* stands as the first method that achieves high-quality 3D-aware generation on challenging datasets like WAYMO [47], while enabling interactive object manipulation.

2. Related Work

3D-aware Image Synthesis. Generative Adversarial Networks (GANs) have achieved remarkable success in 2D image synthesis [14, 22–25], and have recently been extended to 3D-aware image generation. VON [67] and HoloGAN [29] introduce voxel representations to the generator and use neural rendering to project 3D voxels into 2D space. Then, GRAF [39] and Pi-GAN [5] propose to use implicit functions to learn NeRF from single-view image collections, resulting in better multi-view consistency compared to voxel-based methods. GOF [58], ShadeGAN [34], and GRAM [7] introduce occupancy field, albedo field and radiance surface instead of radiance field to learn better 3D shapes. However, high-resolution image synthesis with direct volumetric rendering is usually expensive. Many works [4, 15, 31, 32, 44, 59, 61] resort to convolutional upsamplers to improve the rendering resolution and quality with lower computation overhead. While some other works [40,45] adopt patch-based sampling and sparse-voxel to speed up training and inference. Note that most of these methods are restricted to well-aligned objects and fail on more complex, multi-object scene imagery. Our work instead naturally handles multi-object scenes with spatial disentangled object-level radiance fields, which can

Table 1. **Comparison of *DisCoScene* and relevant works.** *Multiple Objects*: Ability to model multiple objects in a scene. *Radiance Field*: If radiance fields are used to model scenes. *Complex Scene*: Ability to handle complex datasets beyond diagnostic scenes. *Object Editing*: If object-level editing is supported. *No Camera Sequence*: Not requiring ground truth camera sequences.

Model	Multiple Objects	Radiance Field	Complex Scene	Object Editing	No Camera Sequence
GRAF [39]	✗	✓	✗	✓	✓
BlockGAN [29]	✓	✗	✗	✓	✓
GSN [8]	✓	✓	✓	✗	✗
GIRAFFE [31]	✓	✓	✗	✓	✓
DisCoScene	✓	✓	✓	✓	✓

be scaled to very challenging real-world scene datasets.

Scene Generation. Scene generation has been a long-standing task. Early works like [51] attempt to model a complex scene by trying to generate it. Recently, with the successes in generative models, scene generation has been advanced significantly. Among them, one popular line is to resort to the setups of image-to-image translation from given conditions, *i.e.*, semantic masks [19,36,48,49,54,66], object-attribute graph [2]. Although able to synthesize photorealistic scene images, they struggle to manipulate the objects in 3D space due to the lack of 3D understanding. Some works [9, 52, 63, 64] reuse the knowledge from 2D GAN models to achieve scene manipulation like the camera pose. But they suffer from poor multi-view consistency due to inadequate geometry modeling. Another active line of work [9, 30, 31, 63] explores adding 3D inductive biases to the scene representation. BlockGAN [30] and GIRAFFE [31] introduce compositional voxels and radiance fields to encode the object structures, but their object control can only be performed at simple diagnostic scenes. DepthGAN [43] introduces depth as a 3D prior but is hard to achieve manipulation and multi-view consistency. GSN [8] proposes to represent a scene with a grid of local radiance fields. However, since this local radiance field does not properly link to the object semantics, individual objects cannot be manipulated with versatile user control. Our work proposes to use an abstract layout prior to spatially disentangle the whole scene into object-centric radiance fields, which enables 3D-aware image synthesis on challenging real-world imagery like WAYMO [47].

3. Method

The overall framework is illustrated in Fig. 1. We employ layout as an explicit prior to disentangle objects in our approach (Sec. 3.1). Based on the layout prior, we introduce our spatially disentangled radiance fields (Sec. 3.2) and an efficient rendering pipeline (Sec. 3.3) to achieve controllable 3D-aware scene generation. We also describe our global-local discrimination, which makes training on challenging datasets possible (Sec. 3.4). Finally, we discuss
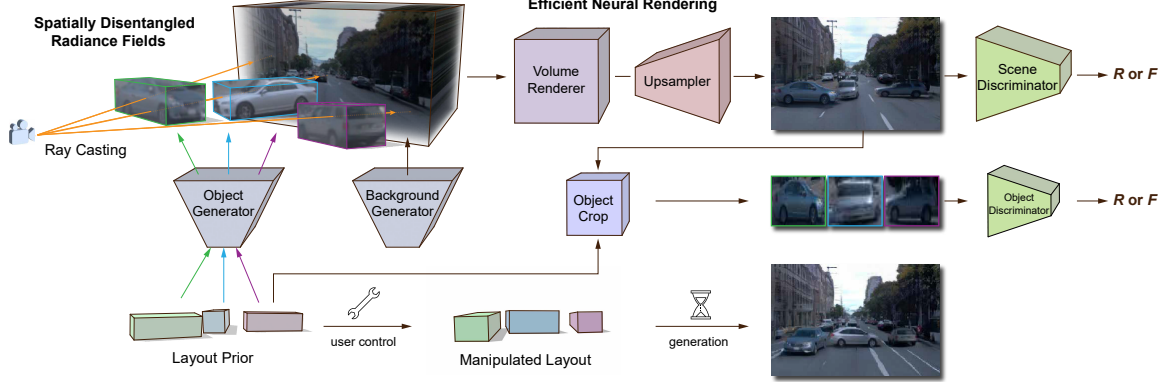
Figure 1. **The overall pipeline of *DisCoScene*.** Conditioned by the *layout prior*, our *spatial disentangle generative radiance fields* generate individual objects and the background. Our *efficient neural rendering* pipeline then composites the scene to a low-resolution feature map with the *volume renderer* and upsamples to the final high-resolution image with the *upsampler*. During training, we propose *global-local discrimination* which applies the *scene discriminator* to the entire image and the *object discriminator* to cropped object patches. During inference, users can manipulate the layout to control the generation of a specific scene at the object level.

our model's training and inference details on 2D image collections (Sec. 3.5).

3.1. Abstract Layout Prior

There exist many representations of a scene, including the popular choice of scene graph [6, 20, 33, 46], where objects and their relations are denoted as nodes and edges. Although graph can describe a scene in rich details, its structure is hard to process and the annotation is laborious to obtain in our case. Therefore, we opt to represent the scene layout in a much-simplified manner – a set of bounding boxes $\mathcal{B} = \{\mathbf{B}_i | i \in [1, N]\}$ without category annotation, where N counts objects in the scene. Concretely, each bounding box is defined with 9 parameters, including rotation $\boldsymbol{a}_i$, translation $\boldsymbol{t}_i$, and scale $\boldsymbol{s}_i$.

$$\mathbf{B}_i = [\boldsymbol{a}_i, \boldsymbol{t}_i, \boldsymbol{s}_i], \qquad (1)$$

$$\boldsymbol{a}_i = [a_x, a_y, a_z], \boldsymbol{t}_i = [t_x, t_y, t_z], \boldsymbol{s}_i = [s_x, s_y, s_z], \quad (2)$$

where $\boldsymbol{a}_i$ comprises 3 Euler angles, which are easier to convert into rotation matrix $\mathbf{R}_i$. Using this notation, the bounding box $\mathbf{B}_i$ can be transformed from a canonical bounding box $\mathbf{C}$, *i.e.*, a unit cube at the coordinate origin:

$$\mathbf{B}_i = \mathrm{b}_i(\mathbf{C}) = \mathbf{R}_i \cdot \mathrm{diag}(\boldsymbol{s}_i) \cdot \mathbf{C} + \boldsymbol{t}_i, \qquad (3)$$

where b_i stands for the transformation of $\mathbf{B}_i$ and $\mathrm{diag}(\cdot)$ yields a diagonal matrix with the elements of $\boldsymbol{s}_i$. Our abstract layout is more friendly to collect and easier to edit, allowing for versatile interactive user control.

3.2. Spatially Disentangled Radiance Fields

Object Representation. Neural radiance field (NeRF) [28] $\mathrm{F}(\boldsymbol{x}, \boldsymbol{v}) \rightarrow (\boldsymbol{c}, \sigma)$ regresses color $\boldsymbol{c} \in \mathbb{R}^3$ and volume density $\sigma \in \mathbb{R}$ from coordinate $\boldsymbol{x} \in \mathbb{R}^3$ and viewing direction $\boldsymbol{v} \in \mathbb{S}^2$, parameterized with multi-layer perceptron (MLP)

networks. Recent attempts propose to condition NeRF with a latent code $\boldsymbol{z}$, resulting in their generative forms [5, 39], $\mathrm{G}(\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{z}) \rightarrow (\boldsymbol{c}, \sigma)$, to achieve 3D-aware object synthesis.

Since we use the layout as an internal representation, it naturally disentangles the whole scene into several objects. We can leverage multiple individual generative NeRFs to model different objects, but it can easily lead to an overwhelmingly large number of models and poor training efficiency. To alleviate this issue, we propose to infer generative object radiance field in the *canonical* space, to allow weight sharing among objects:

$$(\boldsymbol{c}_i, \sigma_i) = \mathrm{G}_{\mathrm{obj}}(\mathrm{b}_i^{-1}(\gamma(\boldsymbol{x})), \boldsymbol{z}_i), \qquad (4)$$

where $\gamma(\cdot)$ is the position encoding function that transforms input into Fourier features. The object generator $\mathrm{G}_{\mathrm{obj}}(\cdot)$ infers each object independently, resulting in *spatially disentangled generative radiance fields*. Note that $\mathrm{G}_{\mathrm{obj}}(\cdot)$ is not conditioned on the viewing direction $\boldsymbol{v}$ because the upsampler of our neural renderer can learn the view-dependent effects similar to previous works [4, 15, 59] (Sec. 3.3).

Spatial Condition. Although object bounding boxes are used as a prior, their latents are still randomly sampled regardless of their spatial configuration, leading to illogical arrangements. To synthesize scene images and infer object radiance fields with proper semantics, we adopt the location and scale of each object as a condition for the generator to encode more consistent intrinsic properties, *i.e.*, shape and category. To this end, we simply modify Eq. (4) by concatenating the latent code with the Fourier features of object location and scale:

$$(\boldsymbol{c}, \sigma) = \mathrm{G}_{\mathrm{obj}}(\mathrm{b}_i^{-1}(\gamma(\boldsymbol{x})), \mathrm{concat}(\boldsymbol{z}_i, \gamma(\boldsymbol{t}_i), \gamma(\boldsymbol{s}_i))). \quad (5)$$

Therefore, semantic clues are injected into the layout in an unsupervised manner, without explicit category annotation.

3

Background Representation. Unlike objects, the background radiance field is only evaluated in the global space. Considering that the background encodes lots of high-frequency signals, we include the viewing direction $\boldsymbol{v}$ to help background generator $G_{bg}(\cdot)$ to be able to learn such details. The background generation can be formulated as:

$$(\boldsymbol{c}_{bg}, \sigma_{bg}) = G_{bg}(\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{z}_{bg}). \tag{6}$$

3.3. Efficient Rendering Pipeline

As aforementioned, we use spatial-disentangled radiance fields to represent scenes. However, naïve point sampling solutions can lead to prohibitive computational overhead when rendering multiple radiance fields. Considering the independence of objects' radiance fields, we can achieve much more efficient rendering by only focusing on the valid points within the bounding boxes.

Ray-Box Intersection in Canonical Space. Similar to NeRF [28], we use the pinhole camera model to perform ray casting. For each object, the points on the rays can be sampled at adaptive depths rather than fixed ones since the bounding box provides clues about where the object locates. Specifically, the cast rays $\mathcal{R} = \{\mathbf{r}_j | j \in [1, S^2]\}$ in a resolution S are transformed into the canonical object coordinate system. Then, Ray-AABB (axis-aligned bounding box) intersection algorithm is applied to calculate the adaptive near and far depth $(d_{j,l,n}, d_{j,l,f})$ of the intersected segment between the ray $\mathbf{r}_j$ and the l-th box $\mathbf{B}_l$. After that, N_d points are sampled equidistantly in the interval $[d_{j,l,n}, d_{j,l,f}]$. It is worth noting that we maintain an intersection matrix $\mathbf{M}$ of size $N \times S^2$, whose elements indicate if this ray intersects with the box. With $\mathbf{M}$, we select only valid points to infer, which can greatly reduce the rendering cost.

Background Point Sampling. We adopt different background sampling strategies depending on the dataset. In general, we do fixed depth sampling for bounded backgrounds in indoor scenes and inherit the inverse parametrization of NeRF++ [65] for complex and unbounded outdoor scenes, which uniformly samples background points in an inverse depth range. More details can be founded in the *Supplementary Materials*.

Composition and Volume Rendering. In our approach, objects are always assumed to be in front of the background. So objects and background can be rendered independently first and composited thereafter. For a ray $\mathbf{r}_j$ intersecting with n_j ($n_j \geq 1$) boxes, its sample points $\mathcal{X}_j = \{\boldsymbol{x}_{j,k} | k \in [1, n_j N_d]\}$ can be easily obtained from the depth range and the intersection matrix $\mathbf{M}$. Since rendering should consider inter-object occlusions, we sort the points $\mathcal{X}$ by depth, resulting in an ordered point set $\mathcal{X}_j^s = \{\boldsymbol{x}_{j,s_k} | s_k \in [1, n_j N_d], d_{j,s_k} \leq d_{j,s_{k+1}}\}$, where d_{j,s_k} denotes the depth of point $\boldsymbol{x}_{j,s_k}$. With color $\boldsymbol{c}(\boldsymbol{x}_{j,s_k})$ and density $\sigma(\boldsymbol{x}_{j,s_k})$ of the ordered set inferred with $G_{obj}(\cdot)$ by Eq. (5), the corresponding pixel $\mathbf{f}(\mathbf{r}_j)$ is calculated as:

$$\mathbf{f}(\mathbf{r}_j) = \sum_{k=1}^{n_j N_d} T_{j,k} \alpha_{j,k} \boldsymbol{c}(\boldsymbol{x}_{j,s_k}), \tag{7}$$

$$T_{j,k} = \exp(-\sum_{o=1}^{k-1} \sigma(\boldsymbol{x}_{j,s_k})\delta_{j,s_o}), \tag{8}$$

$$\alpha_{j,k} = 1 - \exp(-\sigma(\boldsymbol{x}_{j,s_k})\delta_{j,s_k}). \tag{9}$$

For any ray that does not intersect with boxes, its color and density are set to 0 and $-\infty$, respectively. So that the foreground object map $\mathbf{F}$ can be formulated as:

$$\mathbf{F}_j = \begin{cases} \mathbf{f}(\mathbf{r}_j), & \text{if } \exists\, m \in \mathbf{M}_{:,j}, m \text{ is true,} \\ 0, & \text{else.} \end{cases} \tag{10}$$

Since the background points are sampled at a fixed depth, we can directly adopt Eq. (6) to evaluate background points in the global space without sorting. And the background map $\mathbf{N}$ can also be obtained by volume rendering similar to Eq. (7). Finally, $\mathbf{F}$ and $\mathbf{N}$ are alpha-blended into the final image $\mathbf{I}_n$ with alpha extracted from Eq. (9):

$$\mathbf{I}_n = \mathbf{F} + \prod_{k=1}^{n_j N_d} (1 - \alpha_{j,k}) \odot \mathbf{N}. \tag{11}$$

Although our rendering pipeline efficiently composites multiple radiance fields, it still suffers from slow performance when rendering high-resolution images. To mitigate this issue, we render a high-dimensional feature map instead of a 3-channel color in a smaller resolution, followed by a StyleGAN2-like architecture that upsamples the feature map to the target resolution.

3.4. Local & Global Discrimination

Like other GAN-based approaches, discriminators play a crucial role in training. Previous attempts for 3D-aware scene synthesis [8, 31] only adopt scene-level discriminators to critique between rendered scenes and real captures. However, such a scene discriminator pays more attention to the global coherence of the whole scene, weakening the supervision for individual objects. Given that each object, especially those far from the camera, occupies a small portion of the rendered frame, the scene discriminator provides weak learning signal to its radiance field, leading to inadequate training and poor object quality. Besides, the scene discriminator shows only minimal capability in disentangling objects and background, allowing the background generator G_{bg} to overfit the whole scene easily.

Similar to [12], we propose to add an extra object discriminator for local discrimination, leading to better object-level supervision. Specifically, with the 3D layout $\mathbf{B}_i$ spatially disentangling different objects, we project them into

2D space as $\mathbf{B}_i^{2D}$ to extract object patches $\mathcal{P}_\mathbf{I} = \{\mathbf{P}_i | \mathbf{P}_i = \mathrm{crop}(\mathbf{I}, \mathbf{B}_i^{2D})\}$ from synthesized and real scenes images with simple cropping. The object patches are fed into the object discriminator after being scaled to a uniform size. We find that it significantly helps synthesize realistic objects and benefits the disentanglement between objects and the background. More details about our object discrimination are included in the *Supplementary Materials*.

3.5. Training and Inference

Training Objectives. The whole generation process is formulated as $\mathbf{I}_f = \mathrm{G}(\mathcal{B}, \mathcal{Z}, \xi)$, where the generator $\mathrm{G}(\cdot)$ receives a layout $\mathcal{B}$, a latent code set $\mathcal{Z}$ independently sampled from distribution $\mathcal{N}(0, 1)$ to control objects, and a camera pose ξ sampled from a prior distribution p_ξ to synthesize the image $\mathbf{I}_f$. During training, $\mathcal{B}$, $\mathcal{Z}$, ξ are randomly sampled, and the real image $\mathbf{I}_r$ is sampled from the dataset. Besides the generator, we employ the scene discriminator $\mathrm{D}_s(\cdot)$ to guarantee the global coherence of the rendering and the object discriminator $\mathrm{D}_{\mathrm{obj}}(\cdot)$ on individual objects for local discrimination. Generators and discriminators are jointly trained as:

$$\min \mathcal{L}_G = \mathbb{E}[f(-\mathrm{D}_s(\mathbf{I}_f))] + \lambda_1 \mathbb{E}[f(-\mathrm{D}_{\mathrm{obj}}(\mathcal{P}_{\mathbf{I}_f}))], \quad (12)$$

$$\min \mathcal{L}_D = \mathbb{E}[f(-\mathrm{D}_s(\mathbf{I}_r))] + \mathbb{E}[f(\mathrm{D}_s(\mathbf{I}_f))] \quad (13)$$
$$+ \lambda_1 (\mathbb{E}[f(-\mathrm{D}_{\mathrm{obj}}(\mathcal{P}_{\mathbf{I}_r})]) + \mathbb{E}[f(\mathrm{D}_{\mathrm{obj}}(\mathcal{P}_{\mathbf{I}_f}))])$$
$$+ \lambda_2 ||\nabla_{\mathbf{I}_r} \mathrm{D}_s(\mathbf{I}_r)||_2^2 + \lambda_3 \nabla_{\mathcal{P}_{\mathbf{I}_r}} \mathrm{D}_{\mathrm{obj}}(\mathcal{P}_{\mathbf{I}_r})||_2^2),$$

where $f(t) = \log(1 + \exp(t))$ is the softplus function, and $\mathcal{P}_{\mathbf{I}_r}$ and $\mathcal{P}_{\mathbf{I}_f}$ are the extracted object patches of synthesized image $\mathbf{I}_f$ and real image $\mathbf{I}_r$, respectively. λ_1 stands for the loss weight of the object discriminator. The last two terms in Eq. (13) are the gradient penalty regularizers of both discriminators, with λ_2 and λ_3 denoting their weights.

Inference. Besides high-quality scene generation, our method naturally supports object editing by manipulating the layout prior as shown in Fig. 1. Various applications are shown in Sec. 4.3. In particular, ray marching at a small resolution (64) may cause aliasing especially when moving the objects. We adopt supersampling anti-aliasing (SSAA) [42] to perform ray marching at a temporary higher resolution (128) and downsample the feature map to the original resolution before the upsampler. This strategy is used only for object synthesis, and we do not change the background resolution during inference.

4. Experiments

4.1. Settings

Datasets. We evaluate *DisCoScene* on three multi-object scene datasets, including CLEVR [21], 3D-FRONT [10, 11], and WAYMO [47]. CLEVR is a diagnostic multi-object dataset. We use the official script [21] to render scenes

with 2 and random primitives. Our CLEVR dataset consists of $80K$ samples in 256×256 resolution. 3D-FRONT is an indoor scene dataset, containing a collection of $6.8K$ houses with $140K$ rooms. We obtain $4K$ bedrooms after filtering out rooms with uncommon arrangements or unnatural sizes and use BlenderProc to render 20 images per room from random camera positions, resulting in a total of $80K$ images. WAYMO is a large-scale autonomous driving dataset with $1K$ video sequences of outdoor scenes. Six images are provided for each frame, and we only keep the front view. We also apply heuristic rules to filter out small and noisy cars and collect a subset of $70K$ images. Because the width is always larger than height on WAYMO, we adopt the black padding to make images square, similar with StyleGAN2 [25]. More details about data preprocessing and rendering are included in *Supplementary Materials*.

Baselines. We compare with both 2D and 3D GANs. For 2D, we compare with StyleGAN2 [25] on image quality. As for 3D, we compare with EpiGRAF [45], VolumeGAN [59], and EG-3D [4] on object generation, and GIRAFFE [31], GSN [8] on scene generation. We use the baseline models either released along with their papers or official implementations to train on our data.[*]

Implementation Details. We use the same architecture and parameters of the mapping network from StyleGAN2 [25]. For object generator $\mathrm{G}_{\mathrm{obj}}(\cdot)$ and background generator $\mathrm{G}_{\mathrm{bg}}(\cdot)$, we use 8 and 4 Modulated Fully-Connected layers (ModFCs) with 256 and 128 channels, respectively. Ray casting is performed on 64×64 and the feature map is rendered to image with neural renderer. The progressive training strategy from PG-GAN [22] is adopted for better image quality and multi-view consistency. Discriminators $\mathrm{D}_s(\cdot)$ and $\mathrm{D}_{\mathrm{obj}}(\cdot)$ both share the similar architecture of StyleGAN2 but with only half channels. Practically, the resolution of $\mathrm{D}_{\mathrm{obj}}(\cdot)$ is always $1/2$ on WAYMO or $1/4$ on CLEVR and 3D-FRONT of $\mathrm{D}_s(\cdot)$. All our models are trained on $8\times$ V100/A100 GPUs with a batch size of 64. λ_1 is set to 1 to balance object and scene discriminators. λ_2 and λ_3 are set to 1 to maintain training stability. Unless specified, other hyperparamters are same as StyleGAN2. More details about network architecture and training can be found in *Supplementary Materials*.

4.2. Main Results

Qualitative Comparison. Fig. 2 presents the synthesized images in a resolution of 256×256 of our method and baselines on all the datasets. We compare our method on explicit camera control and object editing with baselines.

GSN and EG3D, with a single radiance field, can manipulate the global camera of the synthesized images. GSN highly depends on the training camera trajectories. Thus

---

[*]We fail to train GSN on CLEVR and WAYMO with the official implementation, hence we do not report the quantitative results.
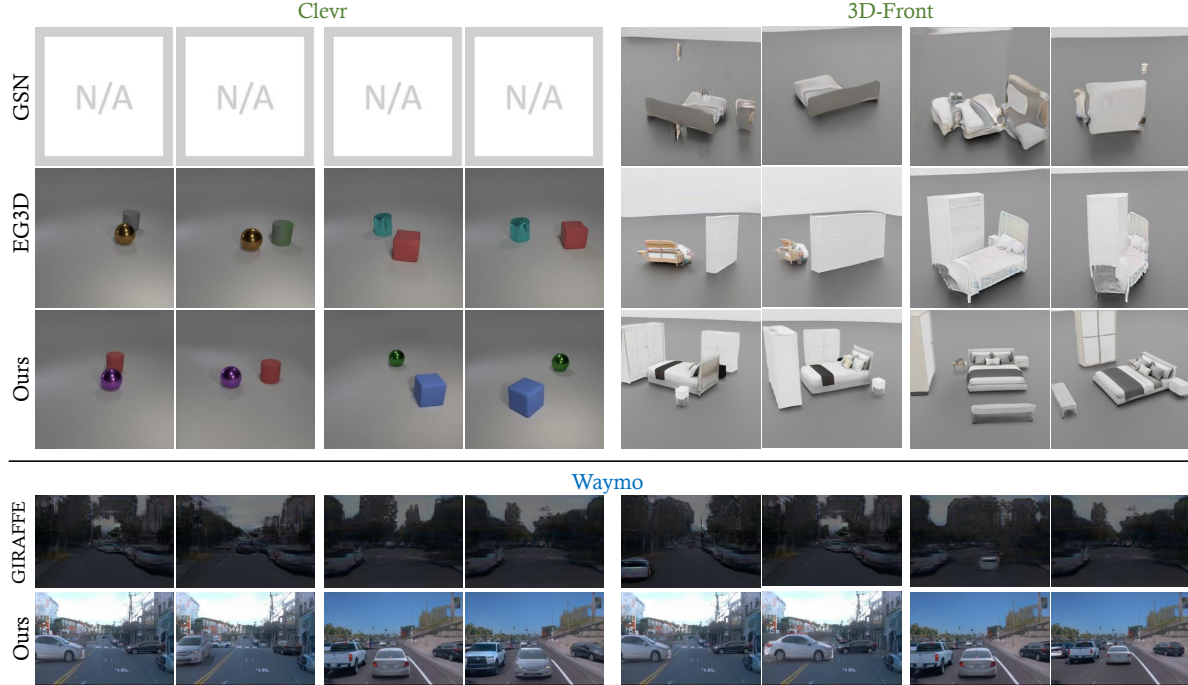
Figure 2. **Qualitative comparison between *DisCoScene* and baselines**. Explicit camera rotation is evaluated on CLEVR and 3D-FRONT. Object rotation (left) and object translation (right) are evaluated on WAYMO. All images are in 256×256 resolution.

Table 2. **Quantitative comparisons on different datasets.** FID, KID ($\times 10^3$) are reported as the evaluation metrics. TR. and INF. denote training and inference costs, evaluated in *V100 days* and *ms/image* (single V100 over $1K$ samples), respectively. Note that we highlight the best results among 3D-aware models.

Model	CLEVR				3D-FRONT		WAYMO	
	FID ↓	KID ↓	TR. ↓	INF. ↓	FID ↓	KID ↓	FID ↓	KID ↓
StyleGAN2 [25]	4.5	3.0	13.3	44	12.5	4.3	15.1	8.3
EpiGRAF [45]	10.4	8.3	16.0	114	107.2	102.3	27.0	26.1
VolumeGAN [59]	7.5	5.1	15.2	90	52.7	38.7	29.9	18.2
EG3D [4]	4.1	12.7	25.8	**55**	19.7	13.5	26.0	45.4
GIRAFFE [31]	78.5	61.5	**5.2**	62	56.5	46.8	175.7	212.1
GSN [8]	—	—	—	—	130.7	87.5	—	—
DisCoScene	**3.5**	**2.1**	18.1	95	**13.8**	**7.4**	**16.0**	**8.4**

in our setting where the camera positions are randomly sampled, it suffers from the poor image quality and multi-view consistency. As for EG3D, although it converges on the datasets, the object fidelity are lower than our method. On CLEVR with a narrow camera distribution, the results of EG3D are inconsistent. In the first example, the color of the cylinder changes from gray to green across different views. Meanwhile, our method learns better 3D structure of the objects and achieves better camera control. On the challenging WAYMO dataset, it is difficult to encode huge street scenes within a single generator, thus we train GI-RAFFE and our *DisCoScene* in the camera space to evaluate object editing. GIRAFFE struggles to generate realistic results and, while manipulating objects, their geometry

and appearance are not preserved well. Our approach is capable of handling these complicated scenarios with good variations. Wherever the object is placed and regardless of how the rotation is carried out, the synthesized objects are substantially better and more consistent than GIRAFFE. It demonstrates the effectiveness of our spatially disentangle radiance fields built upon the layout prior.

Quantitative Comparison. Tab. 2 reports the quantitative metrics on the quality of results, including FID [18] and KID [3]. All metrics are calculated between $50K$ generated samples and all real images. *DisCoScene* consistently outperforms baselines with significant improvement on all datasets. Besides, training cost in *V100 days* and testing cost in *ms/image* (on a single V100 over $1K$ samples) are

also included to reflect the efficiency of our model. Note that the inference cost of 3D-aware models is evaluated on generating radiance fields rather than images. In such a case, EG3D and EpiGRAF are not fast as excepted due to the heavy computation on tri-planes. With comparable training and testing cost, it even achieves similar level of image quality with state-of-the-art 2D GAN baselines, *e.g.*, StyleGAN2 [25], while allowing for explicit camera control and object editing that are otherwise challenging.

4.3. Controllable Scene Generation

The layout prior in our model enables versatile user controls of scene objects. In what follows, we evaluate the flexibility and effectiveness of our model through various 3D manipulation applications in different datasets. Examples are shown in Fig. 3 and more results can be found in *Supplementary Materials*.

Rearranging Objects. We can transform bounding boxes **B** to rearrange (rotation and translation) the objects in the scenes without affecting their appearance. Transforming shapes in CLEVR, furniture in 3D-FRONT, and cars in WAYMO all show consistent results. In particular, rotating symmetric shapes (*i.e.*, spheres and cylinders) in CLEVR shows little changes, suggesting desired multi-view consistency. Our model can properly handle mutual occlusion. Take the blue cube from CLEVR as example (1-st row of Fig. 3), our model can produce new occlusions between it and the grey cylinder and generate high-quality renderings.

Removing and Cloning Objects. Users can update the layout by removing or cloning bounding boxes. Our method seamlessly removes objects with the background inpainted realistically, even without training on any pure background, including the challenging dataset of WAYMO (3-rd row of Fig. 3). Object cloning is also naturally supported, by copying and pasting a box to a new location in the layout.

Restyling Objects. Although appearance and shape are not explicitly modeled by the latent code, we can reuse the encoded hierarchical knowledge to perform object restyling. Like [15, 41, 60], we arbitrarily sample latent codes and perform style-mixing on different layers to achieve independent control over appearance and shape. Fig. 3 presents the restyling results on certain objects, *i.e.*, the front cylinder in CLEVR, the bed in 3D-FRONT, and the left car in WAYMO.

Camera Movement. Explicit camera control is also permitted. Even for CLEVR that is trained on very limited camera ranges, we can rotate the camera up to an extreme side view. Our model also produces consistent results when rotating the camera on 3D-FRONT (2-nd row of Fig. 3).

4.4. Ablation Study

We ablate main components of our approach to better understand their individual contributions. In addition to the FID score that measures the quality of the entire image, we

Table 3. **Ablation analysis** of object discriminator (D_{obj}).

		CLEVR	3D-FRONT	WAYMO
FID	*w/o* D_{obj}	5.0	18.6	19.5
	w/ D_{obj}	**3.5**	**13.8**	**16.0**
FID_{obj}	*w/o* D_{obj}	19.1	33.7	95.1
	w/ D_{obj}	**5.6**	**19.5**	**16.3**

Table 4. **Ablation analysis** of spatial condition (S-Cond).

	FID		FID_{obj}	
	w/ S-Cond	*w/o* S-Cond	*w/* S-Cond	*w/o* S-Cond
3D-FRONT	**13.8**	15.2	**19.5**	23.2

also provide another metric FID_{obj} to measure the quality of individual objects. Specifically, we use the projected 2D boxes to crop objects from the synthesized images and then perform FID evaluation against the ones from real images.

Object Discriminator. The object discriminator D_{obj} plays a crucial role in synthesizing realistic objects, as evaluted in Tab. 3. Obviously, the object fidelity is significantly improved across all datasets with D_{obj}. Also, the quality of the whole scene generation is improved as well, contributed by better objects. Fig. 5 visually shows that our method can successfully disentangle objects from the background with the help of object discriminator. Although the baseline model is able to disentangle objects on 3D-FRONT from simple background to certain extent, the background suffers from the entanglement with objects, resulting in obvious artifacts as well as illogical layout. On more challenging datasets like WAYMO, the complex backgrounds make the disentanglement even more difficult, so that the background model easily overfits the whole scene as a single radiance field. Thanks to the object discriminator, our full model benefits from object supervision, leading to better disentanglement, even without seeing a pure background image.

Spatial Condition. To analyze how spatial condition (S-Cond) affects the quality of generation, we compare results with models trained with and without S-Cond on 3D-FRONT (Fig. 4a). For example, our full model consistently infers beds at the center of rooms, while the baseline predicts random items like tables or nightstands that rarely appear in the middle of bedrooms. These results demonstrate that spatial condition can assist the generator with appropriate semantics from simple layout priors. Note that this correlation between spatial configurations and object semantics is automatically emerged without any supervision. We also numerically compare the image quality on these two models in Tab. 4, which shows that S-Cond also achieves better image quality at both scene- and object-level, because more proper semantics are more in line with the native distribution of real images.

Supersampling Anti-Aliasing. We adopt a simple supersampling (SSAA) strategy to reduce edge aliasing by sam-
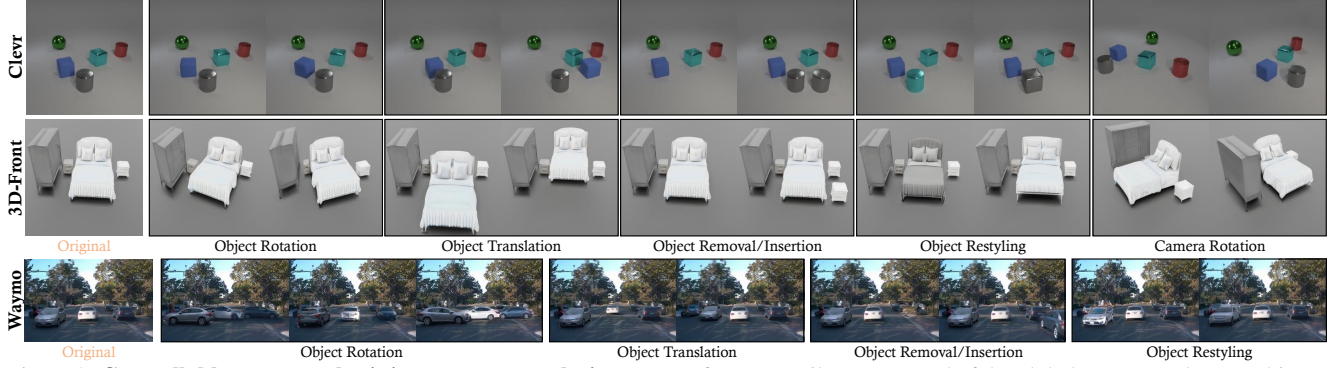
Figure 3. **Controllable scene synthesis in** 256×256 **resolution.** We perform versatile user control of the global camera and scene objects, such as rearrangement, removal, insertion, and restyling. Editings are performed on the original generations (the left one of each row).
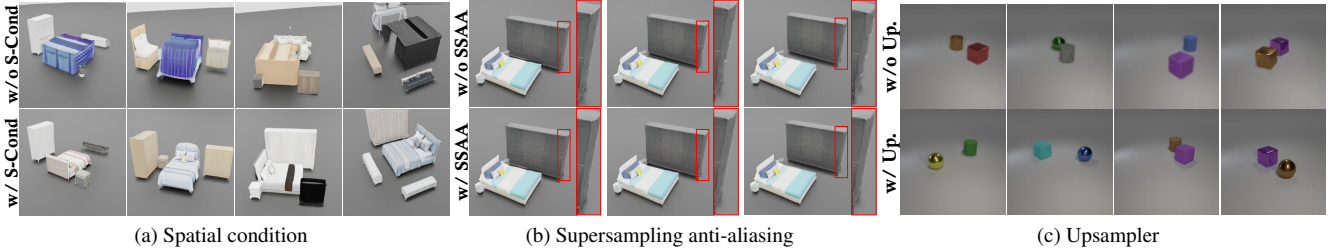

(a) Spatial condition   (b) Supersampling anti-aliasing   (c) Upsampler

Figure 4. **Qualitative comparison for ablations** on spatial condition (S-Cond), supersampling anti-alising (SSAA), and upsampler (Up.).
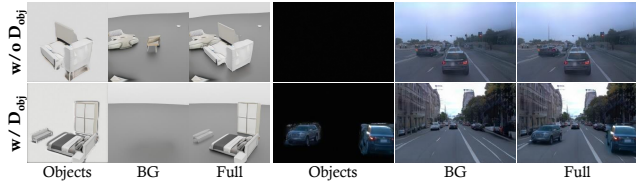

Figure 5. **Ablation on scene disentanglement**. We independently infer objects and background to show the quality of scene disentanglement with regard to object discriminator D_{obj}.

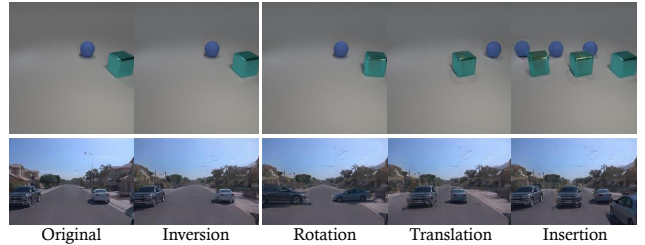Original   Inversion   Rotation   Translation   Insertion

Figure 6. **Real image inversion and editing**.

pling more points during inference (Sec. 3.5). Thanks to our efficient object point sampling, doubling the resolution of foreground points keeps a similar inference speed (105 *ms/image*), comparable with original speed (95 *ms/image*). Results with different sampling points are shown in Fig. 4b. Taking the right boundary of the cabinet as an example (see the zoom-in insets for better visualization), when the cabinet is moved, SSAA achieves more consistent boundary compared with the jaggy one in the baseline.

Neural Renderer for Shadow. We adopt the StyleGAN2-like neural renderer to boost the rendering efficiency (Sec. 3.3). Besides the low computational cost, the added capacity of the neural renderer also brings better implicit modeling of realistic lighting effects such as shadowing. Therefore, without handling the shadowing effect in our rendering pipeline, our model can still synthesize high-quality shaodws on datasets such as CLEVR (Fig. 4c). This is because the large receptive field brought by 3×3 convolutions and upsampler blocks make the neural renderer be aware of the object locations and progressively add shadows to the low resolution features rendered from radiance fields.

5. Discussion and Conclusion

Real Image Editing. Fig. 6 shows that it is possible to embed a real image into the latent space of our pretrained model using pivotal tuning inversion (PTI) [38]. Besides reconstruction, all object manipulation operations are supported to edit the image. As one of the very first steps towards 3D scene editing from a single image, we believe that our method proves a promising venue and can inspire future research efforts along this direction.

Limitations and Future Work. Our model requires the abstract layout prior as the input. For in-the-wild datasets, we need monocular 3D object detector [53] to infer pseudo layouts. While existing approaches attempt to learn the layout in an end-to-end manner, they struggle to generalize to complex scenes consisting of multiple objects. So it would be interesting to explore 3D layout estimation for complex scenes and combine with our approach end-to-end. Also, although our work shows significant improvement over existing 3D-aware scene generators, it is still challenging to learn on the street scenes in the global space due to the

limited model capacity. Large-scale NeRFs [50, 56] might be one potention solutions.

Conclusion. This work presents *DisCoScene*, a method for controllable 3D-aware scene synthesis on challenging datasets. By taking spatially disentangled radiance fields as the representation based on a very abstract layout prior, our method is able to generate high-fidelity scene images and allows for versatile object-level editing.

References

[1] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Int. Conf. Comput. Vis.*, pages 5855–5864, 2021. 1

[2] Daniel Bear, Chaofei Fan, Damian Mrowca, Yunzhu Li, Seth Alter, Aran Nayebi, Jeremy Schwartz, Li F Fei-Fei, Jiajun Wu, Josh Tenenbaum, et al. Learning physical graph representations from visual scenes. *Adv. Neural Inform. Process. Syst.*, 2020. 2

[3] Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018. 6

[4] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022. 1, 2, 3, 5, 6, 11, 12

[5] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 1, 2, 3

[6] Steve Cunningham and Michael J Bailey. Lessons from scene graphs: using scene graphs to teach hierarchical modeling. *Computers & Graphics*, 2001. 3

[7] Yu Deng, Jiaolong Yang, Jianfeng Xiang, and Xin Tong. Gram: Generative radiance manifolds for 3d-aware image generation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022. 2

[8] Terrance DeVries, Miguel Angel Bautista, Nitish Srivastava, Graham W. Taylor, and Joshua M. Susskind. Unconstrained scene generation with locally conditioned radiance fields. In *Int. Conf. Comput. Vis.*, 2021. 1, 2, 4, 5, 6, 12

[9] Dave Epstein, Taesung Park, Richard Zhang, Eli Shechtman, and Alexei A. Efros. Blobgan: Spatially disentangled scene representations. *Eur. Conf. Comput. Vis.*, 2022. 2

[10] Huan Fu, Bowen Cai, Lin Gao, Ling-Xiao Zhang, Jiaming Wang, Cao Li, Qixun Zeng, Chengyue Sun, Rongfei Jia, Binqiang Zhao, et al. 3d-front: 3d furnished rooms with layouts and semantics. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 5, 12

[11] Huan Fu, Rongfei Jia, Lin Gao, Mingming Gong, Binqiang Zhao, Steve Maybank, and Dacheng Tao. 3d-future: 3d furniture shape with texture. *Int. J. Comput. Vis.*, 2021. 5, 12

[12] Raghudeep Gadde, Qianli Feng, and Aleix M Martinez. Detail me more: Improving gan's photo-realism of complex scenes. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 4

[13] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. *arXiv preprint arXiv:2209.11163*, 2022. 1

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Adv. Neural Inform. Process. Syst.*, 2014. 2

[15] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. *arXiv preprint arXiv:2110.08985*, 2021. 1, 2, 3, 7

[16] Michelle Guo, Alireza Fathi, Jiajun Wu, and Thomas Funkhouser. Object-centric neural scene rendering. *arXiv preprint arXiv:2012.08503*, 2020. 1

[17] Kamal Gupta, Justin Lazarow, Alessandro Achille, Larry S Davis, Vijay Mahadevan, and Abhinav Shrivastava. Layouttransformer: Layout generation and completion with self-attention. In *Int. Conf. Comput. Vis.*, 2021. 1

[18] 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 *Adv. Neural Inform. Process. Syst.*, 2017. 6

[19] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 2

[20] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 3

[21] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 5, 12

[22] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *Int. Conf. Learn. Represent.*, 2018. 2, 5

[23] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. In *Adv. Neural Inform. Process. Syst.*, 2021. 2

[24] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2

[25] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2, 5, 6, 7, 11

[26] Jianan Li, Jimei Yang, Aaron Hertzmann, Jianming Zhang, and Tingfa Xu. Layoutgan: Generating graphic layouts with wireframe discriminators. *arXiv preprint arXiv:1901.06767*, 2019. 1

[27] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 1

[28] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Eur. Conf. Comput. Vis.*, 2020. 1, 3, 4

[29] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *Int. Conf. Comput. Vis.*, 2019. 2

[30] Thu H Nguyen-Phuoc, Christian Richardt, Long Mai, Yongliang Yang, and Niloy Mitra. Blockgan: Learning 3d object-aware scene representations from unlabelled images. *Adv. Neural Inform. Process. Syst.*, 2020. 2

[31] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 1, 2, 4, 5, 6, 12

[32] Roy Or-El, Xuan Luo, Mengyi Shan, Eli Shechtman, Jeong Joon Park, and Ira Kemelmacher-Shlizerman. Stylesdf: High-resolution 3d-consistent image and geometry generation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022. 1, 2

[33] Julian Ost, Fahim Mannan, Nils Thuerey, Julian Knodt, and Felix Heide. Neural scene graphs for dynamic scenes. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2856–2865, 2021. 1, 3

[34] Xingang Pan, Xudong Xu, Chen Change Loy, Christian Theobalt, and Bo Dai. A shading-guided generative implicit model for shape-accurate 3d-aware image synthesis. In *Adv. Neural Inform. Process. Syst.*, 2021. 2

[35] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 1

[36] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2

[37] Sida Peng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9054–9063, 2021. 1

[38] Daniel Roich, Ron Mokady, Amit H Bermano, and Daniel Cohen-Or. Pivotal tuning for latent-based editing of real images. *ACM Trans. Graph.*, 2022. 8

[39] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *Adv. Neural Inform. Process. Syst.*, 2020. 1, 2, 3

[40] Katja Schwarz, Axel Sauer, Michael Niemeyer, Yiyi Liao, and Andreas Geiger. Voxgraf: Fast 3d-aware image synthesis with sparse voxel grids. *Adv. Neural Inform. Process. Syst.*, 2022. 2

[41] Yujun Shen, Ceyuan Yang, Xiaoou Tang, and Bolei Zhou. Interfacegan: Interpreting the disentangled face representation learned by gans. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020. 7

[42] A. Sherrod. *Game Graphic Programming*. Course Technology PTR game development series. Course Technology/Charles River Media/Cengage Learning, 2008. 5

[43] Zifan Shi, Yujun Shen, Jiapeng Zhu, Dit-Yan Yeung, and Qifeng Chen. 3d-aware indoor scene synthesis with depth priors. In *Eur. Conf. Comput. Vis.*, 2022. 2

[44] Zifan Shi, Yinghao Xu, Yujun Shen, Deli Zhao, Qifeng Chen, and Dit-Yan Yeung. Improving 3d-aware image synthesis with a geometry-aware discriminator. *Adv. Neural Inform. Process. Syst.*, 2022. 1, 2

[45] Ivan Skorokhodov, Sergey Tulyakov, Yiqun Wang, and Peter Wonka. Epigraf: Rethinking training of 3d gans. In *Adv. Neural Inform. Process. Syst.*, 2022. 1, 2, 5, 6, 11

[46] Henry Sowizral. Scene graphs in the new millennium. *IEEE Computer Graphics and Applications*, 2000. 3

[47] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2, 5, 12

[48] Zhentao Tan, Dongdong Chen, Qi Chu, Menglei Chai, Jing Liao, Mingming He, Lu Yuan, Gang Hua, and Nenghai Yu. Efficient semantic image synthesis via class-adaptive normalization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2021. 2

[49] Zhentao Tan, Qi Chu, Menglei Chai, Dongdong Chen, Jing Liao, Qiankun Liu, Bin Liu, Gang Hua, and Nenghai Yu. Semantic probability distribution modeling for diverse semantic image synthesis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2022. 2

[50] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P Srinivasan, Jonathan T Barron, and Henrik Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022. 9

[51] Zhuowen Tu, Xiangrong Chen, Alan L Yuille, and Song-Chun Zhu. Image parsing: Unifying segmentation, detection, and recognition. *IJCV*, 2005. 2

[52] Jianyuan Wang, Ceyuan Yang, Yinghao Xu, Yujun Shen, Hongdong Li, and Bolei Zhou. Improving gan equilibrium by raising spatial awareness. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022. 2

[53] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Fcos3d: Fully convolutional one-stage monocular 3d object detection. In *Int. Conf. Comput. Vis. Worksh.*, 2021. 8

[54] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 2

[55] Qianyi Wu, Xian Liu, Yuedong Chen, Kejie Li, Chuanxia Zheng, Jianfei Cai, and Jianmin Zheng. Object-compositional neural implicit surfaces. In *Eur. Conf. Comput. Vis.*, 2022. 1

[56] Yuanbo Xiangli, Linning Xu, Xingang Pan, Nanxuan Zhao, Anyi Rao, Christian Theobalt, Bo Dai, and Dahua Lin. Bungeenerf: Progressive neural radiance field for extreme multi-scale scene rendering. In *Eur. Conf. Comput. Vis.*, 2022. 9

[57] Linning Xu, Yuanbo Xiangli, Anyi Rao, Nanxuan Zhao, Bo Dai, Ziwei Liu, and Dahua Lin. Blockplanner: City block generation with vectorized graph representation. In *Int. Conf. Comput. Vis.*, 2021. 1

[58] Xudong Xu, Xingang Pan, Dahua Lin, and Bo Dai. Generative occupancy fields for 3d surface-aware image synthesis. In *Adv. Neural Inform. Process. Syst.*, 2021. 2

[59] Yinghao Xu, Sida Peng, Ceyuan Yang, Yujun Shen, and Bolei Zhou. 3d-aware image synthesis via learning structural and textural representations. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022. 1, 2, 3, 5, 6, 11

[60] Yinghao Xu, Yujun Shen, Jiapeng Zhu, Ceyuan Yang, and Bolei Zhou. Generative hierarchical features from synthesizing images. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 7

[61] Yang Xue, Yuheng Li, Krishna Kumar Singh, and Yong Jae Lee. Giraffe hd: A high-resolution 3d-aware generative model. In *CVPR*, 2022. 2

[62] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Learning object-compositional neural radiance field for editable scene rendering. In *Int. Conf. Comput. Vis.*, pages 13779–13788, 2021. 1

[63] Ceyuan Yang, Yujun Shen, and Bolei Zhou. Semantic hierarchy emerges in deep generative representations for scene synthesis. *IJCV*, 2021. 2

[64] Chen Zhang, Yinghao Xu, and Yujun Shen. Decorating your own bedroom: Locally controlling image generation with generative adversarial networks. *IEEE Conf. Comput. Vis. Pattern Recog. Worksh.*, 2021. 2

[65] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 4, 11

[66] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Int. Conf. Comput. Vis.*, 2017. 2

[67] Jun-Yan Zhu, Zhoutong Zhang, Chengkai Zhang, Jiajun Wu, Antonio Torralba, Joshua B. Tenenbaum, and William T. Freeman. Visual object networks: Image generation with disentangled 3D representations. In *Adv. Neural Inform. Process. Syst.*, 2018. 2

A1. Implementation Details of *DisCoScene*

Background with NeRF++ [65]. The outdoor datasets, *i.e.* WAYMO, have unbounded backgrounds. It is insufficient to model the whole scene in the image within a fixed bounding box. Therefore, we inherit the inverse parametrization of NeRF++ to model the background in WAYMO:

$$\boldsymbol{x} = (x/r, y/r, z/r), \quad (14)$$

where $r = ||\boldsymbol{x}||^2$. The background points are uniformly sampled in an inverse depth range of $[1/R, 0)$ where $R = 2.0$ denotes the starting depth of the background.

Constant Latents for Upsampler. We adopt similar architecture and parameters of the synthesis network from StyleGAN2 [25] as the upsampler for the rendered 2D feature map. Note that since our model handles multiple radiance fields, different spatial locations of the convolution feature maps should be modulated by different $\boldsymbol{w}$ codes belonging to specific objects, making it costly to upsample the feature map. Thus we disable the spatial-aware modulation by setting $\boldsymbol{w}$ as a constant tensor with value 1, which significantly reduces the computation overhead.

A2. Implementation Details of Baselines

Because of the wildly divergent data distribution, the training parameters vary greatly on different datasets. Tab. 5 and Tab. 6 list the detailed training configurations of different datasets for each baseline. *FOV*, $Range_{depth}$, and *#Steps* denote the field of view, the depth range, and the number of sampling steps along a camera ray, respectively. $Range_h$ and $Range_v$ denote the horizontal and vertical angle ranges of the camera pose ξ. *Sample_Dist* denotes the sampling scheme of the camera pose. We only use Gaussian or uniform sampling in our experiments. λ is the loss weight of the gradient penalty.

VolumeGAN [59]. We use the official implementation of VolumeGAN.[†] We train VolumeGAN with $25K$ images. The coordinates range of feature volume is adjustable for different datasets. We adopt the training configuration in Tab. 5 to train VolumeGAN models.

EpiGRAF [45] We use the official implementation of EpiGRAF.[‡] We inherit the patch-wise training scheme to train EpiGRAF with the same data and camera parameters at the target resolution shown in Tab. 5.

EG3D [4]. We use the official implementation of EG3D. [§] Different from VolumeGAN and EpiGRAF, EG3D renders the whole radiance field within a bounding box, so we inherit the larger camera radius than the ones of EpiGRAF and VolumeGAN for training. Since the original EG3D requires pose annotations for training, we add a pose sampler in it to enable the training on all three datasets as the global annotations are not always available. We adjust the loss weight of gradient penalty on different datasets to achieve the best performance. Hyperparameters used for training are available in Tab. 6

[†] https://github.com/genforce/volumegan
[‡] https://github.com/universome/epigraf
[§] https://github.com/NVlabs/eg3d

Table 5. **Training configurations** regarding different datasets for VolumeGAN and EpiGRAF.

Datasets	FOV	Radius	Range$_{depth}$	#Steps	Range$_h$	Range$_v$	Sample_Dist	λ
CLEVR	12.0	1.0	$[0.8, 1.2]$	24	$[\pi/2 - 0.5, \pi/2 + 0.5]$	$[\pi/4 - 0.15, \pi/4 + 0.15]$	Uniform	1
3D-FRONT	12.8	1.0	$[0.7, 1.3]$	24	$[0, 2\pi]$	$[3\pi/8 - 0.2, 3\pi/8 + 0.2]$	Uniform	1
WAYMO	12.0	1.0	$[0.7, 1.3]$	24	$[\pi/2 - 0.5, \pi/2 + 0.5]$	$[\pi/2 - 0.15, \pi/2 + 0.15]$	Uniform	1

Table 6. **Training configurations** regarding different datasets for EG3D.

Datasets	FOV	Radius	Range$_{depth}$	#Steps	Range$_h$	Range$_v$	Sample_Dist	λ
CLEVR	18.0	1.7	$[0.1, 2.6]$	96	$[\pi/2 - 0.5, \pi/2 + 0.5]$	$[\pi/4 - 0.15, \pi/4 + 0.15]$	Uniform	2
3D-FRONT	18.8	2.7	$[2.2, 3.3]$	96	$[0, 2\pi]$	$[3\pi/8 - 0.2, 3\pi/8 + 0.2]$	Uniform	2
WAYMO	18.0	1.7	$[0.1, 2.6]$	96	$[\pi/2 - 0.5, \pi/2 + 0.5]$	$[\pi/2 - 0.15, \pi/2 + 0.15]$	Uniform	5

GSN [8]. We use the official GSN implementation.[¶]. GSN highly dependents on input camera sequences and we find it very difficult to converge at a narrow camera distribution, *i.e.* WAYMO and CLEVR. On 3D-FRONT, we set the length of camera sequence to 1, and it can converge to some extent. We don't leverage depth supervision for a fair comparison with our method.

GIRAFFE [31]. We use the official implementation of GIRAFFE.[‖] The number of boxes for training follows the configuration of ours on each dataset. The bounding box generator of GIRAFFE is tuned specifically for each dataset for a fair comparison.

A3. Data Preparation

Clevr [21]. We use the official script [21] to render scenes with Cube, Cylinder, and Sphere primitives. The camera position is jittered in a small scale. And the dataset is rendered in a 256×256 resolution with $80K$ samples.

3D-Front [10, 11]. We use BlenderProc to render 20 images per room in 3D-FRONT. We move the center of each room to the coordinate origin and then sample the camera positions on the upper sphere between $2r$ to $3r$ where r is the diagonal length of room.

Waymo [47]. We only keep the front view of WAYMO for the model training. However, there exist lots of occluded and noisy cars in WAYMO, we design several heuristic rules to filter it. Specifically, we require the camera depth of car is less than $40m$ and the area of cars is larger than 40000 pixels in original image size (1920×1280). We then adopt the black padding to make images square and then resize it in to 256×256 resolution.

A5. Efficiency of Rendering Pipeline

Naïve point sampling solutions where the density and color of spatial points are inferred with multiple object radi-

Table 7. **Ablation analysis** of efficient rendering (ER).

	TR.↓		INF.↓	
	w/ ER	*w/o* ER	*w/* ER	*w/o* ER
CLEVR	**18.1**	29.2	**95**	180
3D-FRONT	**22.3**	38.1	**110**	330
WAYMO	**19.2**	30.9	**100**	195

ance fields can lead to prohibitive computational overhead. Therefore we propose an efficient rendering pipeline by only focusing on the valid points within the bounding boxes. Tab. 7 presents the training cost in *V100 days* and testing cost in *ms/image* (on a single V100 over $1K$ samples) with our efficient rendering and Naïve rendering. Our rendering pipeline can handle multiple objects efficiently, with nearly 1.5 and 2 times faster training and inference speed, respectively.

A6. Additional Results

We include a demo video, which shows more results of various 3D manipulation applications. From the video, we can see that our method both achieves good generation quality and enables precise object control. We also include comparisons with the state-of-the-art methods, *i.e.*, EG3D [4] and GIRRAFE [31], in the video.

[¶]https://github.com/apple/ml-gsn
[‖]https://github.com/autonomousvision/giraffe