Decompositional Neural Scene Reconstruction with Generative Diffusion Prior

Supplementary Material

We provide details on the decompositional reconstruction process, training procedures, experimental setup, and a discussion of limitations. Additionally, we highly recommend watching the demo video on our webpage for a more intuitive and visually engaging presentation of the results.

A. Generalizability to in-the-wild videos

Our method demonstrates robust generalizability to in-the-wild indoor scenes. Fig. S.1 presents reconstruction results of in-the-wild YouTube videos using 15 input views, with camera poses calibrated via COLMAP [24] and object masks obtained from SAM2 [21].

B. Decompositional Reconstruction

We first present preliminary of decompositional scene reconstruction, also known as object-compositional reconstruction, which aims to reconstruct each object in the scene individually—including both foreground objects and the background—rather than representing the entire scene as a single, inseparable mesh.

B.1. Decompositional Representation

Following previous work [13, 17, 28, 29], we utilize a set of posed RGB images and their corresponding instance masks to achieve decompositional reconstruction of objects, treating the background also as an object.

As described in the main paper, for a scene with k objects, we predict k signed distance function (SDF)s for each point $\boldsymbol{p}$ and the j-th $(1 \leq j \leq k)$ SDF $s_j(\boldsymbol{p})$ is for the j-th object. The scene SDF $s(\boldsymbol{p})$ is the minimum of the object SDFs:

$$s(\boldsymbol{p}) = \min_{1 \leq j \leq k} s_j(\boldsymbol{p}), \tag{S.1}$$

The normal $\boldsymbol{n}(\boldsymbol{p})$ is the gradient of $s(\boldsymbol{p})$, while normal $\boldsymbol{n}^j(\boldsymbol{p})$ is the gradient of $s_j(\boldsymbol{p})$:

$$\boldsymbol{n}(\boldsymbol{p}) = \nabla s(\boldsymbol{p}), \quad \boldsymbol{n}^j(\boldsymbol{p}) = \nabla s_j(\boldsymbol{p}) \tag{S.2}$$

Next, we transform each point's SDFs into instance semantic logits $\boldsymbol{h}(\boldsymbol{p}) = [h_1(\boldsymbol{p}), h_2(\boldsymbol{p}), \ldots, h_k(\boldsymbol{p})]$, where

$$h_j(\boldsymbol{p}) = \gamma / (1 + \exp(\gamma \cdot s_j(\boldsymbol{p}))), \tag{S.3}$$

where $\gamma = 10$ is a fixed parameter in our implementation.

B.2. Volume Rendering

For each camera ray $\boldsymbol{r} = (\boldsymbol{o}, \boldsymbol{d})$ with $\boldsymbol{o}$ as the ray origin (camera center) and $\boldsymbol{d}$ as the viewing direction, n points $\{\boldsymbol{p}_i = \boldsymbol{o} + t_i \boldsymbol{d} \mid i = 0, 1, \ldots, n-1\}$ are sampled, where t_i is the distance from the sample point to the camera center. We predict k SDFs and the color $\boldsymbol{c}_i$ for each point $\boldsymbol{p}_i$ along the ray. Then we compute scene SDF s_i, normal $\boldsymbol{n}_i$ and instance sematic logits $\boldsymbol{h}_i$ for point $\boldsymbol{p}_i$ by Eqs. (S.1) to (S.3). Next, we convert the scene SDF $s(\boldsymbol{p})$ into density σ for volume rendering as in NeRF [16] following the method introduced in VolSDF [33]:

$$\sigma(s) = \begin{cases} \frac{1}{2\beta} \exp(\frac{s}{\beta}) & s \leq 0 \\ \frac{1}{\beta}(1 - \exp(-\frac{s}{\beta})) & s \geq 0 \end{cases}, \tag{S.4}$$

where β is a learnable parameter. We then calculate the discrete accumulated transmittance T_i and discrete opacity α_i as follows:

$$T_i = \prod_{j=0}^{i-1} (1 - \alpha_j), \quad \alpha_i = 1 - exp(-\sigma_i \delta_i), \tag{S.5}$$

where δ_i represents the distance between neighboring sample points along the ray.

Using volume rendering, the predicted scene color $\hat{\boldsymbol{C}}(\boldsymbol{r})$, depth $\hat{D}(\boldsymbol{r})$, normal $\hat{\boldsymbol{N}}(\boldsymbol{r})$ and instance semantic $\hat{\boldsymbol{S}}(\boldsymbol{r})$ for the ray r are computed as:

$$\begin{aligned} \hat{\boldsymbol{C}}(\boldsymbol{r}) &= \sum_{i=0}^{n-1} T_i \alpha_i \boldsymbol{c}_i, \quad \hat{D}(\boldsymbol{r}) = \sum_{i=0}^{n-1} T_i \alpha_i t_i, \\ \hat{\boldsymbol{N}}(\boldsymbol{r}) &= \sum_{i=0}^{n-1} T_i \alpha_i \boldsymbol{n}_i, \quad \hat{\boldsymbol{S}}(\boldsymbol{r}) = \sum_{i=0}^{n-1} T_i \alpha_i \boldsymbol{h}_i, \end{aligned} \tag{S.6}$$

Additionally, replacing the scene SDF s with j-th object SDF s_j in Eqs. (S.4) and (S.5) allows rendering of the normal $\hat{\boldsymbol{N}}^j(\boldsymbol{r})$ and mask $\hat{O}^j(\boldsymbol{r})$ for j-th object as:

$$\hat{\boldsymbol{N}}^j(\boldsymbol{r}) = \sum_{i=0}^{n-1} T_i^j \alpha_i^j \boldsymbol{n}_i^j, \quad \hat{O}^j(\boldsymbol{r}) = \sum_{i=0}^{n-1} T_i^j \alpha_i^j, \tag{S.7}$$

B.3. Loss function

RGB Reconstruction Loss Given input images, we employ RGB reconstruction loss $\mathcal{L}_C$ to minimize the difference between ground-truth pixel color and the rendered color. We follow the Yu *et al*. [36] here for the RGB reconstruction loss:

$$\mathcal{L}_C = \sum_{\boldsymbol{r} \in \mathcal{R}} ||\hat{\boldsymbol{C}}(\boldsymbol{r}) - \boldsymbol{C}(\boldsymbol{r})||_1, \tag{S.8}$$

where $\hat{\boldsymbol{C}}(\boldsymbol{r})$ is the rendered color from volume rendering and $\boldsymbol{C}(\boldsymbol{r})$ denotes the ground truth.

Figure S.1. **Generalizability to YouTube videos with 15 input views.** The reconstruction results highlight our model's robust ability to generalize to in-the-wild indoor scenes.

Depth Consistency Loss Monocular depth and normal cues [36] can greatly benefit indoor scene reconstruction. For the depth consistency, we minimize the difference between rendered depth $\hat{D}(\boldsymbol{r})$ and the depth estimation $\bar{D}(\boldsymbol{r})$ from the Depthfm model [9]:

$$\mathcal{L}_D = \sum_{\boldsymbol{r}\in\mathcal{R}} ||(w\hat{D}(\boldsymbol{r}) + q) - \bar{D}(\boldsymbol{r})||^2, \tag{S.9}$$

where w and q are the scale and shift values to match the different scales. We solve w and q with a least-squares criterion, which has the closed-form solution. Please refer to the supplementary material of [36] for a detailed computation process.

Normal Consistency Loss We utilize the normal cues $\bar{\boldsymbol{N}}(\boldsymbol{r})$ from Omnidata model [6] to supervise the rendered normal through the normal consistency loss, which comprises L1 and angular losses:

$$\mathcal{L}_N = \sum_{\boldsymbol{r}\in\mathcal{R}} ||\hat{\boldsymbol{N}}(\boldsymbol{r}) - \bar{\boldsymbol{N}}(\boldsymbol{r})||_1 + ||1 - \hat{\boldsymbol{N}}(\boldsymbol{r})^T \bar{\boldsymbol{N}}(\boldsymbol{r})||_1. \tag{S.10}$$

The rendered normal $\hat{\boldsymbol{N}}(\boldsymbol{r})$ and normal cues $\bar{\boldsymbol{N}}(\boldsymbol{r})$ will be transformed into the same coordinate system by the camera pose.

Instance Semantic Loss We minimize the semantic loss between rendered instance semantic logits of each pixel and the ground-truth pixel instance class. The instance semantic objective is implemented as a cross-entropy loss:

$$\mathcal{L}_S = \sum_{\boldsymbol{r}\in\mathcal{R}} \sum_{j=1}^{k} -\bar{h}_j(\boldsymbol{r}) \log h_j(\boldsymbol{r}). \tag{S.11}$$

The $\bar{h}_j(\boldsymbol{r})$ is the ground-truth semantic probability for j-th object, which is 1 or 0.

Eikonal Loss and Smoothness Loss Following common practice [18, 33], we add an Eikonal regularization and smoothness regularization term on the sampled points to regularize the SDF learning by:

$$\mathcal{L}_{eik} = \sum_{i=0}^{n-1} (||\nabla s(\boldsymbol{p}_i)||_2 - 1), \tag{S.12}$$

$$\mathcal{L}_{smo} = \sum_{i=0}^{n-1} (||\nabla s(\boldsymbol{p}_i) - \nabla s(\tilde{\boldsymbol{p}}_i)||_1), \tag{S.13}$$

where $\tilde{\boldsymbol{p}}_i$ is randomly sampled nearby the $\boldsymbol{p}_i$.

Background Smoothness Loss Following the previous work RICO [13], we use background smoothness loss to regularize the geometry of the occluded background to be smooth. Specifically, we randomly sample a $\mathcal{P} \times \mathcal{P}$ size patch every $\mathcal{T}_\mathcal{P}$ iterations within the given image and compute semantic map $\hat{\boldsymbol{H}}(\boldsymbol{r})$ and a patch mask $\hat{M}(\boldsymbol{r})$:

$$\hat{M}(\boldsymbol{r}) = \mathbb{1}[\arg\max(\hat{\boldsymbol{H}}(\boldsymbol{r})) \neq 1], \tag{S.14}$$

wherein the mask value is 1 if the rendered class is not the background, thereby ensuring only the occluded background is regulated. Subsequently, we calculate the background depth map $\bar{D}(\boldsymbol{r})$ and background normal map $\bar{\boldsymbol{N}}(\boldsymbol{r})$ using the background SDF exclusively. The patch-based

background smoothness loss is then computed as:

$$\mathcal{L}(\hat{D}) = \sum_{d=0}^{3} \sum_{m,n=0}^{\mathcal{P}-1-2^d} \hat{M}(\boldsymbol{r}_{m,n}) \odot (|\hat{D}(\boldsymbol{r}_{m,n}) - \hat{D}(\boldsymbol{r}_{m,n+2^d})| + |\hat{D}(\boldsymbol{r}_{m,n}) - \hat{D}(\boldsymbol{r}_{m+2^d,n})|), \tag{S.15}$$

$$\mathcal{L}(\hat{\boldsymbol{N}}) = \sum_{d=0}^{3} \sum_{m,n=0}^{\mathcal{P}-1-2^d} \hat{M}(\boldsymbol{r}_{m,n}) \odot (|\hat{\boldsymbol{N}}(\boldsymbol{r}_{m,n}) - \hat{\boldsymbol{N}}(\boldsymbol{r}_{m,n+2^d})| + |\hat{\boldsymbol{N}}(\boldsymbol{r}_{m,n}) - \hat{\boldsymbol{N}}(\boldsymbol{r}_{m+2^d,n})|), \tag{S.16}$$

$$\mathcal{L}_{bs} = \mathcal{L}(\hat{D}) + \mathcal{L}(\hat{\boldsymbol{N}}) \tag{S.17}$$

Object Distinction Regularization Loss Following ObjectSDF++ [29], we employ a regularization term on object SDFs to penalize the overlap between any two objects:

$$\mathcal{L}_{sdf} = \sum_{i=0}^{n-1} (\sum_{j=1}^{k} ReLU(-s_j(\boldsymbol{p}_i) - s(\boldsymbol{p}_i))). \tag{S.18}$$

C. More Training Details

C.1. Object-compositional Reconstruction

Overall Loss We employ all the aforementioned losses in Appendix B.3 during the object-compositional reconstruction stage:

$$\mathcal{L}_{recon} = \mathcal{L}_C + \lambda_D \mathcal{L}_D + \lambda_N \mathcal{L}_N + \lambda_S \mathcal{L}_S + \lambda_{bs} \mathcal{L}_{bs} + \lambda_{eik} \mathcal{L}_{eik} + \lambda_{smo} \mathcal{L}_{smo} + \lambda_{sdf} \mathcal{L}_{sdf}, \tag{S.19}$$

where the loss weights are set as $\lambda_D = 0.1$, $\lambda_N = 0.05$, $\lambda_S = 1.0$, $\lambda_{bs} = 0.1$, $\lambda_{eik} = 0.1$, $\lambda_{smo} = 0.005$, $\lambda_{sdf} = 0.5$ following previous work [13, 29, 36].

Optimization of Visibility Grid At the end of the object-compositional reconstruction stage, when the transmittance achieves sufficient accuracy, we optimize the visibility grid G. During this process, all input views are rendered M times using Eq. (S.6), and the accumulated transmittance T_i is utilized to optimize the visibility value of point $\boldsymbol{p}_i$. Note that M is a hyperparameter that impacts the final visibility grid values; in our implementation, we set $M = 20$. After the optimization, the visibility grid is frozen for the subsequent geometry and appearance optimization stages.

C.2. Geometry Optimization

Input for the Diffusion Model In the common case, the encoder of Stable Diffusion [22] is used to encode an image into the latent code z during Score Distillation Sampling (SDS), followed by employing the UNet of Stable Diffusion to predict the score $\hat{\epsilon}$ and compute the SDS loss. However, the encoding process is relatively slow in computational speed. To address this issue and facilitate efficient training, following the previous work [2, 20], at each training iteration, we directly downsample the concatenated map $\tilde{n}_j$, which consists of the normal map $\hat{\boldsymbol{N}}^j(\boldsymbol{r})$ and mask map $\hat{O}^j(\boldsymbol{r})$ rendered by Eq. (S.7) for sampled j-th object, into the latent code z dimension. This approach reduces the computation time by approximately half without causing any performance degradation compared to directly inputting the normal map $\hat{\boldsymbol{N}}^j(\boldsymbol{r})$ into the encoder.

Overall Loss We employ reconstruction loss $\mathcal{L}_{recon}$ in Eq. (S.19) and visibility-guided geometry SDS loss $\mathcal{L}_{\text{SDS}}^{g-v}$ in the geometry optimization stage:

$$\mathcal{L}_{geo} = \mathcal{L}_{recon} + \lambda_{sds}^{geo} \mathcal{L}_{\text{SDS}}^{g-v}, \tag{S.20}$$

where $\lambda_{sds}^{geo} = 1e^{-5}$ in our implementation.

C.3. Appearance Optimization

Rendering Loss in Appearance Optimization Stage We employ two types of color rendering losses during the appearance optimization stage to ensure consistency between the observed regions and the input views. The first type derives directly from the input views, where we apply $\mathcal{L}_C$ as defined in Eq. (S.8). The second type leverages useful appearance information distilled from the reconstruction network. For this, we randomly sample camera views within the scene, render RGB and visibility maps, and use regions with high visibility for appearance supervision. The sum of these two color rendering losses is denoted as $\mathcal{L}_C^a$.

Depth-guided Panorama Inpainting and Loss for BG Applying appearance SDS $\mathcal{L}_{\text{SDS}}^{a-v}$ for background appearance optimization often leads to degenerated results, *e.g.* introducing non-existent objects as shown in Fig. S.3, due to the lack of clear geometric cues in the background from the local camera perspective. To address this issue, inspired by previous work [25, 27], we adopt depth-guided inpainting [37] to refine the low-visibility regions of the background panorama color map. Specifically, we first generate the original RGB, visibility, and depth panorama maps, as shown in Fig. S.2 (a, b, e). Next, we obtain the inpainting mask (Fig. S.2 (c)) for regions where the visibility map falls below a threshold τ (set to $\tau = 0.2$ in our implementation). Finally, we apply depth-guided inpainting to produce the inpainted RGB panorama map (Fig. S.2 (f)).

To supervise the background appearance during the appearance optimization stage, we transform the inpainted RGB panorama map into a set of perspective images with corresponding camera poses. At each training iteration, we sample B perspective images $\boldsymbol{C}_B$ along with their associated camera poses. For these poses, we render the

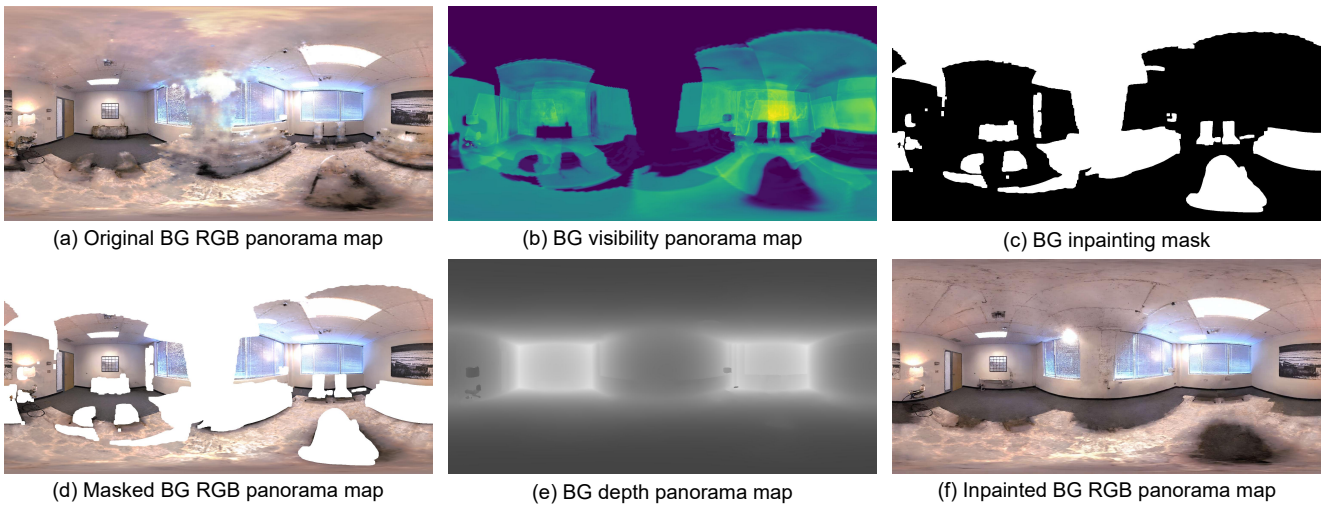

Figure S.2. **Panorama inpainting of background**. We present the original panorama map, the inpainted panorama map, the visibility panorama map, as well as the mask and depth guidance used in the depth-guided panorama inpainting process.

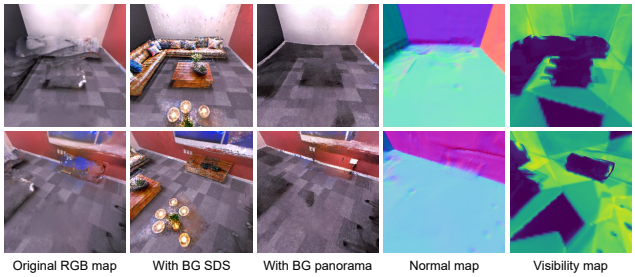

Figure S.3. **Failure case of directly optimizing background with SDS loss.** Incorporating appearance SDS for the background may result in hallucinated non-existent objects in low-visibility regions, even when the smooth normal map indicates no objects in the background. On the contrary, the background map obtained after optimizing depth-guided panorama inpainting produces a cleaner and more reasonable background texture.

background color maps $\hat{\boldsymbol{C}}_B$, and define the background panorama loss as:

$$\mathcal{L}_{\text{bg-pano}} = \frac{1}{B}\sum_{i=1}^{B} ||\hat{\boldsymbol{C}}_B - \boldsymbol{C}_B||_1, \tag{S.21}$$

Overall Loss We employ color rendering loss $\mathcal{L}_C^a$, background panorama loss $\mathcal{L}_{\text{bg-pano}}$ in Eq. (S.21) and visibility-guided appearance SDS loss $\mathcal{L}_{\text{SDS}}^{a-v}$ in the appearance optimization stage:

$$\mathcal{L}_{app} = \lambda_C^a \mathcal{L}_C^a + \lambda_{\text{bg-pano}} \mathcal{L}_{\text{bg-pano}} + \mathcal{L}_{\text{SDS}}^{a-v}, \tag{S.22}$$

where $\lambda_C^a = \lambda_{\text{bg-pano}} = 1e^4$ in our implementation.

Export UV Map Following previous works [2, 20], we utilize the trained ψ to export the appearance of object mesh as UV map by the xatlas [35]. Our object mesh with exported UV map supports direct use and editing in common 3D software, *e.g.* Blender [4], as shown in Fig. S.4. We export 1024×1024 UV map for foreground objects and 2048×2048 UV map for the background in our case.

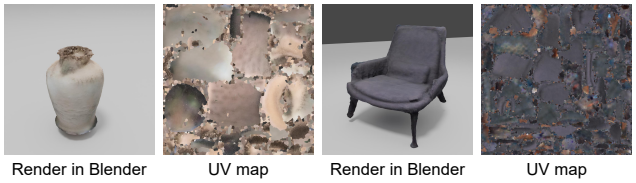

Figure S.4. **Visualization of UV mapping and rendering results.** Our method produces completed meshes with detailed UV maps, enabling photorealistic rendering in common 3D software such as Blender.

D. More Experiment Details

D.1. Baselines Details

For MonoSDF [36], RICO [13], and ObjectSDF++ [29], which are designed for reconstruction, we directly utilize the official code to obtain the reconstructed meshes and rendered images. For FreeNeRF [32], which focuses on novel view synthesis and does not include reconstruction code, we first predict depth maps and RGB images for densely sampled camera views within the scene. We then apply TSDF Fusion [5] to integrate the predicted depth maps into a TSDF volume and export the resulting mesh. ZeroNVS [23] trains a diffusion model to synthesize novel views of scenes from a single image. To adapt it for multi-view inputs, we follow the approach of ReconFusion [30], using the input

view closest to the novel view as the conditioning view for the diffusion model. We denote the adapted version as ZeroNVS*. Subsequently, we use MonoSDF to reconstruct the scene mesh from the images synthesized by ZeroNVS*.

D.2. Data Preparation

Monocular Cues We utilize the pre-trained DepthFM model [9] and Omnidata model [6] to generate the depth map $\bar{D}$ and normal map $\bar{\boldsymbol{N}}$ for each RGB image, respectively. While depth cues provide semi-local relative information, normal cues are inherently local, capturing fine geometric details. As a result, we expect surface normals and depth to complement each other effectively. It is worth noting that estimating absolute scale in general scenes remains challenging; therefore, $\bar{D}$ should be interpreted as a relative depth cue.

GT Instance Mask For the ScanNet++ [34] dataset, we use the official rendering engine to generate instance masks for each image based on the provided GT mesh and per-vertex 3D instance annotations. For the Relica [26] datasets, we utilize the original instance masks from vMAP [12], which are overly fragmented, and manually merge adjacent fragmented instance masks into coherent objects. Furthermore, since both ScanNet++ and Replica only provide a complete mesh of the scene, we derive the background GT mesh by removing the object meshes from the total mesh and manually filling the holes.

Notably, with the rapid advancement of segmentation and tracking models, such as SAM [11] and SAM2 [21], it's more feasible to extract object masks directly from images using off-the-shelf models. These tools could inspire further progress in decompositional neural scene reconstruction.

D.3. Reconstruction Metrics Details

Following previous research [13, 29, 36], we evaluate the Chamfer Distance (CD) in cm, F-score with a threshold of $5cm$ and Normal Consistency (NC) for 3D scene and object reconstruction. Consistent with previous studies [13, 29, 36], reconstruction is evaluated only on visible areas for the entire scene, while complete meshes are assessed for individual objects and background meshes. These metrics are defined in Tab. S.1.

Since the baselines ZeroNVS* [23], FreeNeRF [32] and MonoSDF [36] can only reconstruct the total scene and cannot decompose it into individual objects, we evaluate the metrics only for the total scene, *i.e.*, the total scene reconstruction metrics and rendering metrics.

D.4. Implementation Details

We implement our model in PyTorch [19] and utilize the Adam optimizer [10] with an initial learning rate of $5e-4$. In the object-compositional reconstruction stage, we sample 1024 rays per iteration, and in the geometry and appearance optimization stages, we render 128×128 images for normal, mask, and color maps. We use 2048×1024 for the background panorama map.

For visibility guided SDS:

$$w^v(z) = \begin{cases} w_0 + m_0 V(z) & \text{if } V(z) \leq \tau \\ w_1 + m_1 V(z) & \text{if } V(z) > \tau \end{cases}, \tag{S.23}$$

we set $\tau = 0.5, w_0 = 20, m_0 = -38, w_1 = 2, m_1 = -2$ for the geometry optimization stage. Under this configuration, $w^v(z)$ achieves a maximum value of 20 when $V(z) = 0$, a minimum value of 0 when $V(z) = 1$, and a value of 1 when $V(z) = \tau = 0.5$. For appearance optimization stage, we set $\tau = 0.3, w_0 = 1, m_0 = 0, w_1 = 0, m_1 = 0$, which results in $w^v(z) = 1$ when $V(z) \leq 0.3$ and $w^v(z) = 0$ when $V(z) > 0.3$.

Our model is trained for 80000 iterations on both Replica [26] and ScanNet++ [34] datasets. The geometry optimization stage and appearance optimization stage begin at the 35000^{th} and 75000^{th} iterations, respectively. All experiments are conducted on a single NVIDIA-A100 GPU, requiring approximately 10 hours to complete the training of a single scene.

D.5. Training Time Comparison

For a fair comparison with prior work [13, 29, 36], we train our model and all baselines for 80,000 iterations in all main paper experiments. Detailed training time and performance results are provided in Tab. S.2, showing that our method outperforms baselines in approximately 4.5 hours per scene with 50,000 iterations.

E. Scene Editing Details

E.1. Text-based Editing

With our decompositional representation, which breaks down each object's representation in the scene, we can seamlessly edit the representation of any object in the scene based on the text prompt, with the generative capabilities of our SDS diffusion prior. With the two forms of SDS prior we introduced, *i.e.*, the geometry SDS prior and the appearance SDS prior, we can freely edit both the geometry and appearance of objects. During the editing process, we exclude the reconstruction loss for the edited object and disable the visibility guidance.

Geometry Editing We realize geometry editing for objects in the geometry optimization stage using geometry SDS loss $\mathcal{L}^g_{\text{SDS}}$. During optimization, we replace the original object prompt with the desired object prompt while continuing to sample novel camera views around the original object's bounding box. This ensures that the desired object retains the same location as the original one.

Table S.1. **Evaluation metrics.** We show the evaluation metrics with their definitions that we use to measure reconstruction quality. P and P^* are the point clouds sampled from the predicted and the ground truth mesh. $\boldsymbol{n_p}$ is the normal vector at point $\boldsymbol{p}$.

Metric	Definition
Chamfer Distance (CD)	$\frac{\textit{Accuracy}+\text{Completeness}}{2}$
Accuracy	$\underset{\boldsymbol{p}\in P}{\text{mean}}\left(\underset{\boldsymbol{p}^*\in P^*}{\min}\|\|\boldsymbol{p}-\boldsymbol{p}^*\|\|_1\right)$
Completeness	$\underset{\boldsymbol{p}^*\in P^*}{\text{mean}}\left(\underset{\boldsymbol{p}\in P}{\min}\|\|\boldsymbol{p}-\boldsymbol{p}^*\|\|_1\right)$
F-score	$\frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}}$
Precision	$\underset{\boldsymbol{p}\in P}{\text{mean}}\left(\underset{\boldsymbol{p}^*\in P^*}{\min}\|\|\boldsymbol{p}-\boldsymbol{p}^*\|\|_1<0.05\right)$
Recall	$\underset{\boldsymbol{p}^*\in P^*}{\text{mean}}\left(\underset{\boldsymbol{p}\in P}{\min}\|\|\boldsymbol{p}-\boldsymbol{p}^*\|\|_1<0.05\right)$
Normal Consistency	$\frac{\textit{Normal Accuracy}+\textit{Normal Completeness}}{2}$
Normal Accuracy	$\underset{\boldsymbol{p}\in P}{\text{mean}}\left(\boldsymbol{n}_{\boldsymbol{p}}^T\boldsymbol{n}_{\boldsymbol{p}^*}\right)$ s.t. $\boldsymbol{p}^*=\underset{\boldsymbol{p}^*\in P^*}{arg\,min}\|\|\boldsymbol{p}-\boldsymbol{p}^*\|\|_1$
Normal Completeness	$\underset{\boldsymbol{p}^*\in P^*}{\text{mean}}\left(\boldsymbol{n}_{\boldsymbol{p}}^T\boldsymbol{n}_{\boldsymbol{p}^*}\right)$ s.t. $\boldsymbol{p}=\underset{\boldsymbol{p}\in P}{arg\,min}\|\|\boldsymbol{p}-\boldsymbol{p}^*\|\|_1$

Table S.2. **Training time and performance comparison.** Our method outperforms baselines in 4.5 hours per scene with 50,000 iterations.

Total Iter	RICO			ObjectSDF++			Ours		
	Time	CD↓	F-Score↑	Time	CD↓	F-Score↑	Time	CD↓	F-Score↑
40000	2.63h	21.30	49.47	2.34h	18.48	52.51	2.59h	12.96	61.87
50000	3.16h	17.37	52.33	2.93h	13.36	60.37	4.52h	4.51	72.66
60000	3.82h	14.63	57.74	3.42h	5.42	70.19	6.54h	4.35	73.23
80000	5.01h	12.09	63.39	4.48h	5.10	70.87	10.45h	4.33	73.32

Appearance Editing We perform object appearance editing during the appearance optimization stage using the appearance SDS loss $\mathcal{L}^a_{\text{SDS}}$. For this task, we not only modify the object prompt but also update the negative prompt in Stable Diffusion [22], as suggested in the prompt engineering tutorial [1]. Empirically, we observe that appearance optimization is more sensitive to the choice of the negative prompt compared to geometry optimization. For scene stylization, we use a consistent style prompt for editing the appearance of not only all objects but also generating the background panorama, which is achieved through depth-guided ControlNet [37].

E.2. VFX Editing

The object meshes reconstructed by our method feature detailed UV maps, making them compatible with common 3D software and enabling diverse and photorealistic VFX editing. We implement our VFX editing in Blender, as demonstrated in our main paper. More specifically,

- *"**Freeze it**"* utilizes the Geometry Nodes Modifier and applies a glass material over the original object.
- *"**Ignite it**"* employs the Quick Smoke Effect, setting the *Flow Type* to *Fire* and *Smoke*, with fire color adjustments via Shading Nodes.
- *"**Break it by a ball**"* uses the Cell Fracture Effect to divide the object into multiple fragments, assigning both the object and ball as Rigid Bodies for physics-based simulation in Blender.

F. Comparison with Image-to-3D Method

Unlike decompositional scene reconstruction methods, which recover object geometry along with location, rotation, and scale simultaneously from multi-view images, an alternative approach is to use image-to-3D models for ex-

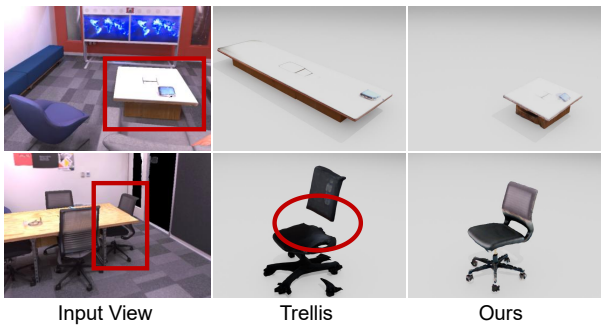

Figure S.5. **Comparison with image-to-3D method Trellis [31].** For better visualization, we adjust the location and rotation of Trellis results manually.

tracting individual objects within a scene. However, as shown in Fig. S.5, these models (*e.g.*, Trellis [31]) face significant challenges in recovering the location, rotation, and scale of objects, with severe performance deterioration under occlusion, making them less applicable for scene reconstruction regardless of views compared to our method.

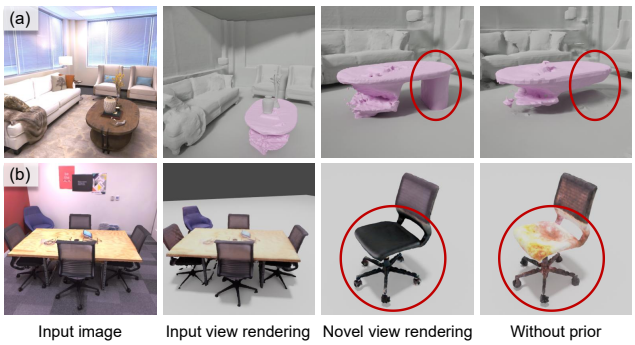

Figure S.6. **Qualitative Examples for Failure Cases.** We present failure cases for both geometry optimization and appearance optimization. The first column displays the input view, while the second and third columns show our results rendered from the input view and a novel view, respectively. The final column provides the corresponding results without applying the geometry prior (top) or appearance prior (bottom), highlighting the improvements introduced by our generative prior.

G. Failure Cases and Limitation

In this section, we present and analyze examples of representative failure. Fig. S.6 (a) demonstrates that our method may produce non-harmonious structures with inaccurate text prompts. For instance, in this example, we use the prompt *"A tea table"*, but the table in this case does not conform to the conventional concept of a tea table. A similar issue arises during appearance optimization, as shown in Fig. S.6 (b), where we use the prompt *"A black ergonomic chair"*, but the chair in this case is not entirely black—it appears somewhat gray—resulting in a non-harmonious appearance in the completed regions. We believe that leveraging implicit prompts [7] could help alleviate such issues related to text prompts.

Moreover, our method optimizes each object independently in each iteration, using the 3D location information from the reconstruction module alone. This could be further improved by forming functional object groups by composing neighboring objects. Within these groups, SDS can be employed to optimize inter-object relationships and plausible layouts [3, 38]. Additionally, SDS-based methods [2, 20] struggle to reconstruct loose geometries such as grass, hair, sky, and fur, which are challenging to describe with text prompts. In contrast, concurrent methods [8, 14, 15, 30] that directly generate novel view images from sparse input views without relying on text prompts may mitigate this limitation. However, these methods often fail to maintain the 3D consistency of objects across views, achieve object-level editing, or reconstruct regions obscured by occlusions.

References

[1] Andrew. Sdxl-styles. https://stable-diffusion-art.com/sdxl-styles/, 2023. 6

[2] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In *International Conference on Computer Vision (ICCV)*, 2023. 3, 4, 7

[3] Yongwei Chen, Tengfei Wang, Tong Wu, Xingang Pan, Kui Jia, and Ziwei Liu. Comboverse: Compositional 3d assets creation using spatially-aware diffusion guidance. In *European Conference on Computer Vision (ECCV)*, 2024. 7

[4] Blender Online Community. Blender - a 3d modelling and rendering package. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 4

[5] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA)*, 1996. 4

[6] Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In *International Conference on Computer Vision (ICCV)*, 2021. 2, 5

[7] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H. Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022. 7

[8] Ruiqi Gao*, Aleksander Holynski*, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul P. Srinivasan, Jonathan T. Barron, and Ben Poole*. Cat3d: Create anything in 3d with multi-view diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 7

[9] Ming Gui, Johannes S. Fischer, Ulrich Prestel, Pingchuan Ma, Dmytro Kotovenko, Olga Grebenkova, Stefan Andreas Baumann, Vincent Tao Hu, and Björn Ommer. Depthfm: Fast monocular depth estimation with flow matching. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2025. 2, 5

[10] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[11] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. In *International Conference on Computer Vision (ICCV)*, 2023. 5

[12] Xin Kong, Shikun Liu, Marwan Taher, and Andrew J Davison. vmap: Vectorised object mapping for neural field slam. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 5

[13] Zizhang Li, Xiaoyang Lyu, Yuanyuan Ding, Mengmeng Wang, Yiyi Liao, and Yong Liu. Rico: Regularizing the unobservable for indoor compositional reconstruction. In *International Conference on Computer Vision (ICCV)*, 2023. 1, 2, 3, 4, 5

[14] Fangfu Liu, Wenqiang Sun, Hanyang Wang, Yikai Wang, Haowen Sun, Junliang Ye, Jun Zhang, and Yueqi Duan. Reconx: Reconstruct any scene from sparse views with video diffusion model. *arXiv preprint arXiv:2408.16767*, 2024. 7

[15] Xi Liu, Chaoyi Zhou, and Siyu Huang. 3dgs-enhancer: Enhancing unbounded 3d gaussian splatting with view-consistent 2d diffusion priors. *arXiv preprint arXiv:2410.16266*, 2024. 7

[16] 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 *European Conference on Computer Vision (ECCV)*, 2020. 1

[17] Junfeng Ni, Yixin Chen, Bohan Jing, Nan Jiang, Bin Wang, Bo Dai, Puhao Li, Yixin Zhu, Song-Chun Zhu, and Siyuan Huang. Phyrecon: Physically plausible neural scene reconstruction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 1

[18] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *International Conference on Computer Vision (ICCV)*, 2021. 2

[19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 5

[20] Lingteng Qiu, Guanying Chen, Xiaodong Gu, Qi Zuo, Mutian Xu, Yushuang Wu, Weihao Yuan, Zilong Dong, Liefeng Bo, and Xiaoguang Han. Richdreamer: A generalizable normal-depth diffusion model for detail richness in text-to-3d. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3, 4, 7

[21] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. 1, 5

[22] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3, 6

[23] Kyle Sargent, Zizhang Li, Tanmay Shah, Charles Herrmann, Hong-Xing Yu, Yunzhi Zhang, Eric Ryan Chan, Dmitry Lagun, Li Fei-Fei, Deqing Sun, and Jiajun Wu. ZeroNVS: Zero-shot 360-degree view synthesis from a single real image. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 4, 5

[24] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[25] Mira Slavcheva, Dave Gausebeck, Kevin Chen, David Buchhofer, Azwad Sabik, Chen Ma, Sachal Dhillon, Olaf Brandt, and Alan Dolhasz. An empty room is all we want: Auto-

matic defurnishing of indoor panoramas. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2024. 3

[26] 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 Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 5

[27] Qi Wang, Ruijie Lu, Xudong Xu, Jingbo Wang, Michael Yu Wang, Bo Dai, Gang Zeng, and Dan Xu. Roomtex: Texturing compositional indoor scenes via iterative inpainting. In *European Conference on Computer Vision (ECCV)*, 2024. 3

[28] Qianyi Wu, Xian Liu, Yuedong Chen, Kejie Li, Chuanxia Zheng, Jianfei Cai, and Jianmin Zheng. Object-compositional neural implicit surfaces. In *European Conference on Computer Vision (ECCV)*, 2022. 1

[29] Qianyi Wu, Kaisiyuan Wang, Kejie Li, Jianmin Zheng, and Jianfei Cai. Objectsdf++: Improved object-compositional neural implicit surfaces. In *International Conference on Computer Vision (ICCV)*, 2023. 1, 3, 4, 5

[30] Rundi Wu, Ben Mildenhall, Philipp Henzler, Keunhong Park, Ruiqi Gao, Daniel Watson, Pratul P. Srinivasan, Dor Verbin, Jonathan T. Barron, Ben Poole, and Aleksander Holynski. Reconfusion: 3d reconstruction with diffusion priors. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 4, 7

[31] Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. *arXiv preprint arXiv:2412.01506*, 2024. 7

[32] Jiawei Yang, Marco Pavone, and Yue Wang. Freenerf: Improving few-shot neural rendering with free frequency regularization. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 4, 5

[33] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 1, 2

[34] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *International Conference on Computer Vision (ICCV)*, 2023. 5

[35] Jonathan Young. xatlas. https://github.com/jpcy/xatlas, 2021. 4

[36] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 1, 2, 3, 4, 5

[37] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *International Conference on Computer Vision (ICCV)*, 2023. 3, 6

[38] Xiaoyu Zhou, Xingjian Ran, Yajiao Xiong, Jinlin He, Zhiwei Lin, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. Gala3d: Towards text-to-3d complex scene generation via layout-guided generative gaussian splatting. In *International Conference on Machine Learning (ICML)*, 2024. 7