Set-the-Scene: Global-Local Training for Generating Controllable NeRF Scenes

Dana Cohen-Bar Elad Richardson Gal Metzer Raja Giryes Daniel Cohen-Or

Tel Aviv University

Figure 1: Set-the-Scene allows generating composable and controllable scenes from text prompts and 3D object proxies. (left) The scene is represented using a set of proxies, defining the location, coarse shape, and text prompt of each target object. A set of NeRFs are then optimized with respect to the object proxies and an additional scene text prompt. (right) By manipulating the proxies, the scene can be edited without additional fine-tuning.

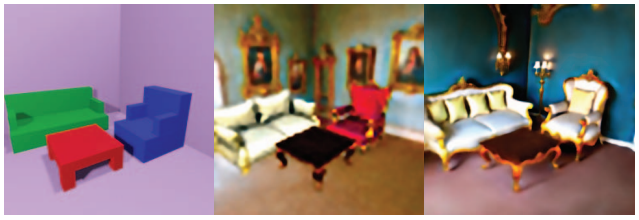
Abstract

Recent breakthroughs in text-guided image generation have led to remarkable progress in the field of 3D synthesis from text. By optimizing neural radiance fields (NeRF) directly from text, recent methods are able to produce remarkable results. Yet, these methods are limited in their control of each object's placement or appearance, as they represent the scene as a whole. This can be a major issue in scenarios that require refining or manipulating objects in the scene. To remedy this deficit, we propose a novel Global-Local training framework for synthesizing a 3D scene using object proxies. A proxy represents the object's placement in the generated scene and optionally defines its coarse geometry. The key to our approach is to represent each object as an independent NeRF. We alternate between optimizing each NeRF on its own and as part of the full scene. Thus, a complete representation of each object can be learned, while also creating a harmonious scene with style and lighting match. We show that using proxies allows a wide variety of editing options, such as adjusting the placement of each independent object, removing objects from a scene, or refining an object. Our results show that Set-the-Scene offers a powerful solution for scene synthesis and manipulation, filling a crucial gap in controllable text-to-3D synthesis.

1. Introduction

Creating high-quality 3D content has traditionally been a time-consuming process, requiring specialized skills and knowledge. However, recent advances in text-to-3D synthesis [19, 13, 14, 26] are revolutionizing the generation of 3D scenes. These methods use pretrained text-to-image diffusion models to optimize a Neural Radiance Field (NeRF) and generate 3D objects that match a given text prompt.

In spite of these exciting advancements, current solutions still lack the capability to create a specific envisioned scene. That is because controlling the generation of text prompts alone is extremely challenging, especially if one wants to describe specific objects with defined dimensions and geometry, and locate them at specific positions. Moreover, once generated, modifying specific aspects of the scene, while leaving the others untouched, can prove to be challenging. Various aspects of a scene may need to be modified after generation, such as the position or orientation of certain objects, or the texture or geometry of individual components. One may also want to export a single object to be used in another scene. However, current methods represent the scene as a whole, and objects are interdependent in their representation, making it impossible to edit a specific scene component or use objects in other scenes.



Input Local Only Global-Local

Figure 2: **Importance of Global-Local training.** (Input) The proxy objects used to define the scene. (Local Only) A scene where each object is optimized only for itself; notice how objects look pasted and do not match in terms of color scheme. (Global-Local) Our global-local training, which also interleaves global training steps of the entire scene.

Given that the generation process of scenes using current text-to-3D methods takes a considerable amount of time, the need for interactive editing capabilities has become even more apparent. One may potentially save time and gain control over individual objects by generating them separately and then just rendering them together at inference time. However, such an approach has several limitations: it cannot generate objects that interact with each other; it cannot ensure consistency in style between objects; and it cannot model the interaction among objects, like shadows and other global shading effects, see Figure 2 (A).

In this paper, we introduce a novel framework for synthesizing a controllable scene using text and object proxies, using a **Global-Local** approach. The key idea is to represent the scene as a composition of multiple object NeRFs, each built around an object proxy. The models are jointly optimized to "locally" represent the required object and "globally" to be part of the larger scene. Both the local and the global optimization propagate gradients into the same models, creating a harmonious scene composed of disentangled objects, see Figure 2 (B). For optimizing our objects and scenes, we follow [19] and use the score distillation loss. Our method leverages the composability of our representation and iteratively alternates between localized training of individual objects and optimizing the scene as a whole, where objects are dependent on their representation. When optimizing a single NeRF, we simply render it on its own from a random viewpoint and apply score distillation based on a text prompt describing the object. For scene-level optimization, we shift the rays using a rigid transformation to match the desired placement defined by each object proxy and apply score distillation with a "scene text prompt".

In many scenarios it is desirable to not only define the placement of an object, but also its dimensions and coarse geometry. Therefore, we also optionally apply a shape loss [14] on each object proxy to guide it towards a specific shape. In addition, we demonstrate that our approach enables the definition of multiple object proxies that can be linked to a single object NeRF. This permits the specification of replicated objects that are intended to be located in multiple positions throughout the scene (e.g., chairs around a table), while aggregating the score distillation from the different placements to optimize a single NeRF

Our proposed Global-Local approach not only provides more control during the training process, but also allows for better editing and fine-tuning of generated scenes. Specifically, using object proxies, we can easily control the placement of an object without the need for further refinement and even remove or duplicate the object as desired. Additionally, we can selectively fine-tune only parts of the scene by defining the set of proxies that are trained and fine-tuning the respective NeRF with modified text prompts. Furthermore, we demonstrate that the object proxy can be used to define geometry edits on the coarse shape, which are then applied during the fine-tuning process.

The contributions of our paper are threefold: (i) we propose to represent each object in the scene as a separate NeRF around a proxy, which allows getting a disentangled model for each object, (ii) we introduce a new optimization strategy that interleaves between single-object optimization and scene optimization, resulting in self-contained objects that can be combined to create a plausible scene, and (iii) our strategy provides control over the generated scene, both before and after its creation.

2. Related Work

Text-Driven Shape Generation 3D shape generation from text has been a well-researched topic for the past couple of years. Text2Mesh [15], ClipMesh [9] and Tango [2] use Clip [20] to optimize a triangular mesh to match an input text prompt. ClipForge [23] trains a point cloud generator conditioned on Clip embeddings, such that novel shapes can be generated solely by a single matching text prompt. DreamFields [8] uses Clip supervision to guide a 3D NeRF scene to match a target text.

There has been an extraordinary development in NeRF papers in recent years. Originally used for capturing scenes from a collection of images [16, 1, 17], NeRFs have recently also been adopted for shape generation [8, 19, 26, 14, 13, 12]. DreamFusion [19] first introduced *Score-Distillation* for generating novel objects conditioned on an input text prompt by leveraging a pretrained 2D diffusion model. Latent-NeRF [14] extended DreamFusion to the latent domain in order to leverage the publicly available Stable-Diffusion [21]. Additionally, Latent-NeRF introduced *Sketch-Shapes* as proxies for specifying a desired approximate target geometry. Yet, their approach is designed to generate only a single object and not multiple ones.

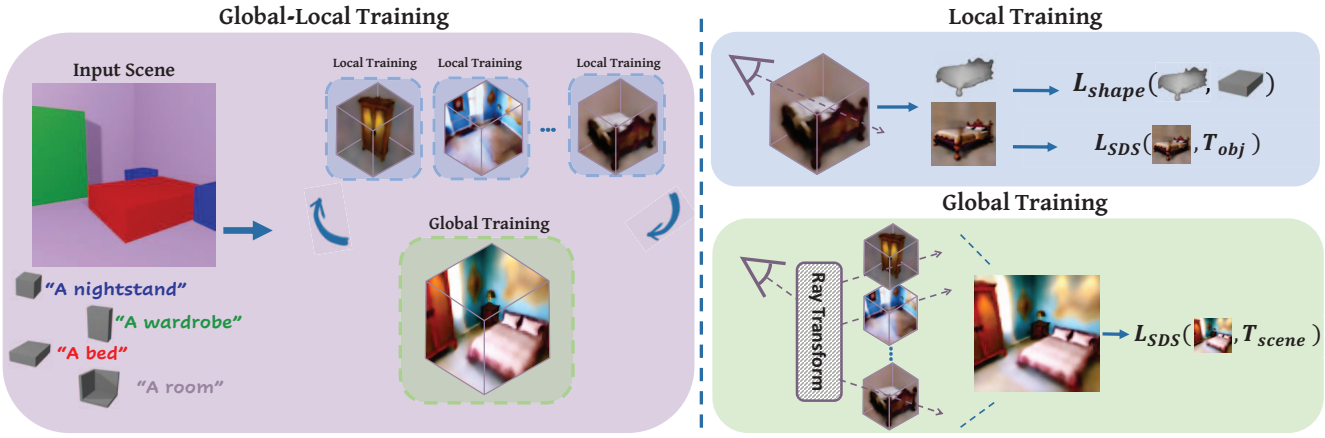


Figure 3: **Set-the-Scene Training pipeline.** A scene is first defined using a set of proxies, where each proxy is coupled with a location and a text prompt. Given an input scene, we then apply a global-local training procedure where we alternate between locally optimizing each object on its own and optimizing the entire scene rendered together as a whole.

Scene Generation Scene generation has also been tackled before using different approaches. Koh *et al*. [10] uses point cloud rendering as a 3D consistent representation and a 2D GAN architecture to synthesize novel realistic and spatially consistent viewpoints of indoor scenes. SceneDreamer [3] is able to generate large-scale 3D landscapes from a collection of images, leveraging a 3D consistent *bird-eye-view* that is trained using a 2D GAN objective. SceneScape [6] generates novel parts of a scene conditioned on an input text description of the sense and an initial image, by iteratively warping and out-painting the previously generated image according to specified camera movements. These methods generate the scene as a whole and do not offer composable control over the generated scenes.

Composable Neural Radiance Fields Decomposing neural scenes enables controlling different objects in the scene in a disentangled manner, allowing for explicit editing, removal, and addition of objects. Compressible-composable NeRF [24] utilizes a tensor representation of the scene for compression, which also enables rendering multiple objects together as a unified scene. GIRAFFE [18] uses an adversarial loss to train a decomposable generator of feature fields, which when rendered together, form a single scene. The disentangled control over each individual feature field allows controlling the location and appearance of each individual object in the scene.

Object-Centric NeRF [7] is able to render multiple NeRF implicit models altogether, where each model was independently optimized to capture a single object. Similarly, Panoptic Neural Fields [11] also trains separate implicit models for each object, plus a single model for the background. Additionally, each object is associated with a 3D bounding box, which is considered in the merging policy between implicit models at the volumetric rendering stage of the entire scene. Nerflets [29] propose a set of local NeRFs that represent a scene as a collection of decomposed objects. Each of them maintains its own spatial position, orientation, and extent, allowing for efficient structure-aware 3D scene representation from images. Object-NeRF [28] is able to learn individual per-object NeRF models of a cluttered scene, through object-level supervision that is achieved by leveraging a rough segmentation mask of each training image. We adopt the same ray aggregation strategy as Object-NeRF for rendering multiple implicit models in a single image.

Note that all the above methods are not generative nor text-guided and are limited in their editing capabilities. Similarly to our global-local paradigm, DisCoScene [27] also learns to generate controllable scenes at inference time, yet they require a 3D object-level training dataset, while our method does not use any 3D supervision.

3. Method

We turn to describe Set-the-Scene. We start by defining object proxies and composable NeRF rendering and then formulate our Global-Local training. Finally, we show how one may easily apply post-training edits using our method.

3.1. Composable NeRF Rendering

In order to allow object-level optimization and generation, we represent our scenes as a set of composable NeRFs. For explaining our proposed strategy, we first describe general NeRF rendering and the concept of *object proxies* for improved control over the scene.

NeRF Rendering A single NeRF [16] represents a 3D volumetric scene with a 5D function f_Θ that maps a 3D coordinate $\mathbf{x} = (x, y, z)$ and a 2D viewing direction $\mathbf{d} = (\theta, \phi)$ into a volume density σ and an emittance color $\mathbf{c} = (r, g, b)$. The rendering in Equation (1) is utilized to render the scene from a defined camera location o, by shooting rays from o through each pixel and integrating them for color. More specifically, given a ray $\mathbf{r}$ originating at $\mathbf{o}$ with direction $\mathbf{d}$, we query f_Θ at points $\mathbf{x}_i = \mathbf{o} + t_i\mathbf{d}$ that are sequentially sampled along the ray to get densities $\{\sigma_i\}$ and colors $\{\mathbf{c}_i\}$. Finally, the pixel color is taken to be:

$$\hat{\mathbf{C}}(\mathbf{r}) = \sum_i T_i \alpha_i \mathbf{c}_i, T_i = \prod_{j<i}(1-\alpha_i),$$
$$\alpha_i = 1 - \exp(-\sigma_i\delta_i), \delta_i = t_{i+1} - t_i \tag{1}$$

where δ_i is the step size, α_i is the opacity, and T_i is the transmittance.

Object Proxies When rendering multiple NeRFs in the same scene, one first has to define their respective locations in the scene. We do so through *object proxies*, where each proxy is associated with a NeRF model, offset position, orientation, and size, which define its settings in the scene. We designed our method so that a single NeRF can be associated with multiple proxies. This allows rendering scenes with numerous instances of the same object, which in turn can be optimized together, as discussed in Section 3.2.

Multi-Proxy NeRF Rendering When rendering multiple objects together, we divide the sampled points from each ray between the different objects, such that each object gets a fraction of the points. In order to integrate a single ray over different proxies, each set of points has to undergo a rigid transformation from the scene coordinate system to the proxy coordinate system, such that the computed colors and opacity values are calculated at the object level. This allows sharing information between different proxies associated with a single NeRF model, and between the Local-Global training phases described in Section 3.2.

We generalize the treatment in Equation 1 by summing over different NeRFs and transforming their input points according to their proxy parameters, i.e.,

$$\hat{\mathbf{C}}(\mathbf{r}) = \sum_i T_i \alpha_i^{(k)} \mathbf{c}_i^{(k)},$$
$$x_i^{(k)} = R^{(k)}(x_i + \mathrm{loc}^{(k)}), \tag{2}$$

where $\alpha_i^{(k)}$ is the output of the kth NeRF, R^k is a rigid matrix defined by the proxy's scale and orientation, and $\mathrm{loc}^{\mathrm{k}}$ is a 3D coordinate defined by the proxy's location. The index k is selected to be $k = i \bmod N_{\mathrm{obj}}$, where N_{obj} is the total number of objects in the scene, i.e., we sample alternatively between each of the objects equitably.

3.2. Global-Local Training

Given our composing mechanism, we now turn to describe our training losses and supervision technique.

Score Distillation To optimize a NeRF using a text-prompt we follow the score distillation loss proposed in [19]. Score distillation turns a pretrained diffusion model $\mathcal{M}$ to a critic that is able to provide per-pixel gradients that measure the similarity of a given image to a target text.

$$\nabla_x L_{sds} \sim \epsilon_\theta(x, \mathcal{T}) - \epsilon, \tag{3}$$

where ϵ is a random noise purposely added to the image x, ϵ_θ is the predicted noise by $\mathcal{M}$, and $\mathcal{T}$ is the target text prompt. This allows us to optimize a NeRF to gradually match a given text prompt. In practice, we apply the score-distillation directly to the latent representation and only later decode the results as proposed in [14].

Interleaved Training Our method involves an iterative process where we alternate between optimizing each object individually and optimizing the entire scene as a whole. This allows us to take advantage of our composable representation and create objects that harmonize well together but can still be rendered independently. The object-level iterations are especially important for optimizing occluded areas, which might not be visible at all on the scene level.

In practice, for object-level iterations we choose one of the scene objects and render it in its canonical coordinate system, $\mathcal{O}_i$, such that it is located at the origin and we optimize it with a user-provided text prompt that describes it (*e.g.* "a wardrobe"). For scene-level iterations, we render all the objects together based on their proxies in the scene coordinate system $\mathcal{S}$ and use a text prompt describing the scene as a whole (*e.g.* "a baroque bedroom").

Defining Proxy Geometry A proxy object is always used to define a NeRF placement in the scene. For even higher levels of control, we adopt the shape loss from [14] and allow user-defined shape proxies for each individual object in the scene. The shape constraints allow users to define proxy geometries in the form of 3D-like sketches and control the dimensions and structure of the generated object, resulting in a much more controllable process.

In practice, the geometry proxy constraint is imposed through an auxiliary loss function, applied both on the scene scale $\mathcal{S}$ and on each individual object scale $\mathcal{O}_i$ alongside the score distillation loss function L_{sds}:

$$L_{shape} = CE(\alpha_{NeRF}(p), \alpha_{GT}(p)) \cdot (1 - e^{-\frac{d^2}{2\sigma_S}}),$$

where α_{NeRF} is the NeRF's occupancy, α_{GT} is the occupancy of the specified proxy, d is the distance to the proxy's surface, and σ_S is a hyperparameter that controls the leniency of the constraint.

Figure 4: **Set-the-Scene results.** The same proxy setting can be used to create different styles of the same scene. The scene prompts are shown for each generated scene, and corresponding prompts are used for each object. For example "a kid bedroom style wardrobe, closed doors" or "a baroque chair".

3.3. Post-Training Editing

Given a generated scene, one might wish to modify some aspects of it. We propose several different tools for refining and editing a generated scene.

Placement Editing The composable formulation of Set-the-Scene inherently allows editing objects' placement in the scene. This is done by changing the proxy location and updating the rays accordingly during rendering. The same technique can also easily duplicate or remove objects.

Shape Editing To modify an object's geometry, we simply edit the proxy's geometry and then fine-tune the scene for more iterations. This allows defining shape edits without having to extract a mesh from the implicit NeRF representation. Only the weights of the relevant NeRF are updated in the fine-tuning, and we alternate between rendering it in its canonical coordinate system $\mathcal{O}_i$ and with the rest of the scene in $\mathcal{S}_i$. Scene-level iterations are key to ensuring that the object remains consistent with the scene when edited.

Color Editing Finally, we show that one may also edit the color scheme of a generated object. This is done by using an architecture where the density and albedo predictions are separable, and the albedo is predicted using a set of additional fully-connected layers. During fine-tuning we can then optimize the albedo layers to guarantee that the generated shape will not change while modifying its color.

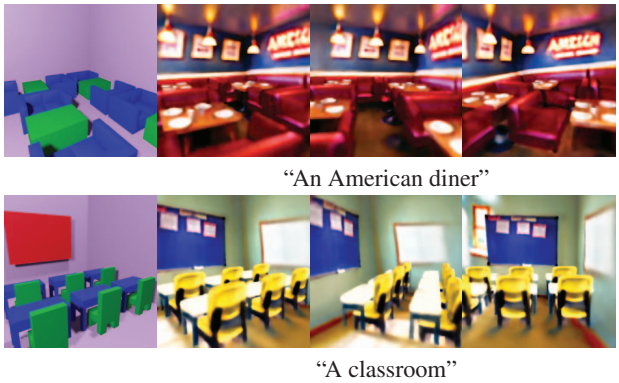


Figure 5: **Scene generation results.** Our method is able to handle complex scenes with multiple repeating objects.

4. Experiments

We now turn to a set of experiments that validate and highlight the generation capabilities of Set-the-Scene.

Implementation Details Our method uses the Stable-Diffusion 2.0 model [21] implemented in *Diffusers* [25]. For score-distillation, we use [14]. During training we iteratively pass over the objects one after the other and apply 10 training steps for each object, global iterations are interleaved between the objects. We train for about 15K iterations, with about 5K of them being global iterations.

Figure 6: **Objects with no shape priors.** For some objects it is beneficial to define only their location without explicitly stating their shape. Here we highlight how different objects can be generated on top of the table.

4.1. Scene Generation

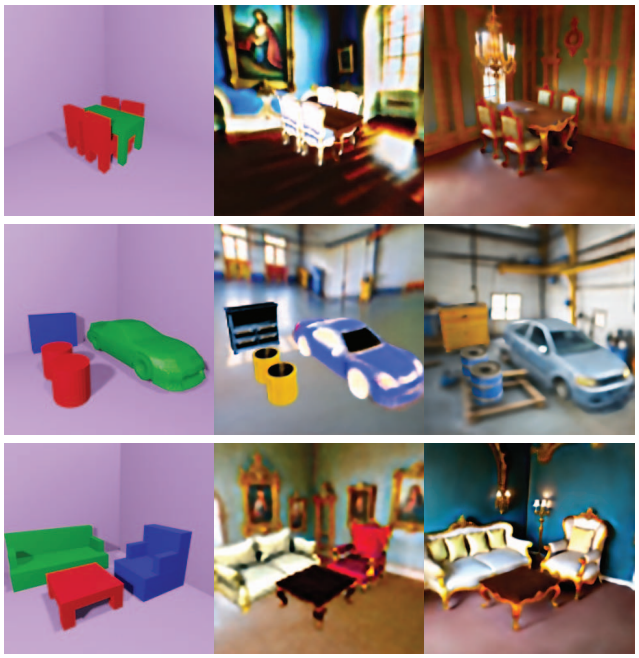
Qualitative Results Figure 4 shows results of generated scenes guided by different text prompts. One can see that our method closely follows the given proxies in terms of location and coarse shape. For example, observe how in the first row the wardrobe and bed are consistently placed in the scene even when the guiding text prompt changes. Nevertheless, our method is still able to expressively alter the style of the generated shapes according to the guiding text prompt given the geometric constraints, thus offering both control and expressiveness. In Figure 5, the objects are used multiple times within a single scene. This is done by defining a set of object proxies that share the same object NeRF, explicitly enforcing similarity between a set of duplicated objects in a scene, which is difficult to achieve when generating a scene directly without object proxies. This design choice also enables aggregating information about how an object is viewed from different locations and viewing angles into a single NeRF model, instead of training each model separately. Figure 6 shows results where an object in the scene is generated without any specific geometry constraint, but rather only with its respective location in the scene and a guiding text prompt, showing the flexibility of our controls. Finally, Figure 7 presents the exact text prompts used for a specific scene along with the convergence of each object and the scene as a whole during the optimization process.

Comparisons Recent text-to-3D methods generate the scene as a whole and do not utilize a composable representation. This makes it harder to control and can cause it to fail on complex scenes. To highlight this issue, Figure 8 shows a result of Latent-NeRF [14] on our scene text prompt. Observe how Latent-NeRF struggles with generating the complex scene. We note that although methods like Dreamfusion [19] might be able to generate better scenes due to their larger diffusion model [22], they still would not provide explicit control. Furthermore, as Latent-NeRF is the basis of our local optimization process, comparison to it highlights the improvements gained using our scheme.

Figure 7: **Convergence process of Set-the-Scene** for different NeRFs composing the scene. The bottom row shows the entire scene. A specific text prompt is given for each object, specified below each row of images.

Figure 8: **Comparison with Latent-NeRF.** When generating “A princess bedroom“ Latent-NeRF struggles with generating the complex scene.

Ablation Study Next, we perform an ablation study over our proposed Global-Local training scheme. Figure 9 shows results for the same proxy configuration with and without the global optimization phase, which makes several important aspects of joint optimization noticeable. First, observe



Local Training | Global-Local Training

Figure 9: **Impact of Global-Local training.** Here we show the importance of global training for three different scenes. Observe how our Global-Local training policy results in more coherent scenes, compared to scenes where each object is optimized independently that are inconsistent.

how color schemes between objects do not match well without joint optimization steps, for example, the sofa set which is generated in two different colors. This supports the claim that without global optimization steps the model will not be able to match the objects well. Moreover, the results generated without applying global optimization steps do not blend well within the scene and tend to look like they have been pasted over, which is expected for objects that were trained separately. By contrast, our method generates globally consistent scenes with realistic shadows generated on the background object, resulting in more natural renderings.

User Study We conducted a user study to quantitatively assess the effectiveness of our training scheme. We chose 10 scenes and applied both our Global-Local and Local-Only training schemes. We then gave 40 participants two sets of tasks. In the ranking task, participants are presented with two results side-by-side (in random order) and are asked to choose their preferred result with respect to two aspects: (a) realism; and (b) compatibility between the objects in the scene. In the second set, participants are presented with only a single result and are asked to rank it on a scale of 1-5 with respect to (a) realism; (b) object compatibility; and (c) text fidelity. All results are shown as short videos showing the moving scene to allow participants to better evaluate the generated results. Table 1 shows the outcome of our study. One can see that using our Global-Local method is superior to a Local-Only solution and results in generally higher scores for the generated images.

	Local Training	Global-Local Training
Realism Rank (↑)	19.2%	**80.8%**
Compatibility Rank (↑)	20.5%	**79.5%**
Realism Score (↑)	1.92 ± 0.28	$\mathbf{3.48 \pm 0.36}$
Compatibility Score (↑)	2.91 ± 0.40	$\mathbf{4.2 \pm 0.33}$
Text Fidelity Score (↑)	2.69 ± 0.34	$\mathbf{3.94 \pm 0.35}$

Table 1: **User study results.** Each respondent is asked to rank both Local-Only and Global-Local training with respect to various aspects.

4.2. Scene Editing

Having shown our scene generation capabilities, we now turn to evaluate our ability to edit already-generated shapes.

Placement Editing Figure 10 shows some examples of placement editing. Due to our composable representation, Set-the-Scene inherently allows for editing of object placement after training by editing the object proxies, adding new proxies, or even removing existing ones. Note that this can be done without any additional fine-tuning steps. One can additionally apply some fine-tuning steps to further improve the result with respect to the new placement.

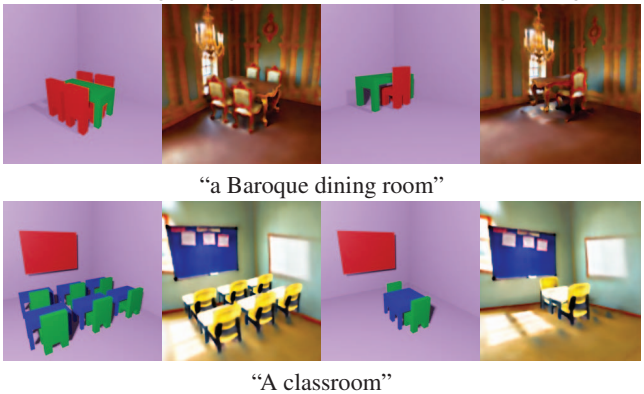


Figure 10: **Location Editing.** Here we show how the composable nature of our representation allows removing or moving objects around. Given an optimized scene, we can simply move or remove each proxy, and still produce high-fidelity results of the same scene.

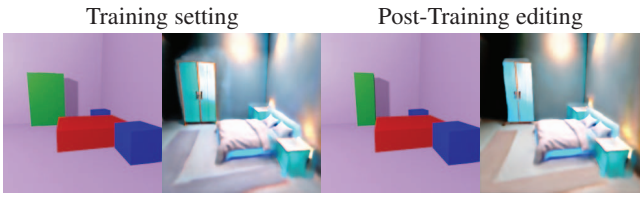


Figure 11: **Geometric Editing**. (left) Input scene, (right) Edited scene. Observe how narrowing the closet proxy shapes results in a similar effect on the generated closet, while keeping the rest of the scene content intact.

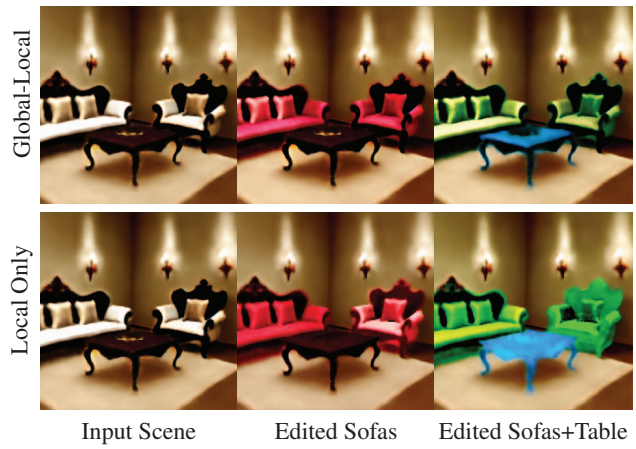


Figure 12: **Color editing.** Using a dedicated albedo head, we can fine-tune the NeRFs to change the color scheme while keeping the geometry and other objects unchanged. For each edited object the object text prompt was changed according to the desired color, for example the prompt "a baroque sofa'' was changed to ''"a red baroque sofa".

Geometry Editing Figure 11 shows how we can edit the geometry of a specific NeRF within the scene by changing its corresponding object, which allows for easy and intuitive control over the implicit NeRF. As the proxy geometry is based on a user-defined mesh, the corresponding mesh can be easily edited with existing tools such as MeshLab [4] or Blender [5]. One can see that after fine-tuning the edited result remains faithful to the original scene.

Appearance Editing Finally, Figure 12 demonstrates that our method can easily change the appearance of specified objects in the scene independently of other objects or the geometry of the edited object. Our method is able to change the color scheme of both sofas while keeping them well-matched with one another. This is not the case when optimizing without global iterations, as also shown in the same Figure. Observe for example how the exact shade of red does not much well when using only local iterations.

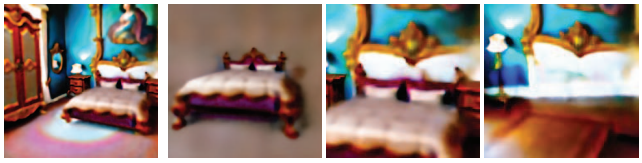


Figure 13: **Limitations.** In the scene view, the golden headboard is seen as a part of the bed. In the object view, the bed is seen as a complete object without it.

Limitations. While our experiments show the capabilities of our approach, it is worthwhile to note that there are still some limitations. First of all, the quality of our results is governed by the local generation process which is based on score distillation with a latent diffusion model. This approach still generally lacks in terms of quality and resolution, and in turn, limits our generation's capabilities. Another limitation is that the objects in the scene may be generated in the background NeRF as textures, without a corresponding geometry, as shown in Figure 13. We believe that this occurs due to the limited viewing angles available when optimizing an indoor scene. Finally, our convergence time is also governed by the number of unique NeRFs in the scene, where adding more objects requires more optimization time, as each object requires its own local iterations.

5. Conclusion

In this paper we have presented Set-the-Scene, a method for generating controllable scenes using a composable NeRF representation. Our method builds upon two main techniques. First, a composable NeRF representation where each object in the scene is represented using a dedicated NeRF with its placement controlled using an object proxy. Second, a Global-Local optimization process. Building on the flexibility of the composable representation, each object is locally optimized based on its text prompt, while also getting optimized globally with the rest of the scene. This allows us to create complete scenes, with objects that are consistent with one another.

Set-the-Scene is shown to be superior compared to existing text-to-3D solutions in terms of control and editing capabilities. Additionally, our method can easily be applied on top of any existing optimization-based single-object text-to-3D solution, enhancing them with several forms of control over the object proxies, as well as easy-to-use editing techniques that require little to no fine-tuning. We believe Set-the-Scene to be a useful step forward towards a more controllable text-to-3D future.

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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. 2

[2] Yongwei Chen, Rui Chen, Jiabao Lei, Yabin Zhang, and Kui Jia. Tango: Text-driven photorealistic and robust 3d stylization via lighting decomposition. *arXiv preprint arXiv:2210.11277*, 2022. 2

[3] Zhaoxi Chen, Guangcong Wang, and Ziwei Liu. Scenedreamer: Unbounded 3d scene generation from 2d image collections. *arXiv preprint arXiv:2302.01330*, 2023. 3

[4] Paolo Cignoni, Massimiliano Corsini, and Guido Ranzuglia. Meshlab: an open-source 3d mesh processing system. *ERCIM News*, 2008(73), 2008. 8

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

[6] Rafail Fridman, Amit Abecasis, Yoni Kasten, and Tali Dekel. Scenescape: Text-driven consistent scene generation. *arXiv preprint arXiv:2302.01133*, 2023. 3

[7] Michelle Guo, Alireza Fathi, Jiajun Wu, and Thomas Funkhouser. Object-centric neural scene rendering. *arXiv preprint arXiv:2012.08503*, 2020. 3

[8] Ajay Jain, Ben Mildenhall, Jonathan T. Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. 2022. 2

[9] Nasir Mohammad Khalid, Tianhao Xie, Eugene Belilovsky, and Popa Tiberiu. Clip-mesh: Generating textured meshes from text using pretrained image-text models. *SIGGRAPH Asia 2022 Conference Papers*, 2022. 2

[10] Jing Yu Koh, Harsh Agrawal, Dhruv Batra, Richard Tucker, Austin Waters, Honglak Lee, Yinfei Yang, Jason Baldridge, and Peter Anderson. Simple and effective synthesis of indoor 3d scenes. *arXiv preprint arXiv:2204.02960*, 2022. 3

[11] Abhijit Kundu, Kyle Genova, Xiaoqi Yin, Alireza Fathi, Caroline Pantofaru, Leonidas J Guibas, Andrea Tagliasacchi, Frank Dellaert, and Thomas Funkhouser. Panoptic neural fields: A semantic object-aware neural scene representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12871–12881, 2022. 3

[12] Ting-Hsuan Liao, Songwei Ge, Yiran Xu, Yao-Chih Lee, Badour AlBahar, and Jia-Bin Huang. Text-driven visual synthesis with latent diffusion prior. *arXiv preprint arXiv:2302.08510*, 2023. 2

[13] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. *arXiv preprint arXiv:2211.10440*, 2022. 1, 2

[14] Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-nerf for shape-guided generation of 3d shapes and textures. In *CVPR*, 2023. 1, 2, 4, 5, 6

[15] Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. Text2mesh: Text-driven neural stylization for meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13492–13502, 2022. 2

[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. *Communications of the ACM*, 65(1):99–106, 2021. 2, 4

[17] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022. 2

[18] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11453–11464, 2021. 3

[19] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022. 1, 2, 4, 6

[20] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 2

[21] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021. 2, 5

[22] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding, 2022. 6

[23] Aditya Sanghi, Hang Chu, Joseph G Lambourne, Ye Wang, Chin-Yi Cheng, and Marco Fumero. Clip-forge: Towards zero-shot text-to-shape generation. *arXiv preprint arXiv:2110.02624*, 2021. 2

[24] Jiaxiang Tang, Xiaokang Chen, Jingbo Wang, and Gang Zeng. Compressible-composable nerf via rank-residual decomposition. *arXiv preprint arXiv:2205.14870*, 2022. 3

[25] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers, 2022. 5

[26] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A Yeh, and Greg Shakhnarovich. Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. *arXiv preprint arXiv:2212.00774*, 2022. 1, 2

[27] Yinghao Xu, Menglei Chai, Zifan Shi, Sida Peng, Ivan Skorokhodov, Aliaksandr Siarohin, Ceyuan Yang, Yujun Shen, Hsin-Ying Lee, Bolei Zhou, et al. Discoscene: Spatially disentangled generative radiance fields for controllable 3d-

aware scene synthesis. *arXiv preprint arXiv:2212.11984*, 2022. 3

[28] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13779–13788, 2021. 3

[29] Xiaoshuai Zhang, Abhijit Kundu, Thomas Funkhouser, Leonidas Guibas, Hao Su, and Kyle Genova. Nerflets: Local radiance fields for efficient structure-aware 3d scene representation from 2d supervisio. In *CVPR*, 2023. 3