Disentangled 3D Scene Generation with Layout Learning

Dave Epstein [1 2] **Ben Poole** [2] **Ben Mildenhall** [2] **Alexei A. Efros** [1] **Aleksander Holynski** [1 2]

Abstract

We introduce a method to generate 3D scenes that are disentangled into their component objects. This disentanglement is unsupervised, relying only on the knowledge of a large pretrained text-to-image model. Our key insight is that objects can be discovered by finding parts of a 3D scene that, when rearranged spatially, still produce valid configurations of the same scene. Concretely, our method jointly optimizes multiple NeRFs from scratch—each representing its own object—along with a *set of layouts* that composite these objects into scenes. We then encourage these composited scenes to be in-distribution according to the image generator. We show that despite its simplicity, our approach successfully generates 3D scenes decomposed into individual objects, enabling new capabilities in text-to-3D content creation. See our project page for results and an interactive demo: https://dave.ml/layoutlearning/

1. Introduction

A remarkable ability of many seeing organisms is object individuation (Piaget et al., 1952), the ability to discern separate objects from light projected onto the retina (Wertheimer, 1938). Indeed, from a very young age, humans and other creatures are able to organize the physical world they perceive into the three-dimensional entities that comprise it (Spelke, 1990; Wilcox, 1999; Hoffmann et al., 2011). The analogous task of object discovery has captured the attention of the artificial intelligence community from its very inception (Roberts, 1963; Ohta et al., 1978), since agents that can autonomously parse 3D scenes into their component objects are better able to navigate and interact with their surroundings.

Fifty years later, generative models of images are advancing at a frenzied pace (Nichol et al., 2021; Ramesh et al., 2022; Saharia et al., 2022; Yu et al., 2022; Chang et al., 2023). While these models can generate high-quality samples, their internal workings are hard to interpret, and they do not explicitly represent the distinct 3D entities that make up the images they create. Nevertheless, the priors learned by these models have proven incredibly useful across various tasks involving 3D reasoning (Hedlin et al., 2023; Ke et al., 2023; Liu et al., 2023; Luo et al., 2023; Wu et al., 2023), suggesting that they may indeed be capable of decomposing generated content into the underlying 3D objects depicted.

One particularly exciting application of these text-to-image networks is 3D generation, leveraging the rich distribution learned by a diffusion model to optimize a 3D representation, *e.g.* a neural radiance field (NeRF, Mildenhall et al., 2020), such that rendered views resemble samples from the prior. This technique allows for text-to-3D generation without any 3D supervision (Poole et al., 2022; Wang et al., 2023b), but most results focus on simple prompts depicting just one or two isolated objects (Lin et al., 2023; Wang et al., 2023c).

Our method builds on this work to generate complex scenes that are automatically disentangled into the objects they contain. To do so, we instantiate and render *multiple NeRFs* for a given scene instead of just one, encouraging the model to use each NeRF to represent a separate 3D entity. At the crux of our approach is an intuitive definition of objects as parts of a scene that can be manipulated independently of others while keeping the scene "well-formed" (Biederman, 1981). We implement this by learning a set of different layouts—3D affine transformations of every NeRF—which must yield composited scenes that render into in-distribution 2D images given a text prompt (Poole et al., 2022).

We find that this lightweight inductive bias, which we term *layout learning*, results in surprisingly effective object disentanglement in generated 3D scenes (Figure 1), enabling object-level scene manipulation in the text-to-3D pipeline. We demonstrate the utility of layout learning on several tasks, such as building a scene around a 3D asset of interest, sampling different plausible arrangements for a given set of assets, and even parsing a provided NeRF into the objects it contains, all without any supervision beyond just a text prompt. We further quantitatively verify that, despite requiring no auxiliary models or per-example human annotation, the object-level decomposition that emerges through layout learning is meaningful and outperforms baselines.

[1]Department of Computer Science, UC Berkeley [2]Google Research. Correspondence to: <dave@eecs.berkeley.edu>.

Figure 1: **Layout learning generates disentangled 3D scenes** given a text prompt and a pretrained text-to-image diffusion model. We learn an entire 3D scene (left, shown from two views along with surface normals and a textureless render) that is composed of multiple NeRFs (right) representing different objects and arranged according to a learned layout.

Our key contributions are as follows:

- We introduce a simple, tractable definition of objects as portions of a scene that can be manipulated independently of each other and still produce valid scenes.
- We incorporate this notion into the architecture of a neural network, enabling the compositional generation of 3D scenes by optimizing a set of NeRFs as well as a set of layouts for these NeRFs.
- We apply layout learning to a range of novel 3D scene generation and editing tasks, demonstrating its ability to disentangle complex data despite requiring no object labels, bounding boxes, fine-tuning, external models, or any other form of additional supervision.

2. Background

2.1. Neural 3D representations

To output three-dimensional scenes, we must use an architecture capable of modeling 3D data, such as a neural radiance field (NeRF, Mildenhall et al., 2020). We build on MLP-based NeRFs (Barron et al., 2021), that represent a volume using an MLP f that maps from a point in 3D space $\boldsymbol{\mu}$ to a density τ and albedo $\boldsymbol{\rho}$:

$$(\tau, \boldsymbol{\rho}) = f(\boldsymbol{\mu}; \theta).$$

We can differentiably render this volume by casting a ray $\mathbf{r}$ into the scene, and then alpha-compositing the densities and colors at sampled points along the ray to produce a color and accumulated alpha value. For 3D reconstruction, we would optimize the colors for the rendered rays to match a known pixel value at an observed image and camera pose, but for 3D generation we sample a random camera pose, render the corresponding rays, and score the resulting image using a generative model.

2.2. Text-to-3D using 2D diffusion models

Our work builds on text-to-3D generation using 2D diffusion priors (Poole et al., 2022). These methods turn a diffusion model into a loss function that can be used to optimize the parameters of a 3D representation. Given an initially random set of parameters θ, at each iteration we randomly sample a camera c and render the 3D model to get an image $x = g(\theta, c)$. We can then score the quality of this rendered image given some conditioning text y by evaluating the score function of a noised version of the image $z_t = \alpha_t x + \sigma_t \epsilon$ using the pretrained diffusion model $\hat{\epsilon}(z_t; y, t)$. We update the parameters of the 3D representation using score distillation:

$$\nabla_\theta \mathcal{L}_{\text{SDS}}(\theta) = \mathbb{E}_{t,\epsilon,c}\left[w(t)(\hat{\epsilon}(z_t; y, t) - \epsilon)\frac{\partial x}{\partial \theta}\right] \quad (1)$$

where $w(t)$ is a noise-level dependent weighting.

SDS and related methods enable the use of rich 2D priors obtained from large text-image datasets to inform the structure of 3D representations. However, they often require careful tuning of initialization and hyperparameters to yield high quality 3D models, and past work has optimized these towards object generation. The NeRF is initialized with a Gaussian blob of density at the origin, biasing the optimization process to favor an object at the center instead of placing density in a skybox-like environment in the periphery of the 3D representation. Additionally, bounding spheres are used to prevent creation of density in the background. The resulting 3D models can produce high-quality individual objects, but often fail to generate interesting scenes, and the resulting 3D models are a single representation that cannot be easily split apart into constituent entities.

3. Method

To bridge the gap from monolithic 3D representations to scenes with multiple objects, we introduce a more expressive 3D representation. Here, we learn multiple NeRFs along with a set of layouts, *i.e.* valid ways to arrange these NeRFs in 3D space. We transform the NeRFs according to these layouts and composite them, training them to form high-quality scenes as evaluated by the SDS loss with a text-to-image prior. This structure causes each individual NeRF to represent a different object while ensuring that the composite NeRF represents a high-quality scene. See Figure 2 for an overview of our approach.

3.1. Compositing multiple volumes

We begin by considering perhaps the most naïve approach to generating 3D scenes disentangled into separate entities. We simply declare K NeRFs $\{f_k\}$—each one intended to house its own object—and jointly accumulate densities from all NeRFs along a ray, proceeding with training as normal by rendering the composite volume. This can be seen as an analogy to set-latent representations (Locatello et al., 2020; Jaegle et al., 2021a;b; Jabri et al., 2023), which have been widely explored in other contexts. In this case, rather than arriving at the final albedo $\boldsymbol{\rho}$ and density τ of a point $\boldsymbol{\mu}$ by querying one 3D representation, we query K such representations, obtaining a set $\{\boldsymbol{\rho}_k, \tau_k\}_{k=1}^K$. The final density at $\boldsymbol{\mu}$ is then $\tau' = \sum \tau_k$ and the final albedo is the density-weighted average $\boldsymbol{\rho}' = \sum \frac{\tau_k}{\tau'} \boldsymbol{\rho}_k$.

This formulation provides several potential benefits. First, it may be easier to optimize this representation to generate a larger set of objects, since there are K distinct 3D Gaussian density spheres to deform at initialization, not just one. Second, many representations implicitly contain a local smoothness bias (Tancik et al., 2020) which is helpful for

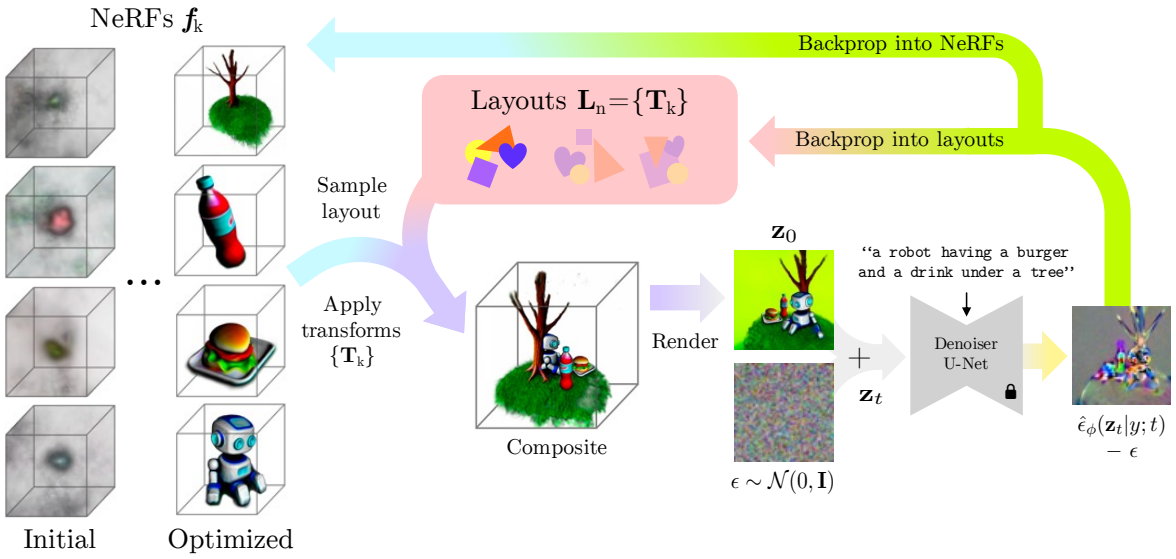

Figure 2: **Method.** Layout learning works by optimizing K NeRFs f_k and learning N different layouts $\mathbf{L}_n$ for them, each consisting of per-NeRF affine transforms $\mathbf{T}_k$. Every iteration, a random layout is sampled and used to transform all NeRFs into a shared coordinate space. The resultant volume is rendered and optimized with score distillation sampling (Poole et al., 2022) as well as per-NeRF regularizations to prevent degenerate decompositions and geometries (Barron et al., 2022). This simple structure causes object disentanglement to emerge in generated 3D scenes.

generating objects but not spatially discontinuous scenes. Thus, our representation might be inclined toward allocating each representation toward a spatially smooth entity, *i.e.* an object.

However, just as unregularized sets of latents are often highly uninterpretable, simply spawning K instances of a NeRF does not produce meaningful decompositions. In practice, we find each NeRF often represents a random point-cloud-like subset of 3D space (Fig. 3).

To produce scenes with disentangled objects, we need a method to encourage each 3D instance to represent a coherent object, not just a different part of 3D space.

3.2. Layout learning

We are inspired by other unsupervised definitions of objects that operate by imposing a simple inductive bias or regularization in the structure of a model's latent space, *e.g.* query-axis softmax attention (Locatello et al., 2020), spatial ellipsoid feature maps (Epstein et al., 2022), and diagonal Hessian matrices (Peebles et al., 2020). In particular, Niemeyer & Geiger (2021) learn a 3D-aware GAN that composites multiple NeRF volumes in the forward pass, where the latent code contains a random affine transform for each NeRF's output. Through this structure, each NeRF learns to associate itself with a different object, facilitating the kind of disentanglement we are after. However, their approach relies on pre-specified independent distributions of each object's location, pose, and size, preventing scaling beyond narrow datasets of images with one or two objects and minimal variation in layout.

In our setting, not only does the desired output comprise numerous open-vocabulary, arbitrary objects, but these objects *must be arranged in a particular way* for the resultant scene to be valid or "well-formed" (Biederman et al., 1982). Why not simply learn this arrangement?

To do this, we equip each individual NeRF f_k with its own learnable affine transform $\mathbf{T}_k$, and denote the set of transforms across all volumes a layout $\mathbf{L} \equiv \{\mathbf{T}_k\}_{k=1}^K$. Each $\mathbf{T}_k$ has a rotation $\mathbf{R}_k \in \mathbb{R}^{3\times3}$ (in practice expressed via a quaternion $\mathbf{q} \in \mathbb{R}^4$ for ease of optimization), translation $\mathbf{t}_k \in \mathbb{R}^3$, and scale $s_k \in \mathbb{R}$. We apply this affine transform to the camera-to-world rays $\mathbf{r}$ before sampling the points used to query f_k. This implementation is simple, makes no assumptions about the underlying form of f, and updates parameters with standard backpropagation, as sampling and embedding points along the ray is fully differentiable (Lin et al., 2021). Concretely, a ray $\mathbf{r}$ with origin $\mathbf{o}$ and direction $\mathbf{d}$ is transformed into an instance-specific ray $\mathbf{r}_k$ via the following transformations:

$$\mathbf{o}_k = s_k\left(\mathbf{R}_k\mathbf{o} - \mathbf{t}_k\right) \tag{2}$$

$$\mathbf{d}_k = s_k\mathbf{R}_k\mathbf{d} \tag{3}$$

$$\mathbf{r}_k(t) = \mathbf{o}_k + t\mathbf{d}_k \tag{4}$$

Though we input a different $H \times W$ grid of rays to each f_k, we composite their outputs as if they all sit in the same

Method	Average Score ↑ CLIP B/16 Color	Geo	CLIP L/14 Color	Geo
Random objects	23.4	22.4	17.2	18.3
Per-object SDS	32.3	30.5	27.2	25.9
K NeRFs	26.7	25.4	21.0	21.2
+ Per-NeRF losses	27.3	26.1	21.6	22.6
+ Empty NeRF loss	27.7	26.2	22.8	23.2
+ Learn layout	29.9	28.8	24.9	23.5
+ Learn N layouts	**31.3**	**29.9**	**27.1**	**24.8**
Relative layouts	30.4	29.2	25.7	24.0
View dep. prompt	31.0	29.1	25.6	23.6

Figure 3: **Evaluating disentanglement and quality.** We optimize a model with $K = 3$ NeRFs on a list of 30 prompts, each containing three objects. We then automatically pair each NeRF with a description of one of the objects in the prompt and report average NeRF-object CLIP score (see text for details). We also generate each of the $30 \times 3 = 90$ objects from the prompt list individually and compute its score with both the corresponding prompt and a random other one, providing upper and lower bounds for performance on this task. Training K NeRFs provides some decomposition, but most objects are scattered across 2 or 3 models. Learning one layout alleviates some of these issues, but only with multiple layouts do we see strong disentanglement. We show two representative examples of emergent objects to visualize these differences.

coordinate space—for example, the final density at $\boldsymbol{\mu} = \mathbf{r}(t)$ is the sum of densities output by every f_k at $\boldsymbol{\mu}_k = \mathbf{r}_k(t)$.

Compared to the naïve formulation that instantiates K models with identical initial densities, learning the size, orientation, and position of each model makes it easier to place density in different parts of 3D space. In addition, the inherent stochasticity of optimization may further dissuade degenerate solutions.

While introducing layout learning significantly increases the quality of object disentanglement (Tbl. 3b), the model is still able to adjoin and utilize individual NeRFs in undesirable ways. For example, it can still place object parts next to each other in the same way as K NeRFs without layout learning.

Learning multiple layouts. We return to our statement that objects must be "arranged in a particular way" to form scenes that render to in-distribution images. While we already enable this with layout learning in its current form, we are not taking advantage of one key fact: there are many "particular ways" to arrange a set of objects, each of which gives an equally valid composition. Rather than only learning one layout, we instead learn a distribution over layouts $P(\mathbf{L})$ or a set of N randomly initialized layouts $\{\mathbf{L}_n\}_{n=1}^{N}$. We opt for the latter, and sample one of the N layouts from the set at each training step to yield transformed rays $\mathbf{r}_k$.

With this in place, we have arrived at our final definition of objectness (Figure 2): **objects are parts of a scene that can be arranged in different ways to form valid compositions.** We have "parts" by incorporating multiple volumes, and "arranging in different ways" through multiple-layout learning. This simple approach is easy to implement (Fig. 9), adds very few parameters ($8NK$ to be exact), requires no fine-tuning or manual annotation, and is agnostic to choices of text-to-image and 3D model. In Section 4, we verify that layout learning enables the generation and disentanglement of complex 3D scenes.

Regularization. We build on Mip-NeRF 360 (Barron et al., 2022) as our 3D backbone, inheriting their orientation, distortion, and accumulation losses to improve visual quality of renderings and minimize artifacts. However, rather than computing these losses on the final composited scene, we apply them on a per-NeRF basis. Importantly, we add a loss penalizing degenerate empty NeRFs by regularizing the soft-binarized version of each NeRF's accumulated density, $\boldsymbol{\alpha}_{\text{bin}}$, to occupy at least 10% of the canvas:

$$\mathcal{L}_{\text{empty}} = \max\left(0.1 - \bar{\boldsymbol{\alpha}}_{\text{bin}}, 0\right) \tag{5}$$

We initialize parameters $s \sim \mathcal{N}(1, 0.3)$, $\mathbf{t}^{(i)} \sim \mathcal{N}(0, 0.3)$, and $\mathbf{q}^{(i)} \sim \mathcal{N}(\mu_i, 0.1)$ where μ_i is 1 for the last element and 0 for all others. We use a $10\times$ higher learning rate to train layout parameters. See Appendix A.1 for more details.

4. Experiments

We examine the ability of layout learning to generate and disentangle 3D scenes across a wide range of text prompts. We

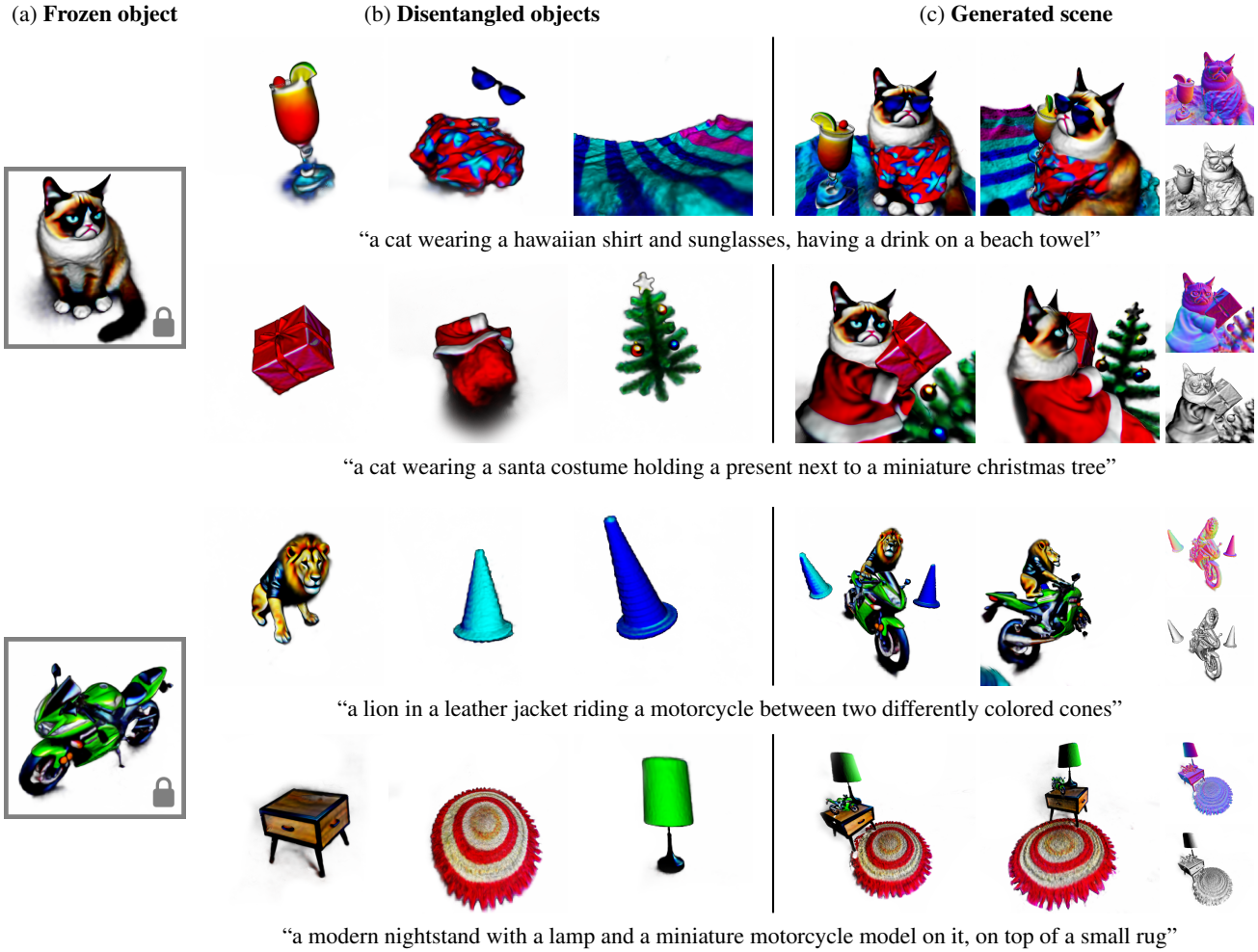

Figure 4: **Conditional optimization.** We can take advantage of our structured representation to learn a scene given a 3D asset in addition to a text prompt, such as a specific cat or motorcycle **(a)**. By freezing the NeRF weights but not the layout weights, the model learns to arrange the provided asset in the context of the other objects it discovers **(b)**. We show the entire composite scenes the model creates in **(c)** from two views, along with surface normals and a textureless render.

first verify our method's effectiveness through an ablation study and comparison to baselines, and then demonstrate various applications enabled by layout learning.

4.1. Qualitative evaluation

In Figure 1, we demonstrate several examples of our full system with layout learning. In each scene, we find that the composited 3D generation is high-quality and matches the text prompt, while the individual NeRFs learn to correspond to objects within the scene. Interestingly, since our approach does not directly rely on the input prompt, we can disentangle entities not mentioned in the text, such as a basket filled with easter eggs, a chef's hat, and a picnic table.

4.2. Quantitative evaluation

Measuring the quality of text-to-3D generation remains an open problem due to a lack of ground truth data—there is no "true" scene corresponding to a given prompt. Similarly, there is no true disentanglement for a certain text description. Following Park et al. (2021); Jain et al. (2022); Poole et al. (2022), we attempt to capture both of these aspects using scores from a pretrained CLIP model (Radford et al., 2021; Li et al., 2017). Specifically, we create a diverse list of 30 prompts, each containing 3 objects, and optimize a model with $K = 3$ NeRFs on each prompt. We compute the 3×3 matrix of CLIP scores ($100\times$ cosine similarity) for each NeRF with descriptions "a DSLR photo of [object 1/2/3]", finding the optimal NeRF-to-object matching and reporting the average score across all 3 objects.

We also run SDS on the $30 \times 3 = 90$ per-object prompts individually and compute scores, representing a maximum attainable CLIP score under perfect disentanglement (we equalize parameter counts across all models for fairness).

As a low-water mark, we compute scores between per-object NeRFs and a random other prompt from the pool of 90.

The results in Table 3b show these CLIP scores, computed both on textured ("Color") and textureless, geometry-only ("Geo") renders. The final variant of layout learning achieves competitive performance, only 0.1 points away from supervised per-object rendering when using the largest CLIP model as an oracle, indicating high quality of both object disentanglement and appearance. Please see Appendix A.3 for a complete list of prompts and more details.

Ablation. We justify the sequence of design decisions presented in Section 3 by evaluating different variants of layout learning, starting from a simple collection of K NeRFs and building up to our final architecture. The simple setting leads to some non-trivial separation (Figure 3a) but parts of objects are randomly distributed across NeRFs—CLIP scores are significantly above random, but far below the upper bound. Adding regularization losses improve scores somewhat, but the biggest gains come from introducing layout learning and then co-learning N different arrangements, validating our approach.

4.3. Applications of layout learning

To highlight the utility of the disentanglement given by layout learning beyond generation, we apply it to various 3D editing tasks. First, we show further results on object disentanglement in Figure 4, but in a scenario where one NeRF is frozen to contain an object of interest, and the rest of the scene must be constructed around it. This object's layout parameters can also be frozen, for example, if a specific position or size is desired. We examine the more challenging setting where layout parameters must also be learned, and show results incorporating a grumpy cat and green motorbike into different contexts. Our model learns plausible transformations to incorporate provided assets into scenes, while still discovering the other objects necessary to complete the prompt.

In Figure 5, we visualize the different layouts learned in a single training run. The variation in discovered layouts is significant, indicating that our formulation can find various meaningful arrangements of objects in a scene. This allows users of our method to explore different permutations of the same content in the scenes they generate.

Inspired by this, and to test gradient flow into layout parameters, we also examine whether our method can be used to arrange off-the-shelf, frozen 3D assets into semantically valid configurations (Figure 6). Starting from random positions, sizes, and orientations, layouts are updated using signal backpropagated from the image model. This learns reasonable transformations, such as a rubber duck shrinking and moving inside a tub, and a shower head moving upwards and pointing so its stream is going into the tub.

"two cats in fancy suits playing snooker"

"a robe, a pair of slippers, and a candle"

"two flamingos sipping on cocktails in a desert oasis"

Figure 5: **Layout diversity.** Our method discovers different plausible arrangements for objects. Here, we optimize each example over $N = 4$ layouts and show differences in composited scenes, *e.g.* flamingos wading inside vs. beside the pond, and cats in different poses around the snooker table.

Finally, we use layout learning to disentangle a pre-existing NeRF containing multiple entities, without any per-object supervision (Fig. 8). We do this by randomly initializing a new model and training it with a caption describing the target NeRF. We require the first layout $\mathbf{L}_1$ to create a scene that faithfully reconstructs the target NeRF in RGB space, allowing all other layouts to vary freely. We find that layout learning arrives at reasonable decompositions of the scenes it is tasked with reconstructing.

5. Related work

Object recognition and discovery. The predominant way to identify the objects present in a scene is to segment two-dimensional images using extensive manual annotation (Kirillov et al., 2023; Li et al., 2022; Wang et al., 2023a), but relying on human supervision introduces challenges and scales poorly to 3D data. As an alternative, an extensive line of work on *unsupervised* object discovery (Russell et al., 2006; Rubinstein et al., 2013; Oktay et al., 2018; Hénaff et al., 2022; Smith et al., 2022; Ye et al., 2022; Monnier et al., 2023) proposes different inductive biases (Locatello et al., 2019) that encourage awareness of objects in a scene. However, these approaches are largely restricted to either 2D images or constrained 3D data (Yu et al., 2021; Sajjadi et al., 2022), limiting their applicability to complex 3D scenes. At the same time, large text-to-image models have been shown to implicitly encode an understanding of entities in their

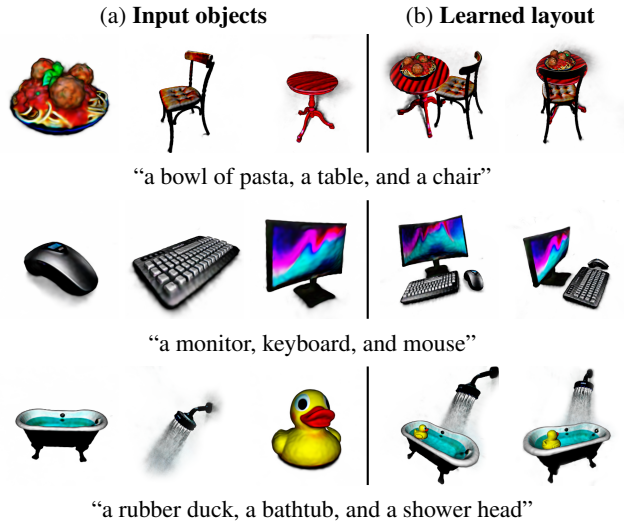

Figure 6: **Optimizing layout.** Allowing gradients to flow only into layout parameters while freezing a set of provided 3D assets results in reasonable object configurations, such as a chair tucked into a table with spaghetti on it, despite no such guidance being provided in the text conditioning.

internals (Epstein et al., 2023), motivating their use for the difficult problem of explicit object disentanglement.

Compositional 3D generation. There are many benefits to generating 3D scenes separated into objects beyond just better control. For example, generating objects one at a time and compositing them manually provides no guarantees about compatibility in appearance or pose, such as "dogs in matching outfits" in Figure 1 or a lion holding the handlebars of a motorcycle in Figure 4. Previous and concurrent work explores this area, but either requires users to painstakingly annotate 3D bounding boxes and per-object labels (Cohen-Bar et al., 2023; Po & Wetzstein, 2023) or uses external supervision such as LLMs to propose objects and layouts (Yang et al., 2023; Zhang et al., 2023), significantly slowing down the generation process and hindering quality. We show that this entire process can be solved without any additional models or labels, simply using the signal provided by a pretrained image generator.

6. Discussion

We present layout learning, a simple method for generating disentangled 3D scenes given a text prompt. By optimizing multiple NeRFs to form valid scenes across multiple layouts, we encourage each NeRF to contain its own object. This approach requires no additional supervision or auxiliary models, yet performs quite well. By generating scenes that are decomposed into objects, we provide users of text-to-3D systems with more granular, local control over the complex creations output by a black-box neural network.

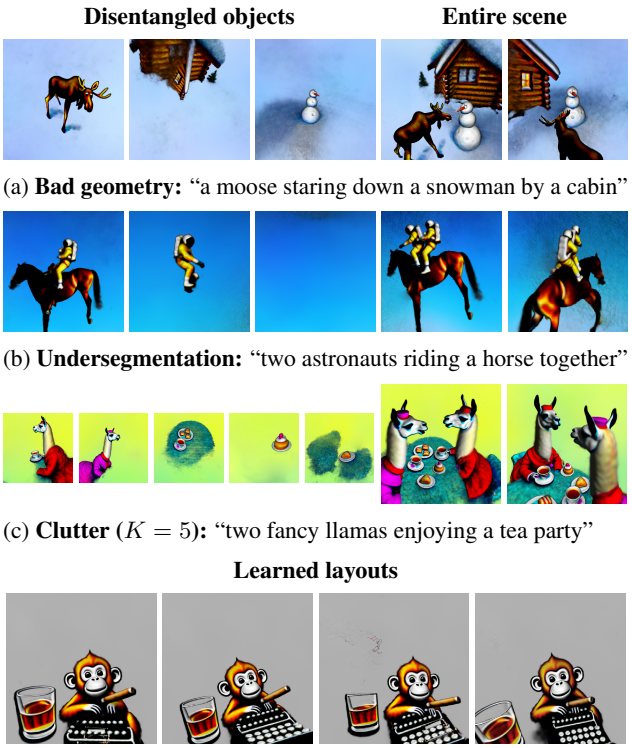

Figure 7: **Limitations.** Layout learning inherits failure modes from SDS, such as bad geometry of a cabin with oddly intersecting exterior walls **(a)**. It also may undesirably group objects that always move together **(b)** such as a horse and its rider, and **(c)** for certain prompts that generate many small objects, choosing K correctly is challenging, hurting disentanglement. In some cases **(d)**, despite different initial values, layouts converge to very similar final configurations.

Though layout learning is surprisingly effective on a wide variety of text prompts, the problem of object disentanglement in 3D is inherently ill-posed, and our definition of objects is simple. As a result, many undesirable solutions exist that satisfy the constraints we pose.

Despite our best efforts, the compositional scenes output by our model do occasionally suffer from failures (Fig. 7) such as over- or under-segmentation and the "Janus problem" (where objects are depicted so that salient features appear from all views, *e.g.* an animal with a face on the back of its head) as well as other undesirable geometries. Further, though layouts are initialized with high standard deviation and trained with an increased learning rate, they occasionally converge to near-identical values, minimizing the effectivness of our method. In general, we find that failures to disentangle are accompanied by an overall decrease in visual quality.

Acknowledgements

We thank Dor Verbin, Ruiqi Gao, Lucy Chai, and Minyoung Huh for their helpful comments, and Arthur Brussee for help with an NGP implementation. DE was partly supported by the PD Soros Fellowship. DE conducted part of this research at Google, with additional funding from an ONR MURI grant.

Impact statement

Generative models present many ethical concerns over data attribution, nefarious applications, and longer-term societal effects. Though we build on a text-to-image model trained on data that has been filtered to remove concerning imagery and captions, recent work has shown that popular datasets contain dangerous depictions of undesirable content[1] which may leak into model weights.

Further, since we distill the distribution learned by an image generator, we inherit the potential negative use-cases enabled by the original model. By facilitating the creation of more complex, compositional 3D scenes, we perhaps expand the scope of potential issues associated with text-to-3D technologies. Taking care to minimize potential harmful deployment of our generative models through using ethically-sourced and well-curated data is of the utmost importance as our field continues to grow in size and influence.

Further, by introducing an unsupervised method to disentangle 3D scenes into objects, we possibly contribute to the displacement of creative workers such as video game asset designers via increased automation. However, at the same time, methods like the one we propose have the potential to become valuable tools at the artist's disposal, providing much more control over outputs and helping create new, more engaging forms of content.

References

Barron, J. T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., and Srinivasan, P. P. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5855–5864, October 2021.

Barron, J. T., Mildenhall, B., Verbin, D., Srinivasan, P. P., and Hedman, P. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5470–5479, June 2022.

Biederman, I. On the semantics of a glance at a scene. In *Perceptual organization*, pp. 213–253. Routledge, 1981.

Biederman, I., Mezzanotte, R. J., and Rabinowitz, J. C. Scene perception: Detecting and judging objects undergoing relational violations. *Cognitive psychology*, 14(2): 143–177, 1982.

Chang, H., Zhang, H., Barber, J., Maschinot, A., Lezama, J., Jiang, L., Yang, M.-H., Murphy, K., Freeman, W. T., Rubinstein, M., et al. Muse: Text-to-image generation via masked generative transformers. In *ICML*, 2023.

Cohen-Bar, D., Richardson, E., Metzer, G., Giryes, R., and Cohen-Or, D. Set-the-scene: Global-local training for generating controllable nerf scenes. In *ICCV*, 2023.

Epstein, D., Park, T., Zhang, R., Shechtman, E., and Efros, A. A. Blobgan: Spatially disentangled scene representations. In *European Conference on Computer Vision*, pp. 616–635. Springer, 2022.

Epstein, D., Jabri, A., Poole, B., Efros, A. A., and Holynski, A. Diffusion self-guidance for controllable image generation. In *Advances in Neural Information Processing Systems*, 2023.

Gupta, V., Koren, T., and Singer, Y. Shampoo: Preconditioned stochastic tensor optimization. In *ICML*, 2018.

Hedlin, E., Sharma, G., Mahajan, S., Isack, H., Kar, A., Tagliasacchi, A., and Yi, K. M. Unsupervised semantic correspondence using stable diffusion. *arXiv preprint arXiv:2305.15581*, 2023.

Hénaff, O. J., Koppula, S., Shelhamer, E., Zoran, D., Jaegle, A., Zisserman, A., Carreira, J., and Arandjelović, R. Object discovery and representation networks. In *European Conference on Computer Vision*, pp. 123–143. Springer, 2022.

Hoffmann, A., Rüttler, V., and Nieder, A. Ontogeny of object permanence and object tracking in the carrion crow, corvus corone. *Animal behaviour*, 82(2):359–367, 2011.

Jabri, A., Fleet, D., and Chen, T. Scalable adaptive computation for iterative generation. In *ICML*, 2023.

Jaegle, A., Borgeaud, S., Alayrac, J.-B., Doersch, C., Ionescu, C., Ding, D., Koppula, S., Zoran, D., Brock, A., Shelhamer, E., et al. Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*, 2021a.

Jaegle, A., Gimeno, F., Brock, A., Vinyals, O., Zisserman, A., and Carreira, J. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pp. 4651–4664. PMLR, 2021b.

Jain, A., Mildenhall, B., Barron, J. T., Abbeel, P., and Poole, B. Zero-shot text-guided object generation with dream fields. In *CVPR*, 2022.

[1] https://crsreports.congress.gov/product/pdf/R/R47569

Ke, B., Obukhov, A., Huang, S., Metzger, N., Daudt, R. C., and Schindler, K. Repurposing diffusion-based image generators for monocular depth estimation. *arXiv preprint arXiv:2312.02145*, 2023.

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023.

Li, A., Jabri, A., Joulin, A., and van der Maaten, L. Learning visual n-grams from web data. In *ICCV*, 2017.

Li, Y., Mao, H., Girshick, R., and He, K. Exploring plain vision transformer backbones for object detection. In *European Conference on Computer Vision*, pp. 280–296. Springer, 2022.

Lin, C.-H., Ma, W.-C., Torralba, A., and Lucey, S. Barf: Bundle-adjusting neural radiance fields. In *ICCV*, 2021.

Lin, C.-H., Gao, J., Tang, L., Takikawa, T., Zeng, X., Huang, X., Kreis, K., Fidler, S., Liu, M.-Y., and Lin, T.-Y. Magic3d: High-resolution text-to-3d content creation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 300–309, 2023.

Liu, R., Wu, R., Van Hoorick, B., Tokmakov, P., Zakharov, S., and Vondrick, C. Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9298–9309, 2023.

Locatello, F., Bauer, S., Lucic, M., Raetsch, G., Gelly, S., Schölkopf, B., and Bachem, O. Challenging common assumptions in the unsupervised learning of disentangled representations. In *international conference on machine learning*, pp. 4114–4124. PMLR, 2019.

Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., and Kipf, T. Object-centric learning with slot attention. *Advances in Neural Information Processing Systems*, 33:11525–11538, 2020.

Luo, G., Dunlap, L., Park, D. H., Holynski, A., and Darrell, T. Diffusion hyperfeatures: Searching through time and space for semantic correspondence. *arXiv preprint arXiv:2305.14334*, 2023.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

Monnier, T., Austin, J., Kanazawa, A., Efros, A. A., and Aubry, M. Differentiable Blocks World: Qualitative 3D Decomposition by Rendering Primitives. In *Neural Information Processing Systems*, 2023.

Müller, T., Evans, A., Schied, C., and Keller, A. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022. doi: 10.1145/3528223.3530127. URL https://doi.org/10.1145/3528223.3530127.

Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., and Chen, M. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

Niemeyer, M. and Geiger, A. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11453–11464, 2021.

Ohta, Y.-i., Kanade, T., and Sakai, T. An analysis system for scenes containing objects with substructures. In *Proceedings of the Fourth International Joint Conference on Pattern Recognitions*, pp. 752–754, 1978.

Oktay, D., Vondrick, C., and Torralba, A. Counterfactual image networks, 2018. URL https://openreview.net/forum?id=SyYYPdg0-.

Park, D. H., Azadi, S., Liu, X., Darrell, T., and Rohrbach, A. Benchmark for compositional text-to-image synthesis. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

Peebles, W., Peebles, J., Zhu, J.-Y., Efros, A., and Torralba, A. The hessian penalty: A weak prior for unsupervised disentanglement. In *ECCV*, 2020.

Piaget, J., Cook, M., et al. *The origins of intelligence in children*, volume 8. International Universities Press New York, 1952.

Po, R. and Wetzstein, G. Compositional 3d scene generation using locally conditioned diffusion. *arXiv preprint arXiv:2303.12218*, 2023.

Poole, B., Jain, A., Barron, J. T., and Mildenhall, B. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

Roberts, L. G. *Machine perception of three-dimensional solids*. PhD thesis, Massachusetts Institute of Technology, 1963.

Rubinstein, M., Joulin, A., Kopf, J., and Liu, C. Unsupervised joint object discovery and segmentation in internet images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1939–1946, 2013.

Russell, B. C., Freeman, W. T., Efros, A. A., Sivic, J., and Zisserman, A. Using multiple segmentations to discover objects and their extent in image collections. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pp. 1605–1614. IEEE, 2006.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E. L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35: 36479–36494, 2022.

Sajjadi, M. S., Duckworth, D., Mahendran, A., van Steenkiste, S., Pavetic, F., Lucic, M., Guibas, L. J., Greff, K., and Kipf, T. Object scene representation transformer. *Advances in Neural Information Processing Systems*, 35: 9512–9524, 2022.

Smith, C., Yu, H.-X., Zakharov, S., Durand, F., Tenenbaum, J. B., Wu, J., and Sitzmann, V. Unsupervised discovery and composition of object light fields. *arXiv preprint arXiv:2205.03923*, 2022.

Spelke, E. S. Principles of object perception. *Cognitive science*, 14(1):29–56, 1990.

Tancik, M., Srinivasan, P. P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J. T., and Ng, R. Fourier features let networks learn high frequency functions in low dimensional domains. In *Neural Information Processing Systems*, 2020.

Wang, C.-Y., Bochkovskiy, A., and Liao, H.-Y. M. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7464–7475, 2023a.

Wang, H., Du, X., Li, J., Yeh, R. A., and Shakhnarovich, G. Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12619–12629, 2023b.

Wang, Z., Lu, C., Wang, Y., Bao, F., Li, C., Su, H., and Zhu, J. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *arXiv preprint arXiv:2305.16213*, 2023c.

Wertheimer, M. Laws of organization in perceptual forms. 1938.

Wilcox, T. Object individuation: Infants' use of shape, size, pattern, and color. *Cognition*, 72(2):125–166, 1999.

Wu, R., Mildenhall, B., Henzler, P., Park, K., Gao, R., Watson, D., Srinivasan, P. P., Verbin, D., Barron, J. T., Poole, B., et al. Reconfusion: 3d reconstruction with diffusion priors. *arXiv preprint arXiv:2312.02981*, 2023.

Yang, Y., Sun, F.-Y., Weihs, L., VanderBilt, E., Herrasti, A., Han, W., Wu, J., Haber, N., Krishna, R., Liu, L., et al. Holodeck: Language guided generation of 3d embodied ai environments. *arXiv preprint arXiv:2312.09067*, 2023.

Ye, V., Li, Z., Tucker, R., Kanazawa, A., and Snavely, N. Deformable sprites for unsupervised video decomposition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2657–2666, 2022.

Yu, H.-X., Guibas, L. J., and Wu, J. Unsupervised discovery of object radiance fields. *arXiv preprint arXiv:2107.07905*, 2021.

Yu, J., Xu, Y., Koh, J. Y., Luong, T., Baid, G., Wang, Z., Vasudevan, V., Ku, A., Yang, Y., Ayan, B. K., et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2(3):5, 2022.

Zhang, Q., Wang, C., Siarohin, A., Zhuang, P., Xu, Y., Yang, C., Lin, D., Zhou, B., Tulyakov, S., and Lee, H.-Y. Scenewiz3d: Towards text-guided 3d scene composition. *arXiv preprint arXiv:2312.08885*, 2023.

Figure 8: **Decomposing NeRFs of scenes.** Given a NeRF representing a scene **(a)** and a caption, layout learning is able to parse the scene into the objects it contains without any per-object supervision **(b)**. We accomplish this by requiring renders of one of the N learned layouts to match the same view rendered from the target NeRF **(c)**, using a simple L_2 reconstruction loss with $\lambda = 0.05$.

A. Appendix

A.1. Implementation details

We use Mip-NeRF 360 as the 3D backbone (Barron et al., 2022) and Imagen (Saharia et al., 2022), a 128px pixel-space diffusion model, for most experiments, rendering at 512px. To composite multiple representations, we merge the output albedos and densities at each point, taking the final albedo as a weighted average given by per-NeRF density. We apply this operation to the outputs of the proposal MLPs as well as the final RGB-outputting NeRFs. We use $\lambda_{\text{dist}} = 0.001, \lambda_{\text{acc}} = 0.01, \lambda_{\text{ori}} = 0.01$ as well as $\lambda_{\text{empty}} = 0.05$. The empty loss examines the mean of the per-pixel accumulated density along rays in a rendered view, $\boldsymbol{\alpha}$, for each NeRF. It penalizes these mean $\bar{\boldsymbol{\alpha}}$ values if they are under a certain fraction of the image canvas (we use 10%). For more robustness to noise, we pass $\boldsymbol{\alpha}$ through a scaled sigmoid to binarize it (Fig. 10), yielding the $\bar{\boldsymbol{\alpha}}_{\text{bin}}$ used in Eq. 5. We sample camera azimuth in $[0°, 360°]$ and elevation in $[-90°, 0°]$ except in rare cases where we sample azimuth in a 90-degree range to minimize Janus-problem artifacts or generate indoor scenes with a diorama-like effect.

We use a classifier-free guidance strength of 200 and textureless shading probability of 0.1 for SDS (Poole et al., 2022), disabling view-dependent prompting as it does not aid in the generation of compositional scenes (Table 3b). We otherwise inherit all other details, such as covariance annealing and random background rendering, from SDS. We optimize our model with Shampoo (Gupta et al., 2018) with a batch size of 1 for 15000 steps with an annealed learning rate, starting from 10^{-9}, peaking at 10^{-4} after 3000 steps, and decaying to 10^{-6}.

Optimizing NGPs. To verify the robustness of our approach to different underlying 3D representations, we also experiment with a re-implementation of Instant NGPs (Müller et al., 2022), and find that our method generalizes to that setting. Importantly, we implement an aggressive coarse-to-fine training regime in the form of slowly unlocking grid settings at resolution higher than 64×64 only after 2000 steps. Without this constraint on the initial smoothness of geometry, the representation "optimizes too fast" and is prone to placing all density in one NGP.

A.2. Pseudo-code for layout learning

In Figs. 9 and 10, we provide NumPy-like pseudocode snippets of the core logic necessary to implement layout learning, from transforming camera rays to compositing multiple 3D volumes to regularizing them.

A.3. CLIP evaluation

To evaluate our approach, we use similarity scores output

```
# Initialize variables
quat = normal((N, K, 4), mean=[0,0,0,1.], std=0.1)
trans = normal((N, K, 3), mean=0., std=0.3)
scale = normal((N, K, 1), mean=1., std=0.3)
nerfs = [init_nerf() for i in range(K)]

# Transform rays for NeRF k using layout n
def transform(rays, k, n):
  rot = quaternion_to_matrix(quat)
  rays['orig'] = rot[n,k] @ rays['orig'] - trans[n,k]
  rays['orig'] *= scale[n,k]
  rays['dir'] = scale[n,k] * rot[n,k] @ rays['dir']
  return rays

# Composite K NeRFs into one volume
def composite_nerfs(per_nerf_rays):
  per_nerf_out = [nerf(rays) for nerf, rays
    in zip(nerfs, per_nerf_rays]
  densities = [out['density'] for out in per_nerf_out]
  out = {'density': sum(densities)}
  wts = [d/sum(densities) for d in densities]
  rgbs = [out['rgb'] for out in per_nerf_out]
  out['rgb'] = sum(w*rgb for w,rgb in zip(wts, rgbs))
  return out, per_nerf_out

# Train
optim = shampoo(params=[nerfs, quat, trans, scale])
for step in range(num_steps):
  rays = sample_camera_rays()
  n = random.uniform(N)
  per_nerf_rays = [
    transform(rays, k, n) for k in range(K)
  ]
  vol, per_nerf_vols = composite_nerfs(per_nerf_rays)
  image = render(vol, rays)
  loss = SDS(image, prompt, diffusion_model)
  loss += regularize(per_nerf_vols)
  loss.backward()
  optim.step_and_zero_grad()
```

Figure 9: **Pseudocode for layout learning**, with segments inherited from previous work abstracted into functions.

```
def soft_bin(x, t=0.01, eps=1e-7):
  # x has shape (..., H, W)
  bin = sigmoid((x - 0.5)/t)
  min = bin.min(axis=(-1, -2), keepdims=True)
  max = bin.max(axis=(-1, -2), keepdims=True)
  return (bin - min) / (max - min + eps)
soft_bin_acc = soft_bin(acc).mean((-1,-2))
empty_loss = empty_loss_margin - soft_bin_acc
empty_loss = max(empty_loss, 0.)
```

Figure 10: **Pseudocode for empty NeRF regularization,** where `soft_bin_acc` computes $\bar{\alpha}_{\text{bin}}$ in Equation 5.

```
'a cup of coffee, a croissant, and a closed book',
'a pair of slippers, a robe, and a candle',
'a basket of berries, a carton of whipped cream, and an orange',
'a guitar, a drum set, and an amp',
'a campfire, a bag of marshmallows, and a warm blanket',
'a pencil, an eraser, and a protractor',
'a fork, a knife, and a spoon',
'a baseball, a baseball bat, and a baseball glove',
'a paintbrush, an empty easel, and a palette',
'a teapot, a teacup, and a cucumber sandwich',
'a wallet, keys, and a smartphone',
'a backpack, a water bottle, and a bag of chips',
'a diamond, a ruby, and an emerald',
'a pool table, a dartboard, and a stool',
'a tennis racket, a tennis ball, and a net',
'sunglasses, sunscreen, and a beach towel',
'a ball of yarn, a pillow, and a fluffy cat',
'an old-fashioned typewriter, a cigar, and a glass of whiskey',
'a shovel, a pail, and a sandcastle',
'a microscope, a flask, and a laptop',
'a sunny side up egg, a piece of toast, and some strips of bacon',
'a vase of roses, a slice of chocolate cake, and a bottle of red wine',
'three playing cards, a stack of poker chips, and a flute of champagne',
'a tomato, a stalk of celery, and an onion',
'a coffee machine, a jar of milk, and a pile of coffee beans',
'a bag of flour, a bowl of eggs, and a stick of butter',
'a hot dog, a bottle of soda, and a picnic table',
'a pothos houseplant, an armchair, and a floor lamp',
'an alarm clock, a banana, and a calendar',
'a wrench, a hammer, and a measuring tape',
'a backpack, a bicycle helmet, and a watermelon'
```

Figure 11: **Prompts used for CLIP evaluation.** Each prompt is injected into the template "a DSLR photo of {prompt}, plain solid color background". To generate individual objects, the three objects in each prompt are separated into three new prompts and optimized independently.

by a pretrained contrastive text-image model (Radford et al., 2021), which have been shown to correlate with human judgments on the quality of compositional generation (Park et al., 2021). However, rather than compute a retrieval-based metric such as precision or recall, we report the raw ($100\times$ upscaled, as is common practice) cosine similarities. In addition to being a more granular metric, this avoids the dependency of retrieval on the size and difficulty of the test set (typically only a few hundred text prompts).

We devise a list of 30 prompts (Fig. 11), each of which lists three objects, spanning a wide range of data, from animals to food to sports equipment to musical instruments. As described in Section 4, we then train models with $K = 3$ NeRFs and layout learning and test whether each NeRF contains a different object mentioned in the prompt. We compute CLIP scores for each NeRF with a query prompt "a DSLR photo of [A/B/C]", yielding a 3×3 score matrix. To compute NeRF-prompt CLIP scores, we average text-image similarity across 12 uniformly sampled views, each 30 degrees apart, at $-30°$ elevation. We then select the best NeRF-prompt assignment (using brute force, as there are only $3! = 6$ possible choices), and run this process across 3 different seeds, choosing the one with the highest mean NeRF-prompt score.