DreamDissector: Learning Disentangled Text-to-3D Generation from 2D Diffusion Priors

Zizheng Yan[123], Jiapeng Zhou[123], Fanpeng Meng[123], Yushuang Wu[123], Lingteng Qiu[123], Zisheng Ye[123], Shuguang Cui[213], Guanying Chen[13], and Xiaoguang Han[213⋆]

[1] Shenzhen Future Network of Intelligence Institute [2] SSE, CUHKSZ
[3] Guangdong Provincial Key Laboratory of Future Networks of Intelligence, CUHKSZ
zizhengyan@link.cuhk.edu.cn

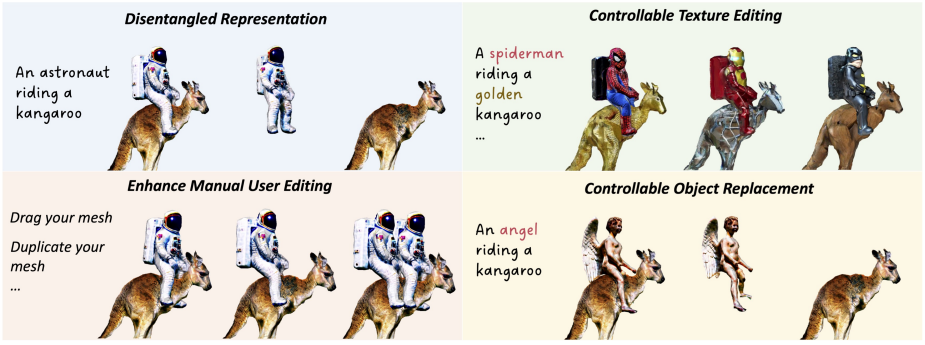

Fig. 1: Results and applications of our DreamDissector. Top left: DreamDissector can generate multiple independent textured meshes with plausible interactions. **Top right**: Facilitating text-guided texturing at the object level. **Bottom left**: Enhancing convenient manual user geometry editing through simple operations. **Bottom right**: Facilitating text-guided controllable object replacement.

Abstract. Text-to-3D generation has recently seen significant progress. To enhance its practicality in real-world applications, it is crucial to generate multiple independent objects with interactions, similar to layer-compositing in 2D image editing. However, existing text-to-3D methods struggle with this task, as they are designed to generate either non-independent objects or independent objects lacking spatially plausible interactions. Addressing this, we propose DreamDissector, a text-to-3D method capable of generating multiple independent objects with interactions. DreamDissector accepts a multi-object text-to-3D NeRF as input and produces independent textured meshes. To achieve this, we introduce the Neural Category Field (NeCF) for disentangling the input NeRF. Additionally, we present the Category Score Distillation Sampling (CSDS), facilitated by a Deep Concept Mining (DCM) module, to tackle the concept gap issue in diffusion models. By leveraging NeCF and CSDS, we can effectively derive sub-NeRFs from the original scene. Further refinement enhances geometry and texture. Our experimental results validate

⋆ Corresponding Author.

the effectiveness of DreamDissector, providing users with novel means to control 3D synthesis at the object level and potentially opening avenues for various creative applications in the future.

Keywords: Text-to-3D · Disentangled Generation · Diffusion Prior

1 Introduction

Creating high-quality 3D content is crucial for enhancing the visual experience in movies, video games, and emerging augmented/virtual reality environments. With the advent of deep learning, numerous techniques for generating 3D content have been proposed [14, 15, 30, 31, 39, 62, 68, 72]. Despite this, the pace of progress in 3D content generation has been relatively slow compared to that of 2D image generation. Recently, text-to-image (T2I) generation has seen remarkable progress [45, 48, 49], propelled by advances in diffusion models [16, 56] and large-scale training data [50,51]. Observing the success of T2I models, researchers have begun adapting the knowledge from T2I models for 3D generation. Poole *et al*. [42] introduced a method known as Score Distillation Sampling (SDS), which optimizes a Neural Radiance Field (NeRF) using the T2I diffusion model for guidance, attracting significant interest in the 3D generation domain.

Among these advancements, there has been research focused on enhancing the practicality of 2D content creation in real-world applications, such as layered image generation [3, 70]. Inspired by this, the concept of "layered" objects capable of interacting in a 3D space is an emerging area of interest for practical applications. Consider the task of building a 3D model where an astronaut is riding a kangaroo, as shown in Figure 1. The ability to individually manipulate each element, such as reorienting the kangaroo or modifying the kangaroo's texture without impacting the astronaut's model, offers significant convenience and flexibility, thereby greatly enhancing the creative workflow for human artists. To facilitate such precise editing, the objects should be independently represented, meaning they should have separate surface meshes at the object level.

However, existing text-to-3D methods struggle to fulfill such a complex task as they are designed to generate either non-independent objects or an assembly of independent objects that lack spatially plausible interactions. CompoNeRF [27] and Comp3D [41] utilize a set of 3D bounding boxes as input conditions for score distillation sampling to generate multiple independent and composable NeRFs. Although the generated local NeRFs can be composed into a global scene, the requirement for 3D input bounding boxes limits the objects to having only simple interactions, *e.g.*, a bed positioned next to a nightstand. Consequently, these methods struggle to generate objects with complex interactions, *e.g.*, an astronaut riding a kangaroo.

To this end, we propose DreamDissector, a text-to-3D method that generates multiple independent objects with interactions. DreamDissector begins with a text-to-3D NeRF containing multiple interacting objects and produces several independent textured meshes, maintaining the interactions and overall appearance

while improving geometries and textures. This process comprises two phases: disentanglement and refinement. Specifically, we introduce a novel Neural Category Field (NeCF) representation that learns a probability distribution for each point in space across all categories. This allows the original density field to be decomposed according to the distribution, enabling the input NeRF to be disentangled into multiple independent sub-NeRFs for each category. For training the disentanglement phase, we introduce a Category Score Distillation Sampling (CSDS) loss, which consists of a set of SDS losses for the subjects depicted in the input NeRF. However, we observed that the CSDS with vanilla diffusion model is inadequate for addressing concept gaps — discrepancies where the object generated from the complete text prompt and the category-specific text prompt occupy different areas in the T2I diffusion model's latent space. This leads to mismatched disentanglement, as shown in Figure 3 left. To overcome these concept gaps, we leverage a technique called Deep Concept Mining (DCM), which personalizes the diffusion model by learning the profound concept portrayed by the input NeRF. This personalized diffusion model significantly enhances the disentanglement phase. Following disentanglement, we transform the sub-NeRFs into DMTets [53] for further refinement, which not only corrects certain artifacts, resulting in improved geometry and texture, but also enables the export of multiple independent surface meshes. Finally, we demonstrate the practicality of DreamDissector through a set of controllable editing applications.

To summarize, our contributions are as follows:

- To the best of our knowledge, we are the first to address the problem of disentangling text-to-3D NeRFs.
- To address the problem, we introduce a novel framework named DreamDissector, including a novel Neural Category Field (NeCF) representation that can disentangle an input NeRF into indepedent sub-NeRFs, a Deep Concept Mining (DCM) technique to facilitate the alignment between sub-NeRFs and concepts through personalizing a diffusion model, and a Category Score Distillation Sampling (CSDS) loss that harnesses DCM to enhance the NeCF learning.
- Experimental results demonstrate the effectiveness of DreamDissector, and additional controllable editing applications illustrate its practicality in real scenarios.

2 Related Works

3D Generative models. Notable advancements have been witnessed in 3D generative models in recent years. Among those advancements, various representations have been explored, including point clouds [1, 37, 39, 65, 68, 72], 3D voxels [15, 30, 55, 61], meshes [14, 71], and implicit representations [8, 33]. However, a substantial performance gap exists between these 3D generative models and their 2D counterparts, primarily attributable to the lack of 3D training data. Despite several endeavors to develop 3D-aware generative models from 2D image

collections, generating high-fidelity assets with 3D consistency and scalability is still challenging.

Text-to-3D Generation. On the other hand, T2I generation thrives on the huge amount of 2D image data. Therefore, researchers intuitively tried to make use of powerful T2I models for later 3D generations. One of the representative methods is to use the CLIP model [43], which can match input texts and rendered images better than any mechanisms before. Some early text-to-3D work directly employed CLIP to supervise 3D features during generation, such as CLIP-Mesh [23] and Text2Mesh [35] focusing on mesh, and so on [17, 22, 25]. Similar to CLIP, the Diffusion model [45, 48, 59] caught researchers' attention for its promising performance in T2I. Poole*et al.* [42] proposed a novel method of Score Distillation Sampling (SDS) to optimize 3D models from their multiple views generated by a 2D diffusion model. Concurrently, Wang *et al.* [57] proposed a score jacobian chaining method to accomplish similar tasks. However, these methods still suffer from multi-view inconsistency problems. To improve the 3D synthesis performance, follow-up research can be roughly classified into several categories — pursuing more effective expressions of SDS [18,20,34,44,60], applying two-stage coarse-to-fine optimization [7, 26], introducing more 3D priors or constructing 3D datasets [28, 29, 32, 52, 54, 67], and utilizing the power of more models such as CLIP [58, 64] and LLMs [19, 66].

Multi-object 3D Generation. For multi-object text-to-3D generation, most methods model the objects in a scene collectively, as seen in Text2NeRF [69], Text2Room [21], and SceneScape [12]. Compared to earlier methods, these can create more photorealistic scenes with intricate geometry and textures. Although this reduces complexity in the design of modeling algorithms, treating multiple objects as a single undivided scene limits the ability to edit each object individually. Some recent work [27, 41] has attempted to simultaneously generate multiple composable 3D objects. However, these methods require additional 3D bounding boxes as input to specify the objects' positions within a scene, which increases the user's workload. There are also some concurrent works addressing the similar problem [11, 13, 73].

3 Preliminary

Neural Radiance Field (NeRF) [36] is a representation proposed to synthesize the novel views of a 3D scene. NeRF employs a multi-layer perception (MLP) to directly model the density σ and the color $\mathbf{c}$ at each point in a 3D space. Given a camera ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$ defined by origin $\mathbf{o}$ and the view direction $\mathbf{d}$, the input points are sampled along the ray at different depth t, and the colors $\mathbf{c}_i$ and densities σ_i are obtained by the MLP. Then, volume rendering is used to compute the final color of the pixels:

$$C(\mathbf{r}) = \sum_{i=1}^{N} \alpha_i (1 - \exp(-\sigma_i \delta_i)) \mathbf{c}_i \tag{1}$$

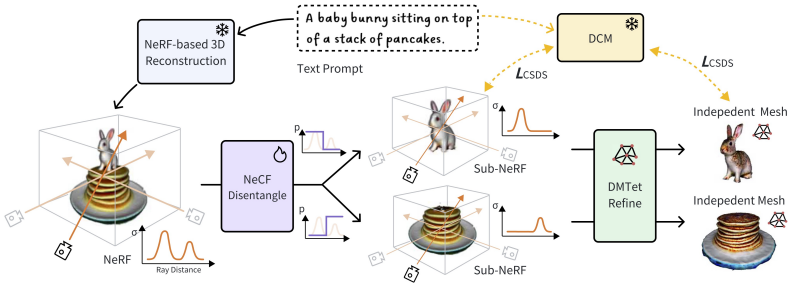

Fig. 2: Method overview. We generate multiple independent interactive 3D objects in a coarse-to-fine manner. Initially, we render a view of the input text-to-3D NeRF for Deep Concept Mining (DCM), obtaining both the T2I diffusion model and the corresponding text embedding. We then use the mined embedding and the T2I diffusion model to train the neural category field (NeCF) using category score distillation sampling (CSDS). After disentangling the input NeRF, we convert the sub-NeRFs into DMTets and fine-tune these for further refinement. Finally, we export independent surface meshes with improved geometries and textures.

where $\alpha_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j)$ denotes the transmittance probability at i, and δ_i denotes the distance between consecutive samples.

Score Distillation Sampling (SDS) is the key to the success of DreamFusion [42], which aims to distill the knowledge from a pre-trained text-to-image diffusion-based generative model [48, 49] to achieve text-to-3D generation. The diffusion model consists of a denoising function $\epsilon_\phi(x_t, y, t)$ [16] that predicts the sampled noise ϵ given the noisy image x_t, the time step t, and text prompt embedding y. By feeding a set of rendered images of the NeRF parameterized by θ to the frozen denoising function $\epsilon_\phi(x_t, y, t)$, the NeRF is optimized towards the direction where the rendered images closer to the high probability density regions conditioned on the text embedding under the 2D diffusion prior. Specifically, the gradients of SDS are estimated as follows,

$$\nabla_\theta L_{SDS}(\phi, \theta) = \mathbb{E}_{t,\epsilon}\left[w(t)\left(\epsilon_\phi(x_t; y, t) - \epsilon\right)\frac{\partial x}{\partial \theta}\right], \tag{2}$$

where $w(t)$ is the weighting function controlling the strength of the SDS guidance.

4 Methodology

4.1 Overview

DreamDissector begins with a text-to-3D Neural Radiance Field (NeRF). Its objective is to disentangle the generated 3D NeRF into separate 3D assets according to the object categories that the NeRF contains. To achieve this, we introduce a 3D representation called the Neural Category Field (NeCF).

This is designed to disentangle the target NeRF into multiple sub-NeRFs while maintaining the original appearance of each object. The NeCF is supervised by our newly introduced Category Score Distillation Sampling (CSDS), an approach that involves a series of Score Distillation Samplings (SDS) conditioned on category-specific text prompts for the sub-NeRFs. Subsequently, the sub-NeRFs are transformed into DMTets [53] for final geometry and texture refinement. Since DMTets can be readily converted into surface meshes, DreamDissector ultimately produces independent surface meshes for each object with the preservation of the actions and interactions, thereby facilitating editing by human artists. An outline of our DreamDissector framework is illustrated in Figure 2.

4.2 Neural Category Field

To render each categorical object in the target NeRF, a straightforward solution is to introduce a sub-NeRF for each object, *e.g.*, a density field and a color field, respectively. Subsequently, each object can be rendered using its density and color field. The entire NeRF can then be rendered by composing these density and color fields according to the principles of volume rendering [10, 40]:

$$\sigma = \sum_{k=1}^{K} \sigma_k, \qquad \mathbf{c} = \frac{1}{\sigma} \sum_{k=1}^{K} \sigma_k \mathbf{c}_k, \tag{3}$$

where K denotes the number of categories. However, this approach requires training additional networks for density and color fields and needs a constraint loss to maintain the appearance consistency of the entire NeRF.

To this end, we propose an alternate formulation for rendering each category object by ***de-composing*** the density field with a probability distribution, namely the category field. Specifically, the above density composition can be reformulated as follows,

$$\sigma = \sum_{k=1}^{K} \frac{\sigma_k}{\sigma} \sigma, \quad \frac{\sigma_k}{\sigma} \in [0, 1] \tag{4}$$

Note that a small number can be added to the density to avoid division by zero. Consequently, the $\frac{\sigma_k}{\sigma}$ can be regarded as a probability simplex [4] as it sums to one, and its elements are non-negative. Inspired by this, we leverage an MLP with the softmax function to model the probability simplex $\frac{\sigma_k}{\sigma}$ directly. Let $\mathbf{p}_i^k$ represents the probability of the i-th point in the 3D space belongs to the k-th category:

$$\mathbf{p}_i^k = \frac{\exp(f_k/T)}{\sum_k^K \exp(f_k/T)}, \quad \mathbf{p}_i^k \in [0, 1] \tag{5}$$

where T denotes the temperature, which controls the sharpness of the probability distribution, $f \in \mathbb{R}^K$ is the output of the category field network. With the category field, the color of the k-th category object can be rendered as follows,

$$C(\mathbf{r})^k = \sum_{i=1}^{N} \alpha_i^k (1 - \exp(-\mathbf{p}_i^k \sigma_i \delta_i)) \mathbf{c}_i, \quad \alpha_i^k = \exp(-\sum_{j=1}^{i-1} \mathbf{p}_i^k \sigma_j \delta_j). \tag{6}$$

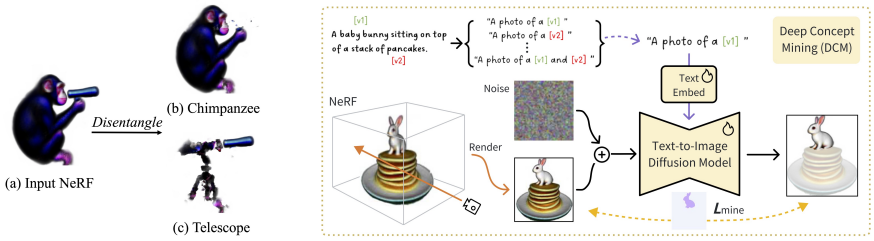

Fig. 3: Left: Concept discrepancy in diffusion models. The text prompt is *"A chimpanzee looking through a telescope"*. **Right: Overview of Deep Concept Mining (DCM).** We finetune the text embedding and the T2I diffusion model with the masked diffusion loss (Eq. 8).

It can be observed that the density σ for each point is scaled by the category field $\mathbf{p}^k$. In other words, we can interpret $\mathbf{p}^k\sigma$ as the density of the sub-NeRF for the k-th category object, *e.g.*, $\sum_k \mathbf{p}_i^k = 1$ and $\sum_k(\mathbf{p}_i^k\sigma) = \sigma$. Furthermore, the color field can be reused and frozen during training, simplifying the training process.

Notably, the design of the NeCF has the following merits: (1) We only need to train an additional category field network, which is more efficient than training additional density and color fields networks. (2) As the original density and color field networks are frozen during training, the re-composition of the sub-NeRFs exactly equals the original NeRF, preserving its original appearance.

4.3 Category Score Distillation Sampling

A naive approach. To train the NeCF, one naive approach is to employ multiple SDS losses to supervise the category field for each category. Specifically, for the object of the k-th category, the gradients of its SDS loss can be formulated as,

$$\nabla_\theta L_{SDS}(\phi, \theta)_k = \mathbb{E}_{t,\epsilon}\left[w(t)\left(\epsilon_\phi(x_t; y_k, t) - \epsilon\right)\frac{\partial x}{\partial \theta}\right], \tag{7}$$

where y_k denotes the text embedding for the k-th category. For example, given a NeRF generated by the prompt: "*a* $[v_1]$ *sitting on a* $[v_2]$*.*", the text prompts for the category objects would be "*a* $[v_1]$" and "*a* $[v_2]$". This can be easily accomplished by human users or by modern LLMs. It is important to note that we do not require SDS for the entire text prompts for training NeCF, and all networks, except for the category field networks, are frozen.

Concept Discrepancy in Diffusion Model. Although the naive approach can handle some simple cases, it fails to disentangle scenes with concept gaps from the text descriptions. The concept gap refers to the discrepancy where the object generated by the complete text prompt and by the category text prompt occupy different areas in the 2D diffusion model's latent space. For instance, the text prompt "*a chimpanzee looking through a telescope.*" would generate a scene

depicting a chimpanzee using a handheld telescope, as shown in Figure 3 left (a). In contrast, the category text prompt "*a telescope*" is more likely to generate a tripod-mounted telescope, since the tripod-mounted telescope is situated in the dominant feature space of the prompt "*a telescope*" while the handheld telescope occupies a marginal feature space. As a result, the learned NeCF will produce a tripod-mounted telescope with the tripod being hidden inside the chimpanzee's body, as illustrated by Figure 3 left (c).

Deep Concept Mining. To address this issue, we propose mining the concepts in the text prompt and aligning them with ones depicted in NeRF for disentanglement, as illustrated in Figure 3 right. To this end, a T2I diffusion model is personalized to denoise a given view rendered by the NeRF into an image depicting one (or several) independent object(s), under the condition of one (or several) specific concept(s). Specifically, we first create a set of prompts where each contains one or several concepts. For each concept or concept combination, we generate the corresponding segmentation mask for a rendered view of the NeRF, through a text-based open-vocabulary segmentation model, *e.g.* Grounded-SAM [46]. Then we utilize the prompt-mask pairs to optimize the text embedding and the diffusion backbone with a concept mining loss with mask attention [2]:

$$L_{mine}(\phi, y_k) = \mathbb{E}_{t,\epsilon}\left[||\epsilon_\phi(x_t; y_k, t) \odot M_k - \epsilon \odot M_k||_2^2\right], \tag{8}$$

where M_k denotes the mask of the k-th category. The DCM module is frozen after optimization to provide better alignment between the independent textual concepts and the sub-NeRFs for better NeCF training with the CSDS loss. The frozen DCM is also used to train the DMTet refinement module as illustrated in Figure 2.

Final Refinement. After training NeCF, we convert the sub-NeRFs into DMTets using the isosurface extraction technique [26], and fine-tune these DMTets with text embeddings and the model from DCM. The rationale is that further refinement can fix the artifacts produced by disentanglement, and DMTets can be easily converted into surface meshes. However, DCM tends to overfit the mined concept in the original NeRF, resulting in over-saturated and unrealistic colors [2, 42]. To address this, we employ the original stable diffusion to fine-tune the colors of the DMTets through additional steps, enhancing their realism. Finally, the DMTets are transformed into textured meshes.

4.4 Overall Pipeline

Therefore, the overall pipeline comprises the following steps:

- Generate a mask for each category in the prompt from a rendered view and use them to optimize the DCM module.
- Freeze the DCM, and train the NeCF network using the CSDS loss for decompose a NeRF into sub-NeRFs of independent objects.
- Convert the sub-NeRFs into DMTets and fine-tune them with the optimized DCM module, then fine-tune the colors of the DMTets using the original stable diffusion to produce the final output.

(a) NeRF (b) View1 (c) View2

Fig. 4: Qualitative Results. The text prompts used to generate the NeRF are available in the supplementary file.

5 Experiments

5.1 Implementation Details

Category field. Our method is implemented based on the publicly available repository threestudio[4]. We use instant-NGP [38] as the NeRF representation, which includes a multi-resolution hash grid and two MLPs for the density and color networks. Similarly, we employ a random initialized hash grid and an MLP for the category field, using the same configuration as the density and color networks. The output dimension of the category field MLP is set to the number of categories in the original NeRF, and the temperature T is set to 0.05 to make the category field approximate a one-hot encoding. The training of the category field follows the Dreamfusion approach, except that the color and density networks

[4] https://github.com/threestudio-project/threestudio

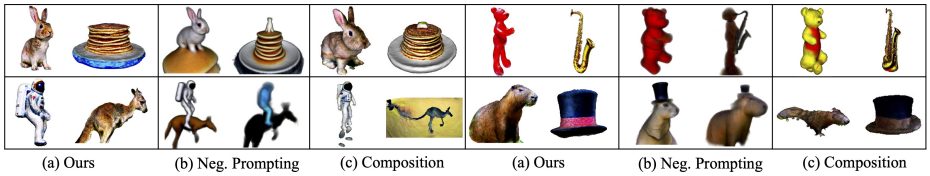

Fig. 5: Comparison with two baselines. We show the independent objects for ease of comparison. The composed objects and more comparisons are available in the supplementary file.

are frozen. The training lasts for 1,000 steps with a batch size of 1, taking approximately 3 minutes.

Deep concept mining. For DCM, we have adopted Stable Diffusion 2.1 as the backbone. Inspired by [2], we adopt a two-stage training strategy for deep concept mining to stabilize the training: fine-tuning the text embedding in the first stage and fine-tuning both the text embedding and the backbone in the second stage. We use a learning rate of 5×10^{-4} for the first stage and a learning rate of 2×10^{-6} for the second stage. The first stage is trained for 400 steps, and the second stage for 100 steps. Notably, the training of DCM takes approximately 6 minutes on an NVIDIA A100, which is time-efficient compared with most text-to-3D methods that require several hours of training on a single GPU, including Dreamfusion [42], Magic3D [26], ProlificDreamer [60], and MVDream [54].

Final refinement. We adopt the same technique used in Magic3D [26] to convert the sub-NeRFs into DMTets. Similarly, we train the DMTets with a batch size of 8 for 5,000 steps. Subsequently, we fix the DMTet geometry and fine-tune the color for 1,000 steps using the original Stable Diffusion. Note that the color fine-tuning is optional as the color is already decent for most cases. During color fine-tuning, we use "unrealistic, low quality, shadow" as the negative prompt to improve quality. Finally, we export the DMTets as textured meshes.

5.2 Results

Main results. The qualitative results are shown in Figure 4. For each case, two views of each object are sampled, and the corresponding text prompts are available in the supplementary file. It can be observed that DreamDissector can effectively disentangle input scenes featuring various complex interactions, such as riding. Notably, DreamDissector can handle cases with large and complex contact surfaces, as demonstrated in the "*an octopus playing the piano*" case, where the octopus's tentacles are disentangled from the piano. Additionally, the final meshes exhibit more realistic and higher-quality textures compared to those in the input NeRF. This improvement is attributed to the final refinement, further demonstrating DreamDissector's practicality.

Comparison. We compare DreamDissector with two baselines: negative prompting and a composition baseline. Negative prompting involves taking the entire text prompt as the positive prompt and identifying the exclusive object as the

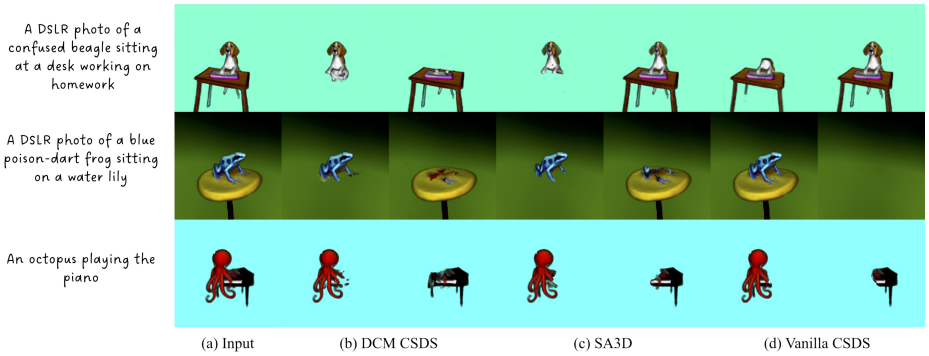

Fig. 6: Using different strategies to disentangle the NeRF. Our DCM CSDS successfully disentangles the sub-NeRFs, whereas SA3D [5] and the vanilla CSDS fail in some cases. Note that the outliers can be easily removed by thresholding in the isosurface extraction.

Table 1: CLIP Score comparisons.

Method	CLIP-B-16	CLIP-B-32	CLIP-L-14
Negative Prompting	0.299	0.296	0.247
Composition	0.281	0.278	0.234
Ours	**0.316**	**0.311**	**0.270**

negative prompt. For example, in the prompt "*a* $[v_1]$ *sitting on* $[v_2]$", the positive prompt for both objects is the entire prompt, while the negative prompt for object $[v_1]$ is "$[v_2]$", and vice versa. Since the most relevant works, CompoNeRF [27] and Comp3D [41], are not open-source, we implemented a composition baseline with a similar idea: training the objects separately and then composing them with further fine-tuning. We compare our method with these baselines both qualitatively and quantitatively. As Figure 5 demonstrates, DreamDissector significantly outperforms the baseline methods. Additionally, we evaluate DreamDissector and the baseline methods using the CLIP Score metric, which measures cosine similarity between the embeddings of the text and image. We conduct this evaluation on both independent and composed objects and compute the average score. As Table 1 shows, our method significantly outperforms the baselines.

5.3 Analysis

DCM for disentanglement. To evaluate the effectiveness of deep concept mining (DCM) in disentangling NeRF, we conducted a comparative analysis with two baselines: (1) the vanilla Category Score Distillation Sampling (CSDS) without DCM, and (2) Segment Anything [24] in 3D with NeRFs (SA3D) [5]. Unlike the fully unsupervised vanilla CSDS, both our DCM approach and SA3D require an input mask for a single view. As Figure 6 shows, the vanilla CSDS struggles

Fig. 7: Illustration of DCM refinement. (a) Disentangled NeRF with artifacts, (b) sampled image of original stable diffusion (SD), (c) sampled image of DCM SD, (d) undesired results from fine-tuning on original SD, and (e) artifacts fixed by DCM SD.

Fig. 8: Ablation study of DCM. We show the sampled images of DCM fine-tuned model. (a) The input rendered image, (b) DCM, (c) without the masked attention loss, (d) without the first stage training, and (e) without the second stage training.

to disentangle NeRF for scenarios with significant concept discrepancies, such as "*a blue poison-dart frog sitting on a water lily*," where the original scene primarily depicts water lily leaves. While SA3D manages to disentangle scenarios involving concept discrepancies, such as the frog, it falls short in more complex cases with extensive occlusions, exemplified by the beagle and octopus cases. In contrast, DCM demonstrates superior performance, successfully disentangling scenarios involving concept discrepancies and significant occlusions.

DCM for refinement. DCM is utilized not only for NeRF disentanglement but also for refining DMTets. We conducted an analysis on the effectiveness of DCM in this refinement. The results are presented in Figure 7. From (a), it can be observed that artifacts remain after disentanglement. Due to the original NeRF's invisible contact surface, a "black hole" appears post-disentanglement. However, using original stable diffusion for DMTet refinement doesn't resolve this, as illustrated in (d). This is because the prompt "*a stack of pancakes*" typically generates images with fruits on the pancakes, as these are prevalent in the high-density regions of stable diffusion, evidenced in (b). Consequently, the fine-tuned DMTet produces fruit from the black hole artifact region. In contrast, DCM stable diffusion closely matches the input pancake, as shown in the first row, effectively fixing the artifacts during DMTet refinement, as seen in (e). This further demonstrates DCM's superiority.

Ablation study on DCM. We perform an ablation study on each component of DCM, including two-stage training and the masked attention loss. Specifically, we sample the images of the mined concept of "*baby bunny*" from the text prompt "*a baby bunny sitting on a stack of pancakes*", using the fine-tuned models. Ide-

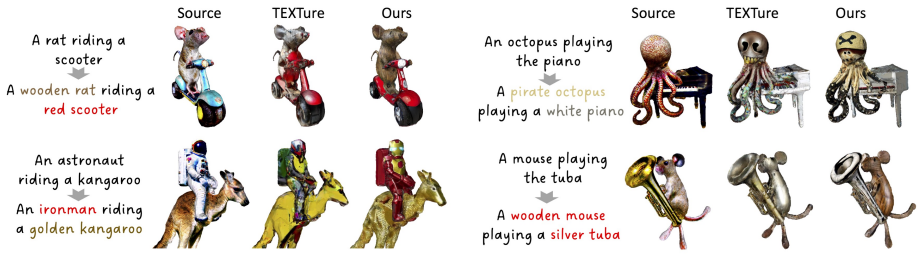

Fig. 9: Illustration of the application of **text-guided texture editing**.

ally, the sampled image should not contain any concepts similar to pancakes. As Figure 8 shows, DCM successfully extracts the concept of "*baby bunny*", whereas other training strategies fail to separate this concept from others, such as items resembling the pancakes upon which it is sitting. This demonstrates DCM's ability to mine independent concepts.

5.4 Applications

Controllable texture editing. Although text-guided texturing has achieved notable progress [6,47], generating textures for complex scenes with multiple objects remains challenging. We evaluated TEXTure [47] in three different cases, as depicted in Figure 9. For the baseline, We treated the multi-object mesh as a single entity and applied TEXTure. For our approach, we applied TEXTure to each object's mesh individually and then combined them. We observed that the textures generated by the baseline approach poorly matched the input prompts and were of low quality. Notably, textures for independent objects were influenced by other objects in the scene, *e.g.*, part of the rat exhibited a red color. In contrast, DreamDissector significantly enhances TEXTure's performance, producing visually appealing and accurate textures.

Controllable object replacement. In addition to controllable texture editing, DreamDissector also has the capability to replace individual objects without affecting other objects in the scene. To achieve this, the target DMTet is fine-tuned while the remaining DMTets are kept fixed. However, deforming a DMTet into a completely different topology object is challenging using SDS-based supervision [26]. Inspired by [7], we initially feed the normals of the DMTet to stable diffusion for several steps, effectively deforming the DMTet. We also empirically observe that fine-tuning the target DMTet only will induce severe mesh interpenetration. To address this, we introduce an interpenetration loss,

$$\mathcal{L}_{interpenetration} = \sum_i \max(\epsilon - (\mathbf{v}_i - \mathbf{v}'_i) \cdot \mathbf{n}'_i, 0), \tag{9}$$

where $\mathbf{v}_i$ represents the i-th vertex of the target DMTet, $\mathbf{v}'_i$ and $\mathbf{n}'_i$ are the vertex and vertex normal of the nearest neighbor of $\mathbf{v}_i$ in other DMTets, respectively, and ϵ is a small tolerance hyper-parameter for interpenetration. The results,

Fig. 10: Illustration of the application of **text-guided object replacement**.

Fig. 11: Illustration of the application of **geometry editing**.

shown in Figure 10, demonstrate that DreamDissector can achieve controllable concept replacement.

Geometric editing by users. To further verify how DreamDissector can facilitate user workflows, we allowed a user to edit objects individually. As demonstrated in Figure 11, an object can be easily modified by simple operations such as scaling, translation, and dragging, thereby highlighting DreamDissector's effectiveness in enhancing human editing capabilities in practical applications.

6 Conclusion

We propose DreamDissector, a novel framework designed to generate multiple, independently interacting objects guided by text. DreamDissector takes a multi-object text-to-3D Neural Radiance Field (NeRF) as input and produces several textured meshes. We introduce the Neural Category Field (NeCF), a representation capable of disentangling the input NeRF into multiple sub-NeRFs. To train NeCF, we present the Category Score Distillation Sampling (CSDS) loss. Moreover, we have observed a concept discrepancy issue in 2D diffusion models, which can degrade disentanglement performance. To address this, we introduce Deep Concept Mining (DCM) to fine-tune the text embeddings and the 2D diffusion model, effectively deriving sub-NeRFs. Additionally, we propose a two-stage refinement process to further refine the geometry and texture, thereby enhancing realism. Experimental results and further applications showcase the effectiveness and practicality of DreamDissector in real-world scenarios.

Acknowledgement The work was supported in part by NSFC with Grant No. 62293482, the Basic Research Project No HZQB-KCZYZ-2021067 of Hetao Shenzhen-HK S&T Cooperation Zone, Guangdong Provincial Outstanding Youth Fund (No. 2023B1515020055), the National Key R&D Program of China with grant No. 2018YFB1800800, by Shenzhen Outstanding Talents Training Fund 202002, by Guangdong Research Projects No. 2017ZT07X152 and No. 2019CX01X104, by Key Area R&D Program of Guangdong Province (Grant No. 2018B030338001), by the Guangdong Provincial Key Laboratory of Future Networks of Intelligence (Grant No. 2022B1212010001), and by Shenzhen Key Laboratory of Big Data and Artificial Intelligence (Grant No. ZDSYS201707251409055). It is also partly supported by NSFC-61931024, NSFC-62172348, and Shenzhen Science and Technology Program No. JCYJ20220530143604010.

References

1. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning representations and generative models for 3d point clouds. In: ICML (2018) 3
2. Avrahami, O., Aberman, K., Fried, O., Cohen-Or, D., Lischinski, D.: Break-a-scene: Extracting multiple concepts from a single image. arXiv preprint arXiv:2305.16311 (2023) 8, 10
3. Bar-Tal, O., Ofri-Amar, D., Fridman, R., Kasten, Y., Dekel, T.: Text2live: Text-driven layered image and video editing. In: ECCV (2022) 2
4. Boyd, S.P., Vandenberghe, L.: Convex optimization. Cambridge university press (2004) 6
5. Cen, J., Zhou, Z., Fang, J., Shen, W., Xie, L., Zhang, X., Tian, Q.: Segment anything in 3d with nerfs. arXiv preprint arXiv:2304.12308 (2023) 11
6. Chen, D.Z., Siddiqui, Y., Lee, H.Y., Tulyakov, S., Nießner, M.: Text2tex: Text-driven texture synthesis via diffusion models. arXiv preprint arXiv:2303.11396 (2023) 13
7. Chen, R., Chen, Y., Jiao, N., Jia, K.: Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation (2023) 4, 13
8. Chen, Z., Zhang, H.: Learning implicit fields for generative shape modeling. In: CVPR (2019) 3
9. Cohen-Bar, D., Richardson, E., Metzer, G., Giryes, R., Cohen-Or, D.: Set-the-scene: Global-local training for generating controllable nerf scenes. In: ICCVW (2023) 21
10. Drebin, R.A., Carpenter, L., Hanrahan, P.: Volume rendering. Siggraph (1988) 6
11. Epstein, D., Poole, B., Mildenhall, B., Efros, A.A., Holynski, A.: Disentangled 3d scene generation with layout learning. arXiv preprint arXiv:2402.16936 (2024) 4
12. Fridman, R., Abecasis, A., Kasten, Y., Dekel, T.: Scenescape: Text-driven consistent scene generation (2023) 4
13. Gao, G., Liu, W., Chen, A., Geiger, A., Schölkopf, B.: Graphdreamer: Compositional 3d scene synthesis from scene graphs. In: CVPR (2024) 4
14. Gao, J., Shen, T., Wang, Z., Chen, W., Yin, K., Li, D., Litany, O., Gojcic, Z., Fidler, S.: Get3d: A generative model of high quality 3d textured shapes learned from images. NeurIPS (2022) 2, 3
15. Henzler, P., Mitra, N.J., Ritschel, T.: Escaping plato's cave: 3d shape from adversarial rendering. In: ICCV (2019) 2, 3

16. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. NeurIPS (2020) 2, 5
17. Hong, F., Zhang, M., Pan, L., Cai, Z., Yang, L., Liu, Z.: Avatarclip: Zero-shot text-driven generation and animation of 3d avatars. Siggraph (2022) 4
18. Hong, S., Ahn, D., Kim, S.: Debiasing scores and prompts of 2d diffusion for view-consistent text-to-3d generation (2023) 4
19. Hong, Y., Zhen, H., Chen, P., Zheng, S., Du, Y., Chen, Z., Gan, C.: 3d-llm: Injecting the 3d world into large language models (2023) 4
20. Huang, Y., Wang, J., Shi, Y., Qi, X., Zha, Z.J., Zhang, L.: Dreamtime: An improved optimization strategy for text-to-3d content creation (2023) 4
21. Höllein, L., Cao, A., Owens, A., Johnson, J., Nießner, M.: Text2room: Extracting textured 3d meshes from 2d text-to-image models (2023) 4
22. Jain, A., Mildenhall, B., Barron, J.T., Abbeel, P., Poole, B.: Zero-shot text-guided object generation with dream fields. CVPR (2022) 4
23. Khalid, N.M., Xie, T., Belilovsky, E., Popa, T.: CLIP-mesh: Generating textured meshes from text using pretrained image-text models. In: Siggraph Asia (2022) 4
24. 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. ICCV (2023) 11
25. Lee, H.H., Chang, A.X.: Understanding pure clip guidance for voxel grid nerf models (2022) 4
26. Lin, C.H., Gao, J., Tang, L., Takikawa, T., Zeng, X., Huang, X., Kreis, K., Fidler, S., Liu, M.Y., Lin, T.Y.: Magic3d: High-resolution text-to-3d content creation. CVPR (2023) 4, 8, 10, 13
27. Lin, Y., Bai, H., Li, S., Lu, H., Lin, X., Xiong, H., Wang, L.: Componerf: Text-guided multi-object compositional nerf with editable 3d scene layout. arXiv preprint arXiv:2303.13843 (2023) 2, 4, 11
28. Liu, R., Wu, R., Hoorick, B.V., Tokmakov, P., Zakharov, S., Vondrick, C.: Zero-1-to-3: Zero-shot one image to 3d object. ICCV (2023) 4
29. Liu, Y., Lin, C., Zeng, Z., Long, X., Liu, L., Komura, T., Wang, W.: Syncdreamer: Generating multiview-consistent images from a single-view image (2023) 4
30. Lunz, S., Li, Y., Fitzgibbon, A., Kushman, N.: Inverse graphics gan: Learning to generate 3d shapes from unstructured 2d data. arXiv preprint arXiv:2002.12674 (2020) 2, 3
31. Luo, S., Hu, W.: Diffusion probabilistic models for 3d point cloud generation. In: CVPR (2021) 2
32. Melas-Kyriazi, L., Rupprecht, C., Laina, I., Vedaldi, A.: Realfusion: 360deg reconstruction of any object from a single image. CVPR (2023) 4
33. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: CVPR (2019) 3
34. Metzer, G., Richardson, E., Patashnik, O., Giryes, R., Cohen-Or, D.: Latent-nerf for shape-guided generation of 3d shapes and textures. CVPR (2023) 4
35. Michel, O., Bar-On, R., Liu, R., Benaim, S., Hanocka, R.: Text2mesh: Text-driven neural stylization for meshes. CVPR (2022) 4
36. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM (2021) 4
37. Mo, K., Guerrero, P., Yi, L., Su, H., Wonka, P., Mitra, N., Guibas, L.J.: Structurenet: Hierarchical graph networks for 3d shape generation. arXiv preprint arXiv:1908.00575 (2019) 3
38. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. TOG (2022) 9

39. Nichol, A., Jun, H., Dhariwal, P., Mishkin, P., Chen, M.: Point-e: A system for generating 3d point clouds from complex prompts. arXiv preprint arXiv:2212.08751 (2022) 2, 3
40. Niemeyer, M., Geiger, A.: Giraffe: Representing scenes as compositional generative neural feature fields. In: CVPR (2021) 6
41. Po, R., Wetzstein, G.: Compositional 3d scene generation using locally conditioned diffusion. arXiv preprint arXiv:2303.12218 (2023) 2, 4, 11
42. Poole, B., Jain, A., Barron, J.T., Mildenhall, B.: Dreamfusion: Text-to-3d using 2d diffusion. arXiv preprint arXiv:2209.14988 (2022) 2, 4, 5, 8, 10, 19, 22, 23, 24
43. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. ICML (2021) 4
44. Raj, A., Kaza, S., Poole, B., Niemeyer, M., Ruiz, N., Mildenhall, B., Zada, S., Aberman, K., Rubinstein, M., Barron, J., Li, Y., Jampani, V.: Dreambooth3d: Subject-driven text-to-3d generation (2023) 4
45. Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical text-conditional image generation with clip latents. 3DVAR (2022) 2, 4
46. Ren, T., Liu, S., Zeng, A., Lin, J., Li, K., Cao, H., Chen, J., Huang, X., Chen, Y., Yan, F., et al.: Grounded sam: Assembling open-world models for diverse visual tasks. arXiv preprint arXiv:2401.14159 (2024) 8
47. Richardson, E., Metzer, G., Alaluf, Y., Giryes, R., Cohen-Or, D.: Texture: Text-guided texturing of 3d shapes. arXiv preprint arXiv:2302.01721 (2023) 13, 19
48. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR (2022) 2, 4, 5
49. 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. NeurIPS (2022) 2, 5
50. Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., et al.: Laion-5b: An open large-scale dataset for training next generation image-text models. NeurIPS (2022) 2
51. Schuhmann, C., Vencu, R., Beaumont, R., Kaczmarczyk, R., Mullis, C., Katta, A., Coombes, T., Jitsev, J., Komatsuzaki, A.: Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. arXiv preprint arXiv:2111.02114 (2021) 2
52. Seo, J., Jang, W., Kwak, M.S., Ko, J., Kim, H., Kim, J., Kim, J.H., Lee, J., Kim, S.: Let 2d diffusion model know 3d-consistency for robust text-to-3d generation (2023) 4
53. Shen, T., Gao, J., Yin, K., Liu, M.Y., Fidler, S.: Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. NeurIPS (2021) 3, 6
54. Shi, Y., Wang, P., Ye, J., Long, M., Li, K., Yang, X.: Mvdream: Multi-view diffusion for 3d generation (2023) 4, 10, 19, 20
55. Smith, E.J., Meger, D.: Improved adversarial systems for 3d object generation and reconstruction. In: CoRL (2017) 3
56. Song, Y., Ermon, S.: Generative modeling by estimating gradients of the data distribution. NeurIPS (2019) 2
57. Wang, H., Du, X., Li, J., Yeh, R.A., Shakhnarovich, G.: Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. CVPR (2023) 4
58. Wang, T., Zhang, B., Zhang, T., Gu, S., Bao, J., Baltrusaitis, T., Shen, J., Chen, D., Wen, F., Chen, Q., Guo, B.: Rodin: A generative model for sculpting 3d digital avatars using diffusion. CVPR (2023) 4

59. Wang, T., Zhang, T., Zhang, B., Ouyang, H., Chen, D., Chen, Q., Wen, F.: Pre-training is all you need for image-to-image translation (2022) 4
60. Wang, Z., Lu, C., Wang, Y., Bao, F., Li, C., Su, H., Zhu, J.: Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation (2023) 4, 10
61. Wu, J., Zhang, C., Xue, T., Freeman, B., Tenenbaum, J.: Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. NeurIPS (2016) 3
62. Wu, Y., Yan, Z., Chen, C., Wei, L., Li, X., Li, G., Li, Y., Cui, S., Han, X.: Scoda: Domain adaptive shape completion for real scans. In: CVPR (2023) 2
63. Xu, D., Chen, W., Peng, W., Zhang, C., Xu, T., Zhao, X., Wu, X., Zheng, Y., Chen, E.: Large language models for generative information extraction: A survey. arXiv preprint arXiv:2312.17617 (2023) 19
64. Xu, Z., Chen, Z., Zhang, Y., Song, Y., Wan, X., Li, G.: Bridging vision and language encoders: Parameter-efficient tuning for referring image segmentation. ICCV (2023) 4
65. Yang, G., Huang, X., Hao, Z., Liu, M.Y., Belongie, S., Hariharan, B.: Pointflow: 3d point cloud generation with continuous normalizing flows. In: ICCV (2019) 3
66. Yang, J., Chen, X., Qian, S., Madaan, N., Iyengar, M., Fouhey, D.F., Chai, J.: Llm-grounder: Open-vocabulary 3d visual grounding with large language model as an agent (2023) 4
67. Yu, X., Xu, M., Zhang, Y., Liu, H., Ye, C., Wu, Y., Yan, Z., Zhu, C., Xiong, Z., Liang, T., et al.: Mvimgnet: A large-scale dataset of multi-view images. In: CVPR (2023) 4
68. Zeng, X., Vahdat, A., Williams, F., Gojcic, Z., Litany, O., Fidler, S., Kreis, K.: Lion: Latent point diffusion models for 3d shape generation. NeurIPS (2022) 2, 3
69. Zhang, J., Li, X., Wan, Z., Wang, C., Liao, J.: Text2nerf: Text-driven 3d scene generation with neural radiance fields (2023) 4
70. Zhang, X., Zhao, W., Lu, X., Chien, J.: Text2layer: Layered image generation using latent diffusion model. arXiv preprint arXiv:2307.09781 (2023) 2
71. Zhang, Y., Chen, W., Ling, H., Gao, J., Zhang, Y., Torralba, A., Fidler, S.: Image gans meet differentiable rendering for inverse graphics and interpretable 3d neural rendering. arXiv preprint arXiv:2010.09125 (2020) 3
72. Zhou, L., Du, Y., Wu, J.: 3d shape generation and completion through point-voxel diffusion. In: ICCV (2021) 2, 3
73. Zhou, X., Ran, X., Xiong, Y., He, J., Lin, Z., Wang, Y., Sun, D., Yang, M.H.: Gala3d: Towards text-to-3d complex scene generation via layout-guided generative gaussian splatting. arXiv preprint arXiv:2402.07207 (2024) 4

7 More Results

7.1 Text Splitting.

The Category Score Distillation Sampling (CSDS) requires splitting the input text prompts into individual objects. We employ GPT-4 for this purpose, a method commonly used and effective for information extraction [63]. We empirically found that GPT-4 has the ability to split very complex text prompts, as shown in Figure 12.

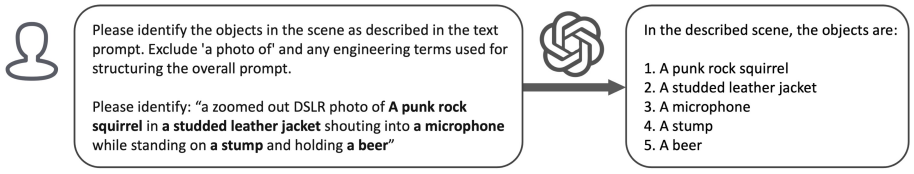

Fig. 12: Text splitting.

7.2 Applications on texture editing

We provide more results on text-guided texture editing, as shown in Figure 13. It can be observed that our method offers greater controllability compared to TEXTure [47].

7.3 Limitations

DreamDissector is likely to fail when objects are in very close contact, such as the body and clothing. We present two examples of this failure in the figure below. The primary reason is the challenge of obtaining clean NeCFs for such complex interactions.

7.4 Results on MVDream

We adopt Dreamfusion [42] as the backbone method for generating the initial text-to-3D NeRF for our main results. To verify the versatility of DreamDissector against different backbone methods, we employ MVDream [54], a recently proposed text-to-3D method, as the backbone. Results are shown in Figure 15. It can be observed that DreamDissector successfully dissects MVDream and produces independent textured meshes with improved geometries and textures.

7.5 Results on disentangled text-to-3D generation

We present additional results on disentangled text-to-3D generation, including those featured in the main paper. These results and text prompts are depicted in Figure 16, 17 and 18.

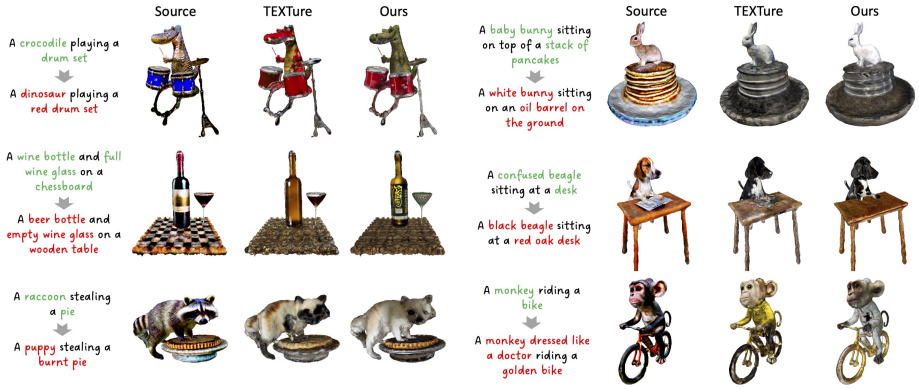

Fig. 13: Text-guided texture editing.

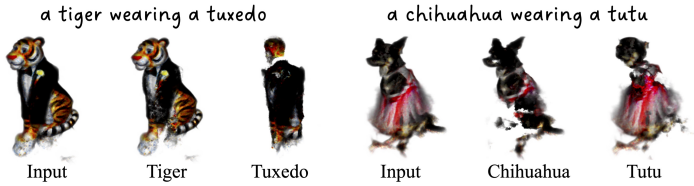

Fig. 14: Failure cases.

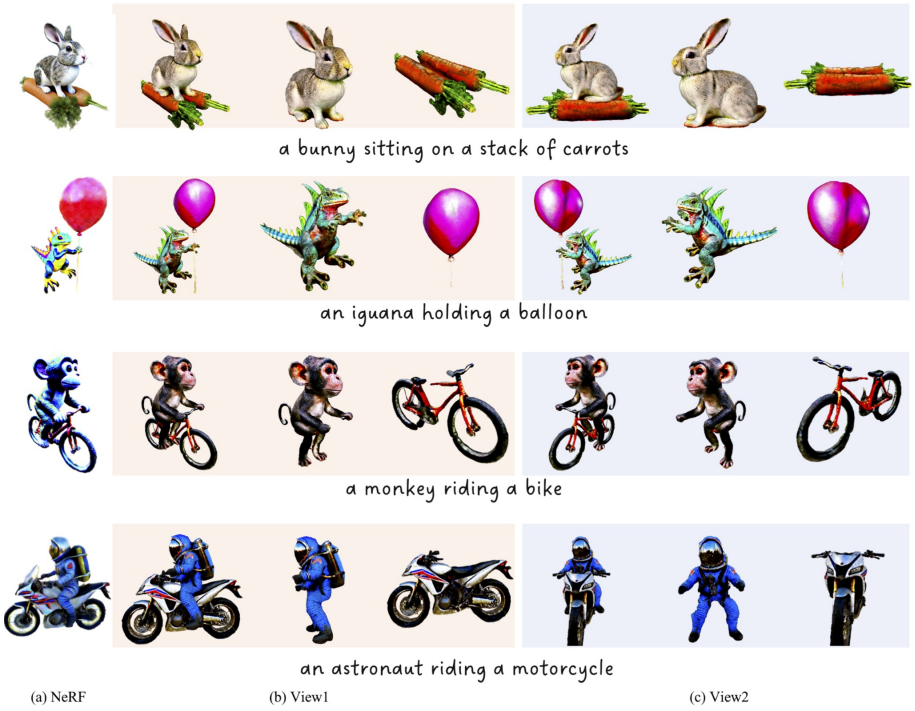

Fig. 15: Qualitative results based on MVDream [54].

7.6 Comparisons with the baselines

Additional comparisons are shown in Figure 19. It should be noted that negative prompting baseline, being intended to generate independent objects, does not associate with composed objects. Therefore, we regard the entire NeRF as the composed object. We also show the results of a text-guided scene generation method, Set-the-Scene [9], shown in Figure 20. These results illustrate the superior performance of our method.

Fig. 16: Qualitative results based on Dreamfusion [42].

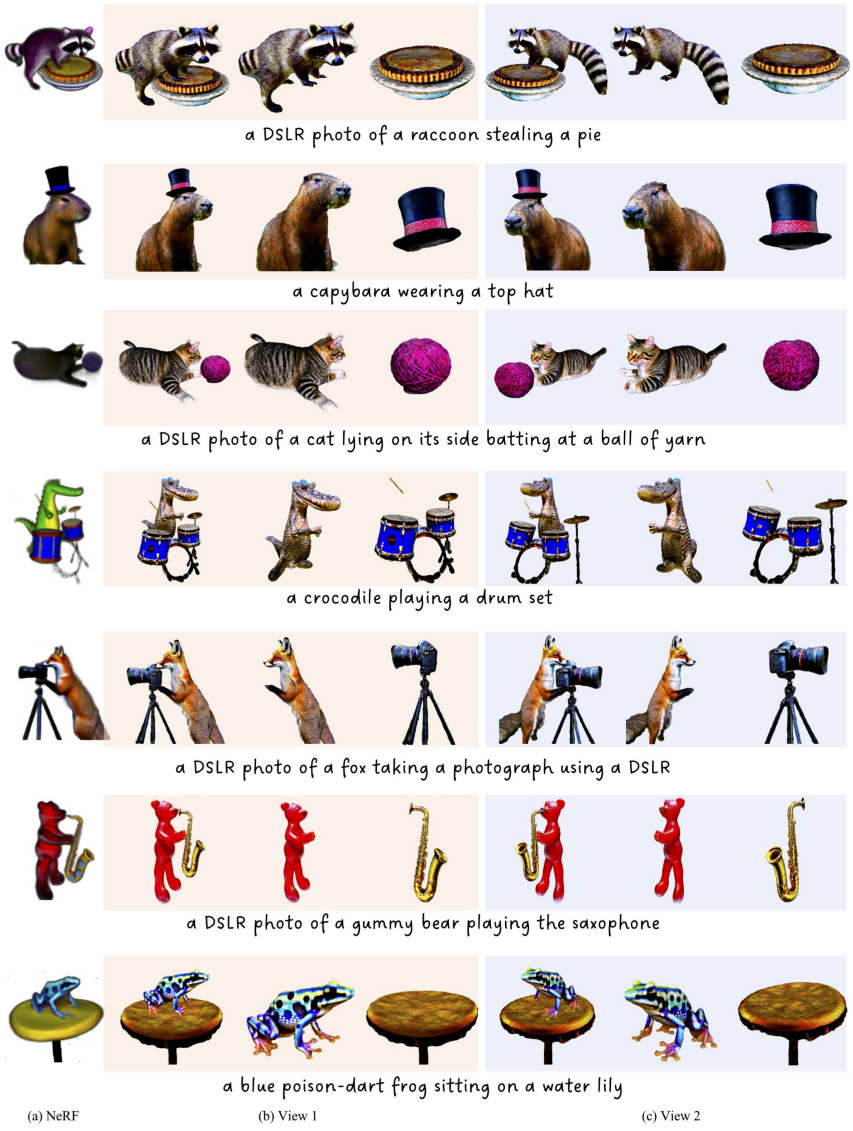

Fig. 17: Qualitative results based on Dreamfusion [42].

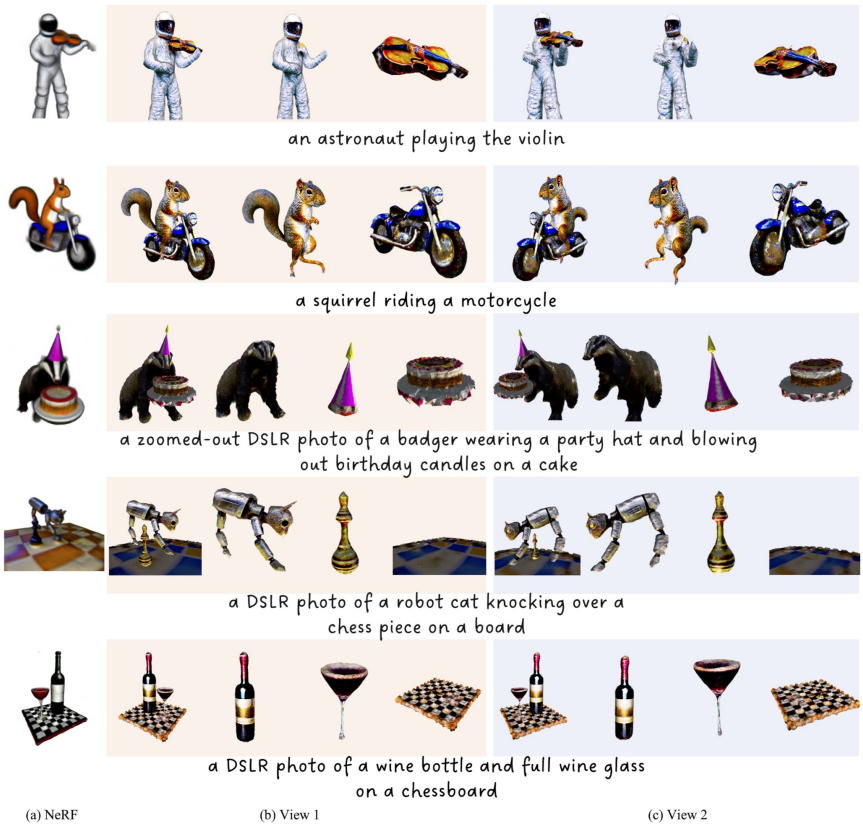

Fig. 18: Qualitative results based on Dreamfusion [42].

(a) Ours (b) Neg. Prompting (c) Composition

Fig. 19: Comparison with baseline methods.

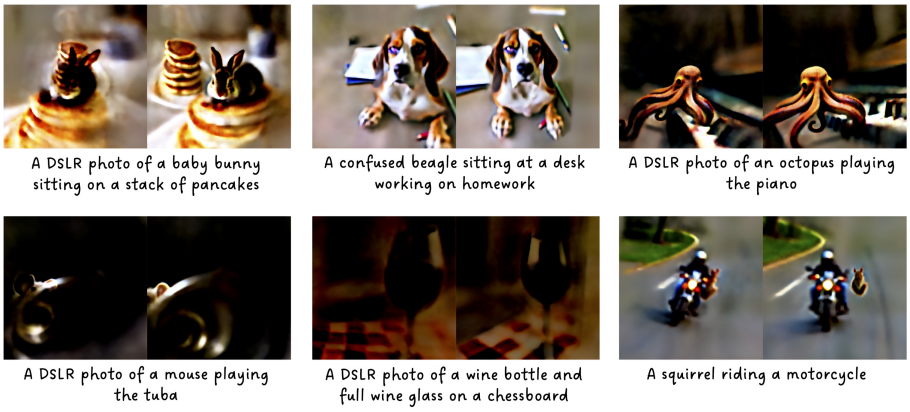

Fig. 20: Results on Set-the-Scene. We show the results on set-the-scene. It can be observed that set-the-scene struggles to model the object-interected scenes.