PhysiBuilder: Physics-Informed Prompt Learning for Interactive 3D Scene Generation and Editing

ZiAng Gu[1], Zhiyuan Ren[1], Feng Liu[1], Tianlong Chen[2], Xiaoming Liu[1]

[1]Michigan State University
[2]University of North Carolina at Chapel Hill
{guziang, renzhiy1, liufeng6, liuxm}@msu.edu, tianlong@cs.unc.edu

Abstract

Advancements in natural language processing and computer vision have opened new avenues for bridging textual descriptions and 3D visual content. Despite these progresses, creating coherent 3D scenes from textual inputs, particularly complex multi-object scenarios, remains a challenge. Existing methods often fall short in accurately interpreting and representing multiple objects from textual cues, such as "a table and a sofa". In light of this, we propose a novel framework, PhysiBuilder, that leverages physical principles to enable the generation and editing of 3D scenes directly from user-provided text. Specifically, we first devise an innovative 'ScenePlannerLLM' that automatically generates 3D scene priors conforming to real-world scenarios from the input text. These priors closely reflect real-world settings, encompassing detailed entity shape representations and a diverse range of scene compositions. Consequently, we introduce a new 'SceneConstructor' module, which models each 3D object through instance-specific NeRF. These two modules, in seamless collaboration, offer a user-friendly interaction with the generated 3D scenes. This integral feature facilitates real-time scene editing, supporting various operations such as moving, duplicating, or removing scene elements, without any re-training or fine-tuning. Experimental results demonstrate that our PhysiBuilder significantly increases fidelity to real-world physical composition during both generation and editing. The code and models will be released upon publication.

Introduction

Creating high-quality 3D content that accurately reflects the physical world holds great potential in various applications, including healthcare (Chengoden et al. 2023), virtual and augmented reality (Sun et al. 2022), robotics (Shridhar, Manuelli, and Fox 2022), and industrial design (Noghabaei et al. 2020). However, the intricacy of its creation process, which requires extensive artistic and technical expertise, significantly limits its wider accessibility. However, integrating natural language processing into this domain (*i.e.,* text-to-3D) could be transformative. Such an advancement could lower the barriers to entry for novices while simultaneously enhancing the productivity of experts. This not only democratizes the 3D content creation process but also opens up new possibilities for application and innovation in 3D content across various sectors.

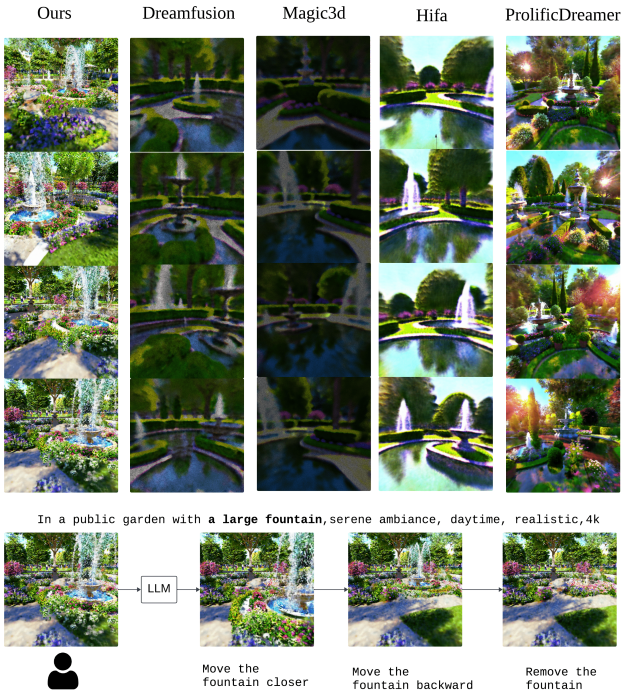


Figure 1: Comparison of our method with previous methods: Demonstrating how our method accurately generates a single identity of the fountain in response to the text prompt, in contrast to previous methods that frequently result in multiple, inconsistent identities.

Recent advancements in text-to-3D generation have been significant, as showcased by studies such as DreamFusion (Poole et al. 2022), Latent 3D Models (Metzer et al. 2023), Magic3D (Lin et al. 2023) and ProlificDreamer (Wang et al. 2023), Text2Room(Höllein et al. 2023), and more (Li et al. 2023a; Liu et al. 2023; Yi et al. 2023; Fang et al. 2023a,b; Cheng et al. 2023). DreamFusion, for instance, innovatively combines diffusion models with Neural Radiance Fields (NeRF) (Mildenhall et al. 2021) to transform text into detailed 3D models. It leverages the strengths of pre-trained text-to-image models, like Stable Diffusion (Rombach et al. 2022), to produce high-quality 2D images that guide the NeRF optimization. Alongside

NeRF, other methodologies such as Deep Marching Tetrahedra (Shen et al. 2021) and 3D Gaussian Splatting (Kerbl et al. 2023) are also explored in recent works (Lin et al. 2023; Tang et al. 2023; Chen et al. 2023a; Li et al. 2023b; Zhu and Zhuang 2023; Huang et al. 2023; Chen et al. 2023b; Katzir et al. 2023; Xu et al. 2023a), further diversifying approaches in text-to-3D generation.

Despite the advancements in text-to-3D generation, current methods have two limitations: **1) Janus Problem for scene**. This phenomenon pertains to the occurrence where the acquired 3D scene representation, rather than displaying the intended 3D scene, shows multiple canonical views of a scene from various perspectives. Specifically, when the model is tasked with generating a 3D sample of a bedroom, the produced object model displays multiple representations of the bed from its canonical viewpoint, as opposed to offering varied perspectives of the same bed (see Fig. 1). **2) Editing challenges.** Once the content is created, users face challenges in editing objects within the scene, especially if the output diverges from their expectation. While optimizing the object with another text prompt is possible, it often results in altering the entire scene, not just specific elements. The root cause of these limitations is the random positioning of objects during stable diffusion guidance, which leads to inconsistencies in 3D modeling. Methods like conventional NeRF (Poole et al. 2022; Lin et al. 2023; Wang et al. 2023; Zhu and Zhuang 2023) struggle to maintain consistent scene portrayal across multiple views due to this randomness, often reconstructing only a single object that satisfies the loss criteria instead of the full scene. While using 3D proxies in lieu of text prompts can enhance control of generated content (Cohen-Bar et al. 2023), their creation is time-consuming and demands specialized expertise.

To address these challenges, we propose a novel text to 3D scene generation and editing framework, **PhysiBuilder**. Our framework is meticulously designed to infuse *physical realism* into prompt design, ensuring that the generated scenes not only adhere to the laws of physics but also efficiently manage the generation of *multiple objects*. This integration into our training pipeline enables users to dynamically edit objects within these scenes in real-time, closely mimicking real-world interactions. Crucially, **PhysiBuilder** achieves these advanced editing capabilities without the need for retraining or fine-tuning the NeRF model. By leveraging language-driven interactions, our aim is to democratize 3D content creation, making it accessible and engaging for a broad spectrum of users, regardless of their technical expertise. This approach marks a significant step towards more intuitive, effective, and lifelike 3D modeling experiences.

For concreteness, our **PhysiBuilder** framework integrates two primary components: *ScenePlannerLLM* and *SceneConstructor*. Initially, *ScenePlannerLLM* generates 3D scene priors from textual prompts, ensuring realistic alignment with physical laws, including gravity and collision detection. This leads to scenes where objects maintain their intrinsic properties like volume and shape. Subsequently, *SceneConstructor* models each object using instance-specific NeRF, utilizing descriptions, entity names, and positions to reconstruct the 3D scene. The instance-specific NeRF is moved from local coordinates to scene coordinates based on the scene prior generated. It enables real-time, language-based scene adjustments by users, marking a major step forward in interactive, realistic 3D scene generation. The synergy between *ScenePlannerLLM* and *SceneConstructor* module allows for an intuitive and user-friendly interaction with the generated 3D scenes. This unique combination facilitates a range of real-time editing capabilities, including moving, duplicating, or removing elements within the scene, all achieved without the necessity for model retraining or fine-tuning.

In summary, the contributions of this work include:

◇ We develop **PhysiBuilder**, an innovative text-to-3D scene framework that facilitates the multi-object generation and offers flexible editing in the scene.

◇ We introduce a *ScenePlannerLLM* module, a self-consistent prompting method that creates realistic 3D scene priors from textual prompts, meticulously adhering to physical laws like gravity and collision.

◇ We introduce the *SceneConstructor*, a novel module that utilizes instance-specific NeRF to model individual objects in a scene, enabling precise 3D reconstruction and flexible editing of each object based on textual prompts.

◇ Through comprehensive experiments, we showcase that our approach improves generation quality and real-time editing capability, paving the way for practical applications in 3D content creation.

Related Works

Text to 3D Generation The success of text-to-image generative models has sparked significant interest in text-to-3D generation. Early efforts like CLIP-forge (Sanghi et al. 2022) synthesize objects using normalizing flow models to generate shape embeddings from text, but require voxel-based 3D training data, limiting scalability. Alternatives like DreamField (Jain et al. 2022) and CLIP-mesh (Mohammad Khalid et al. 2022) overcome this by using pre-trained image-text models (Yang et al. 2021) to optimize 3D representations (NeRFs and meshes) for higher text-image alignment in 2D renderings. Despite not using extensive 3D training data, they often produce less realistic 2D outcomes. Recently, DreamFusion (Poole et al. 2022), Magic3D (Lin et al. 2023), and ProlificDreamer (Wang et al. 2023) leverage advanced text-to-image diffusion models and sophisticated sampling strategies to produce more realistic and detailed 3D models and scenes from textual descriptions. They significantly enhance the quality and realism of 3D content, marking a major leap forward in the field. However, they are limited in generating scenes with multiple objects, focusing primarily on creating detailed and lifelike individual 3D models from text. This highlights a critical future direction in the realm of complex scene generation. Tab. 1 compares our method with prior works.

Layout Generation using LLM Diffusion models has significantly advanced text-to-image generation (Rombach et al. 2022; Saharia et al. 2022; Xu et al. 2023b; Liu et al. 2022). While they produce photorealistic images, control-

Table 1: Comparison of methods in text to 3D scene generation. 'Selective editing' pertains to the ability to modify specific objects within a scene. 'Instant editing' describes the process of altering the scene immediately, without model fine-tuning. 'Interactive Precision editing' is the modification of the scene by language instruction where no undesired artifacts are introduced. Note that SceneWiz3D and Gala3d are concurrent work.

Method	Required Input	3D Representation	# Objects	Editing		
				Selective	Instant	Interactive Precision
DreamField (Jain et al. 2022)	Prompt	NeRF	Single	×	×	×
Clip-forge (Sanghi et al. 2022)	Prompt	Voxel	Single	×	×	×
Clip-mesh (Mohammad Khalid et al. 2022)	Prompt	Mesh	Single/Multi	×	×	×
DreamFusion (Poole et al. 2022)	Prompt	NeRF	Single/Multi	×	×	×
Magic3D (Lin et al. 2023)	Prompt	DMtet+NeRF	Single/Multi	×	×	×
Fantasia3D (Chen et al. 2023a)	Prompt	DMtet	Single	×	×	×
ProlificDreamer (Wang et al. 2023)	Prompt	NeRF	Single/Multi	×	×	×
Hifa (Zhu and Zhuang 2023)	Prompt	NeRF	Single/Multi	×	×	×
DreamGussian (Tang et al. 2023)	Prompt	Gussian Splatting	Single	×	×	×
ControlRoom3D (Schult et al. 2023)	3D bounding box and prompt	Mesh	Multi	×	×	×
Set-the-Scene (Cohen-Bar et al. 2023)	3D Mesh and prompt	NeRF	Multi	✓	✓	×
SceneWiz3D: (Zhang et al. 2023)	Prompt	DMtet+NeRF	Multi	✓	✓	×
Gala3d (Zhou et al. 2024)	Prompt	Gaussian Splatting	Multi	✓	✓	✓
PhysiBuilder	Prompt	NeRF	Multi	✓	✓	✓

ling the generation results remains a challenge, *e.g.* positions of generated objects are unpredictable. Many methods aim to enhance controllability via conditions such as depth maps, segmentation maps or bounding boxes (Zhang, Rao, and Agrawala 2023; Gafni et al. 2022; Bar-Tal et al. 2023; Li et al. 2023d). Recently Large Language Models (LLM) like GPT have undergone significant advancements, tackling a diverse array of tasks. This expansion in capability is largely attributed to the models' extensive training, which equips them with a rich set of priors. As a result, these LLMs can generate detailed 2D or 3D bounding boxes (Cho, Zala, and Bansal 2023; Feng et al. 2023) from text descriptions alone. However, in many instances, LLM tends to generate layouts based on recognized text patterns and structures it was trained on. This often results in outputs that, while syntactically coherent, may not necessarily align with the intricacies of real-world 3D environments.

Recently Chain of Thought (CoT) prompting (Wei et al. 2022) emerges as a potent and resource-efficient strategy for problem-solving with LLMs for tasks requiring reasoning. It guides the LLM to generate intermediate steps or reasoning paths leading to the final answer, instead of the answer itself. When the descriptions are precisely formulated, they enable LLM to apply its autoregressive generation to solve new tasks without retraining LLM. Further studies have enhanced the CoT, such as Self-Consistent CoT, which corrects CoT reasoning errors by offering multiple reasoning paths (Wang et al. 2022), Tree of Thought, which evaluates different reasoning paths to dictate subsequent actions (Yao et al. 2023), and Graph of Thoughts, which represents LLM-produced information as a graph (Besta et al. 2023), among others (Huang et al. 2022; Li et al. 2023c; Yang et al. 2023; Zhou et al. 2022b,a). Despite these innovations primarily addressing issues in the language domain, they have yet to bridge the gap fully to understanding the 3D vision domain.

To enhance the capability of LLMs in generating realistic and spatially accurate 3D layouts, we introduce the nove approach of physics-informed prompt design. It incorporates principles of physics and spatial awareness directly into prompt generation. By crafting physics-grounded prompts, we enable the LLM to produce layouts that are not only visually coherent but also practically plausible. This represents a significant step forward in ensuring that LLM-generated layouts faithfully capture the practicalities and complexities of how objects and spaces coexist and interact in three-dimensional contexts, thereby making the models' outputs aligned with real-world environments.

3D Scene Editing In another area of study, focus has been given to the editing aspect of 3D modeling, e.g., Instruct-NeRF2NeRF(Haque et al. 2023) utilizes 2D image editing for the iterative refinement of neural fields. Further, there's an increasing emphasis on scene comprehension using NeRF. The goal is for NeRF to discern the distinct elements of a scene, with each object represented as a local NeRF (Xu et al. 2023c). This approach offers clarity in identifying which specific NeRF represents each object, enabling more precise editing by adjusting the respective local NeRF. Our method draws inspiration from the ray aggregation technique used in multi-local NeRF rendering, a concept also adopted by Set the Scene (Cohen-Bar et al. 2023) for scene generation. However, SoTA methods often lack support for interactive, real-time editing, requiring fine-tuning for optimal editing outcomes. Our approach takes a different path by establishing a robust decomposition model during training. This allows us to understand the interrelationships among local NeRFs, enhancing their segmentation. Consequently, after training, we don't need to change the NeRF parameters. Merely adjusting the position of the object-specific NeRF enables us to achieve the interactive editing capabilities without artifacts.

Preliminary

Recent advancements in text-to-3D object synthesis have shown the effectiveness of leveraging knowledge distilled from large-scale text-to-image diffusion models. Our approach aims to generate visually appealing 3D scenes from textual descriptions (*e.g.*, 'a table and a chair in the living room') that maintain their aesthetic integrity from any given viewpoint. To achieve this without extensive and costly 3D annotations for every object within the scene, we utilize the

Text-to-Image diffusion model, denoted as ϕ, as an instructional guide during the training phases. This method aligns with the principles established by DreamFusion (Poole et al. 2022), facilitating the creation of 3D content without direct 3D supervision. We succinctly describe the image rendering process from a 3D representation, which is parameterized by θ, as $\mathbf{x} = g(\theta)$. To create a noisy image variant $\mathbf{x}_t$, we introduce sampled noise $\epsilon \in \mathcal{N}(\mathbf{0}, \mathbf{I})$ over the original image $\mathbf{x}$. Given $\mathbf{x}_t$, time step t, and text embedding $\mathbf{y}$, the autoencoder estimates the sampled noise $\hat{\epsilon} = \epsilon_\phi(\mathbf{x}, \mathbf{y}, t)$. A weighted denoising score matching objective is derived as the training objective of the diffusion model: $\mathcal{L}_{diff}(\phi, \mathbf{x}) = \mathbb{E}_{t,\epsilon}[w(t)||\hat{\epsilon} - \epsilon||_2^2]$, where $w(t)$ is a weighting term. Score Distillation Sampling (SDS) (Poole et al. 2022) is used to optimize θ. Formally, the gradient is,

$$\nabla_\theta \mathcal{L}_{\text{SDS}}(\phi, g(\theta)) = \mathbb{E}_{t,\epsilon}\left[w(t)\left(\hat{\epsilon} - \epsilon\right)\frac{\partial \mathbf{x}_t}{\partial \theta}\right]. \quad (1)$$

ProlificDreamer(Wang et al. 2023) suggests modelling the 3D parameter θ as a random variable instead of a constant as in SDS and Variational Score Distillation (VSD):

$$\nabla_\theta \mathcal{L}_{\text{VSD}}(\phi, g(\theta)) = \mathbb{E}_{t,\epsilon}\left[w(t)\left(\hat{\epsilon} - \epsilon_{\text{lora}}(\mathbf{x}_t, \mathbf{y}, t)\right)\frac{\partial \mathbf{x}_t}{\partial \theta}\right], \quad (2)$$

where $\epsilon_{\text{lora}}(\mathbf{x}_t, \mathbf{y}, t)$ is the low-rank adaptation applied to the pre-trained diffusion model. Introducing VSD serves to mitigate issues commonly encountered in SDS, such as oversaturation, over-smoothing, and a lack of diversity.

Method

Fig. 2 illustrates the overall framework of our **PhysiBuilder**. We decompose the text to 3D scene generation task into two steps: (1) Coarse-level 3D Object generation (**ScenePlannerLLM**) and (2) fine-level modeling of multiple 3D objects (**SceneConstructor**). In this section, we also illustrate the capability for instant and interactive scene modifications within the proposed framework.

ScenePlannerLLM: Coarse-level 3D Object Generation

Volumetric primitive shape representation In the realm of 3D scene construction, a scene can be composed of multiple 3D objects, denoted by $\mathcal{S} := \{\mathbf{o}_i\}_{i=1}^N$. Within this composition, each object $\mathbf{o}$ is represented by a composite structure, consisting of a *variable number* of volumetric primitives. Specifically, we define an object $\mathbf{o}$ as the union of its constituent primitives, $\mathbf{o} := \bigcup_{j=1}^M P_j$, where each primitive is a cuboid P, and is characterized by its eight vertices.

The choice of cuboids as the primary geometric primitive offers a balance between structural simplicity and representational flexibility, enabling the approximation of a wide range of object shapes with a limited number of parameters. By adopting a coarse-level design philosophy, we lay the foundational groundwork for efficient scene planning and reconstruction, leveraging the computational prowess of LLMs, *i.e.*, ChatGPT-4 in this paper.

Formally, a single cuboid is defined by its corner vertices in 3D space. A cuboid P can be specified by the pair of diagonal vertices $[v_{\text{min}}, v_{\text{max}}]$, where v_{min} and v_{max} correspond to the minimum and maximum corners of the cuboid respectively. This simplification allows us to systematically encapsulate the spatial domain of the object with respect to its position and size within the 3D scene. The geometric integrity of each cuboid is maintained through predefined relationships between its vertices, ensuring accurate and consistent object representation in the spatial modeling process.

Leveraging LLMs for Scene Interpretation and Generation We employ a novel Input-Output Prompt (IO) methodology, utilizing LLMs to interpret scene descriptions and fabricate corresponding 3D scenes. This process, formalized as $S \sim p_\theta(\mathbf{o}_1, ..., \mathbf{o}_N|\text{IOprompt}))$, transforms narrative scene inputs into detailed, task-specific instructions complemented by illustrative examples, subsequently crafting a mesh representation from the structured LLM output.

Nevertheless, accurately mirroring the complex distribution of real-world data through direct box mesh generation involves intricate 3D reasoning challenges. To address this, we introduce a sophisticated approach inspired by Chain of Thought Prompting (CoT). This strategy, denoted as $\mathcal{S} \sim p_{\text{CoT},\theta}(\{P_1^i, ..., P_M^i\}_N^i = 1, \{\mathbf{o}_1, ..., \mathbf{o}_N\}|\text{CoTprompt}$, instructs the LLM to decompose the scene analysis into sequential, detailed descriptions of each object component. By transforming each object part, P_j^i, into a coherent textual narrative, we bridge the gap between abstract conceptualization and practical, three-dimensional structure. This meticulous breakdown facilitates a more nuanced understanding and generation of complex scenes, guiding the LLM through a series of 'thoughts' or steps to ensure a comprehensive and accurate representation.

Initially, we adopt a textual data representation strategy, which simplifies the transition from abstract descriptions to concrete geometric forms. For instance, a text entry like `{"object1": [[x_min, y_min, z_min, x_max, y_max, z_max], ...], "object2": [...], ...}` translates into a structured framework for subsequent mesh creation.

Physics self-consistent learning While LLMs are proficient in text generation and comprehension within linguistic contexts, their ability to directly interact with and understand 3D spaces is notably limited. This limitation hampers their spatial reasoning capabilities. To address this gap and improve the spatial reasoning of LLMs, we propose a series of designs aimed at moving beyond the conventional Input-Output or CoT prompt. These designs are intended to equip LLMs with enhanced tools for navigating and interpreting spatial information.

Our generation process involves three steps:

1. Coarse to Fine volumetric primitive shape generation

2. Selection based on self-consistency evaluation within the scene.

3. Refinement of the scene adhering to physics laws.

Given a scene description x, we first generate a coarse representation of the scene as a scene

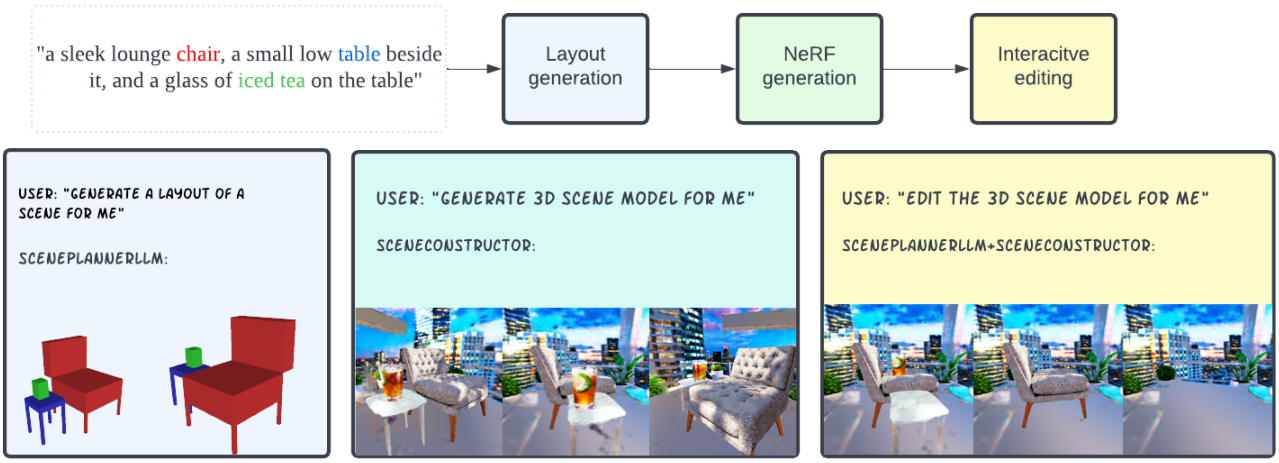


Figure 2: **Overview of our PhysiBuilder**. Our approach integrates three key components: **(a)** ScenePlannerLLM, which utilizes an LLM to generate the 3D layout of the scene that obeys the physics rules. **(b)** SceneConstructor training, which converts the layout into realistic 3d models **(c)**ScenePlannerLLM+SceneConstructor Interactive editing. The Scene Planner LLM can understand the language instructions and provide modifications to the layout. SceneConstructor will edit the 3D scene accordingly.

graph $G = (V, E)$, where G encapsulates the objects within the scene and their spatial relationships: $G \sim p_\theta^{\text{CoT}}(V_1, \ldots, V_N, E_1, \ldots, E_N | x)$. Here, $V = \{v_1, v_2, \ldots, v_n\}$ represents the objects, and $E = \{(v_i, v_j, R_{ij}) | v_i, v_j \in V\}$ encodes the relationships between these objects, incorporating relative positioning R_{ij}. `{"object1": {"positions": {"relation1": "object2"}}, "object2": {"positions": {"relation2": "object1"}}}`.

To refine this representation, we generate a series of detailed volumetric primitive shape representation $\mathcal{S}^{(j)}$ based on G: $\mathcal{S}^{(j)} \sim p_\theta^{\text{CoT}}\left(\{\mathcal{P}_i^1, \ldots, \mathcal{P}_i^M\}_{i=1}^N, \{\mathcal{O}_1, \ldots, \mathcal{O}_N\} \mid G\right)$ for $j = 1, \ldots, k$. The aim is to sample a population of scenes and select the most suitable scene based on evaluation of consistency.

To quantify the selection of the most consistent scene graph, we define a consistency score $\mathcal{C}$ for each sampled scene graph population $\mathcal{S}^j$. This score evaluates the adherence of $\mathcal{S}$ to G based on an LLM-based vote prompt, $\mathcal{C} \sim p_\theta(\mathcal{C}, \mathcal{Z}_1, \ldots, \mathcal{Z}_N | S^{(j)}, G)$. Here, Z represents the comments on the reasonableness of various volumetric primitive shape representations $\mathcal{S}^j$. The optimal volumetric primitive shape $\mathcal{S}_{\text{select}}$ with the highest consistency score is then selected, $\mathcal{S}_{\text{select}} = \underset{\mathcal{S} \in \{\mathcal{S}^{(1)}, \ldots, \mathcal{S}^{(k)}\}}{\arg\max} \mathcal{C}(\mathcal{S})$.

After selecting a suitable scene graph ($\mathcal{S}_{\text{select}}$) from the generated options, we refine it further to ensure it aligns with physical laws. While LLMs may not inherently compute physical positions, we enhance this step through two supplementary Python functions designed to evaluate the scene graph's compliance with physical laws. If any inconsistencies be detected, they are summarized in a prompt. This prompt then aids the LLM in further refining the scene graph to ensure optimal accuracy and compliance.

Collision Check: For objects A and B, a collision occurs if and only if the following conditions are met for all dimensions $d \in \{x, y, z\}$: $\mathbf{a}_{\min,d} \leq \mathbf{b}_{\max,d} \wedge \mathbf{b}_{\min,d} \leq \mathbf{a}_{\max,d}$. The collision is confirmed when at least one of the above conditions does not hold for any dimension.

Gravity Check: To ensure object A is correctly positioned with respect to gravity, two scenarios are considered:

1. *Object on the Ground:* Object A is considered to be on the ground if its minimum y-coordinate is 0: $\mathbf{a}_{\min,y} = 0$.
2. *Object on Top of Another Object:* Object A is considered to be on top of object B if they overlap in the x and z dimensions and A's minimum y-coordinate is greater than B's maximum y-coordinate: $(\mathbf{a}_{\min,x} \leq \mathbf{b}_{\max,x} \wedge \mathbf{b}_{\min,x} \leq \mathbf{a}_{\max,x}) \wedge (\mathbf{a}_{\min,z} \leq \mathbf{b}_{\max,z} \wedge \mathbf{b}_{\min,z} \leq \mathbf{a}_{\max,z}) \wedge \mathbf{a}_{\min,y} > \mathbf{b}_{\max,y}$.

Following the completion of these checks, the selected scene graph, $\mathcal{S}_{\text{select}}$, is updated to reflect the most physically plausible arrangement of objects:

$$\mathcal{S}_{\text{optimal}} \sim p_\theta^{\text{CoT}}\big(\{\mathcal{P}_i^1, \ldots, \mathcal{P}_i^M\}_{i=1}^N, \{\mathcal{O}_1, \ldots, \mathcal{O}_N\} \mid \mathcal{S}_{\text{select}}, \mathcal{R}_{\text{collision}}, \mathcal{R}_{\text{Gravity}}\big). \tag{3}$$

SceneConstructor: Train with edits

Based on the layout provided by a LLM, we represent each object in the scene as a separate local NeRF. To render an entire scene, we combine all foreground NeRF objects with a background NeRF. In the process of training our scene rendering model, we incorporate a 3D prior and utilize a novel approach called "train-with-edits". This technique integrates edits directly into the training process, resulting in more coherent and consistent editing outcomes after training.

Training of Local NeRFs During the local training phase, we sample spherical camera poses and iterate through every local NeRF, updating them one by one. All the rays are initialized from the origin in the world coordinate. We follow

the rendering of a single object as NeRF (Mildenhall et al. 2021). Our method employs the shape loss following Latent-NeRF and Set-the-Scene (Metzer et al. 2023; Cohen-Bar et al. 2023), which aligns the shape of the input bounding occupancy with the output occupancy from NeRF, thereby providing a prior for the 3D scene:

$$\mathcal{L}_{\text{Shape}} = -\sum_{c=1}^{C} \alpha_y_c(p) \log(\alpha_g_c(p)) \cdot \left(1 - e^{-\frac{d^2}{2\sigma_n}}\right). \quad (4)$$

Here d represents the distance to the proxy's surface, and α_g denotes the NeRF's occupancy, and α_y the occupancy of the specified mesh generated by the LLM.

Multi-Local NeRF Training Given all the local NeRFs represent different objects, we additionally represent the scene as a background NeRF follow the scene initialization as ProlificDreamer (Wang et al. 2023). The density function of the NeRF, denoted as σ, is described by $\sigma(\mu) = \lambda_\sigma \left(1 - \frac{|\mu|^2}{r}\right)$, where λ_σ is the strength of the density, r the radius of density, and μ the position coordinate. Setting the radius r to 2.5 and the density strength λ_σ to -10 results in a configuration where the density is lower at the center and higher around the perimeter, creating a region of empty space surrounded by high-density material.

The overall rendering of the scene is given by combination of all the local NeRFs represent the objects, and a background NeRF. To process each ray, we compute the query points along it starting from an origin point o in a specific direction d, leading to the ray $r_i = o + t_i d$, where t is the increment step. These rays are then converted into local coordinates by the rotation R and translation T, adhering to a present layout generated by LLM, as shown by:

$$r_i^k = R(r_i + T).$$

With the rays now in the transformed space r_i^k, we can derive a collection of densities σ_i^k and RGB colors c_i^k corresponding ray r_i for kth local NeRF. Following this, we sort the densities and RGB colors from all the local NeRF according to their step t_i, which also represents the position of the ray, and compute the cumulative RGB value by integral like NeRF, using the formula:

$$\hat{C}(r) = \sum_i T_i \alpha_i c_i = \sum_i T_i (1 - \exp(-\sigma_i \delta_i)) c_i,$$

where δ_i is the difference between consecutive steps, $t_{i+1} - t_i$.

Train with edits In the training of NeRF models, the loss function is applied in the image space. So during the rendering, it leads to an unclear delineation between foreground and background elements. This ambiguity can detrimentally affect the quality of generated results, occasionally causing the background NeRF to improperly overlap with the foreground, *e.g.*, the pillow on the sofa is learned to be on the foreground. This situation may seem fine for the generation. But as long as we start to further edit after the training process, the results will be detrimental. In light of this we leverage LLMs to generate a series of scene edits that maintain the overall coherence of the scene. This method will ensure that after editing the scene is still reasonable. We offer the original layout to the LLM, and ask it to provide the new layout with edits based on the original layout. Using the new layout that we generate for the scene we update the translation matrix T we use for the multi-object rendering, and the SDS loss is used to update the Local NeRFs.

Real-time Scene Editing

Scene editing within our framework is predominantly driven by language descriptions, with the LLM treating these as classification tasks to invoke specific editing actions. Importantly, all editing options below can be performed in real-time, eliminating the need for any fine-tuning steps.

- **Translation**: Responding to user requests, ScenePlannerLLM interprets and edit the 3D center of objects. This translation is then reflected in the rendering phase as the local NeRF adjusts its position.
- **Rotation**: ScenePlannerLLM generates various rotation settings based on the description, with these rotations integrated into the rays in rendering.
- **Object Removal**: ScenePlannerLLM is capable of omitting specific items from a scene. If the textual description specifies objects for removal, SceneConstructor will not render them, effectively removing them from the scene.
- **Duplication**: ScenePlannerLLM can recognize and replicate referenced objects, generating new translations and rotations to logically integrate these duplicates into the scene.

Experiments

Except for the ScenePlannerLLM, all our experiments are carried out on a single NVIDIA A6000 GPU, with each scene requiring roughly 60 minutes of training time. For the text to image generation we use Stable Diffusion V2.1. For LLM, we use GPT4 Turbo. More implementation details are in the supplementary.

3D Generation Evaluation

Qualitative Results We evaluate our method by comparing it to the text-to-3D baselines, DreamFusion (Poole et al. 2022), Mgaic 3D (Lin et al. 2023), and prolificDreamer (Wang et al. 2023), focusing on the quality of generation results. Qualitatively, the results are shown in Fig. 3. We can see that the identity mapping of our generated results and the text prompt is superior to the baselines.

User Study To assess our model's performance quantitatively, we organized a user study. Participants were presented with text prompts describing five different scenes. They were then asked to evaluate each scene from five distinct viewpoints: scene quality, physical plausibility, alignment with the text, scene composition, and overall coherence. Evaluations were based on a scoring system ranging from 1 to 10, with higher scores reflecting more favorable outcomes. Tab. 2 demonstrate that our results receive overall better rating than the baselines.

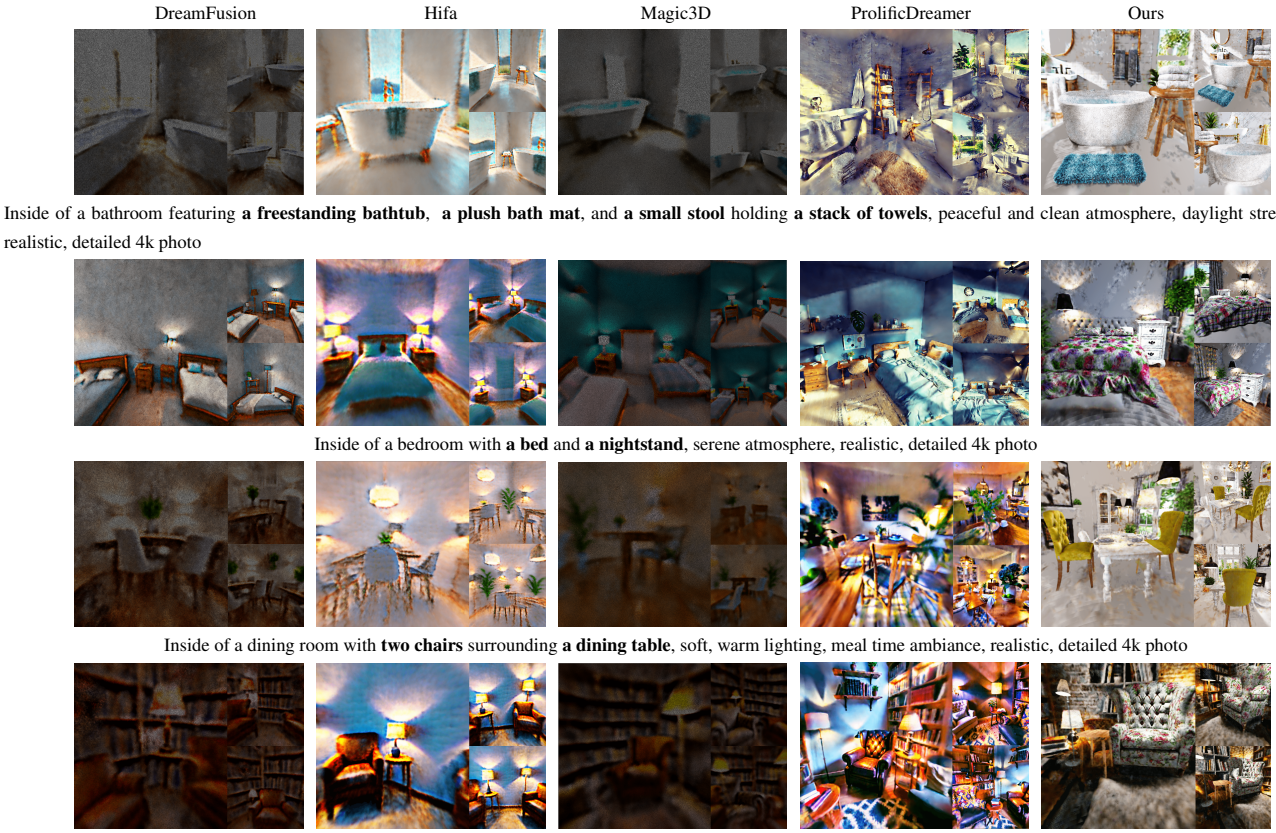


Figure 3: **Quality comparison of text-to-3D scene generation.** This figure highlights the differences in generation outcomes across various methods. Notably, our approach resultss in a significant increase in unintended identity mappings within the scene.

Table 2: User study of different methods in text to 3D scene generation. Participants evaluated each method using five distinct metrics, scoring them on a scale from 1 to 10. Higher scores represent a stronger preference.

Methods	Scene Quality	Physical Plausibility	Text Alignment	Scene Composition	Identity Consistency
DreamFusion	4.7	5	5	5	6.5
Hifa	6.8	7	5.9	7	6.1
Magic3D	4.9	5.6	5.1	5.2	6.1
ProlificDreamer	7.6	**7.4**	6.9	**8.1**	7.3
Ours	**7.8**	7.2	**7.3**	7.7	**8.5**

3D Layout Generation

In this section, we present the outcomes of our 3D layout generation process utilizing ScenePlannerLLM in response to text-based prompts. We compare Volumetric primitive shape generation among the standard prompting method, the chain of thought prompting method. The assessment of the generated 3D layouts is supported by a combination of user studies and a physics-based evaluation criterion to establish their efficacy.

In Fig. 4, we show the qualitative results of our prompting method as compared to other methods. For a comprehensive examination, we have created a dataset comprising text prompts for 25 distinct scenes. The dataset will be available in supplementary. Each scene is populated with 4 to 5 commonly encountered objects. In our selection process, we meticulously excluded any items that are unlikely to be positioned on the ground in real-world settings. We employed a physics-based scoring mechanism where scenes with objects either colliding or not adhering to gravitational principles were assigned a score of zero. Conversely, scenes devoid of such discrepancies received a score of one. The overall score for each method is calculated by aggregating the individual scores and then normalizing this sum by the total number of scenes.

Meanwhile, we also conduct a user study to evaluate the effectiveness of our method as shown in Tab. 4. Similarly, the user is asked to give a score between 1 to 10 where higher indicates better results. It is clear that our layout is substantially better than the two baselines.

Table 3: Physical principle metric.

Methods	Gravity	Collision
Standard	0.64	0.88
CoT	0.60	0.80
Ours w/o physics	0.64	0.68
Ours	**0.88**	**0.96**

Table 4: User study for 3D layout.

Methods	Physical	Scene Quality	Scene Composition
Standard	6.1	5.8	5.7
CoT	6.5	6.2	6.4
Ours	**7.8**	**7.8**	**8.0**

Ablation Study

Effect of the volumetric primitive shape representation Our layout has shown better representation power of the scene, but will this representation be helpful in terms of the

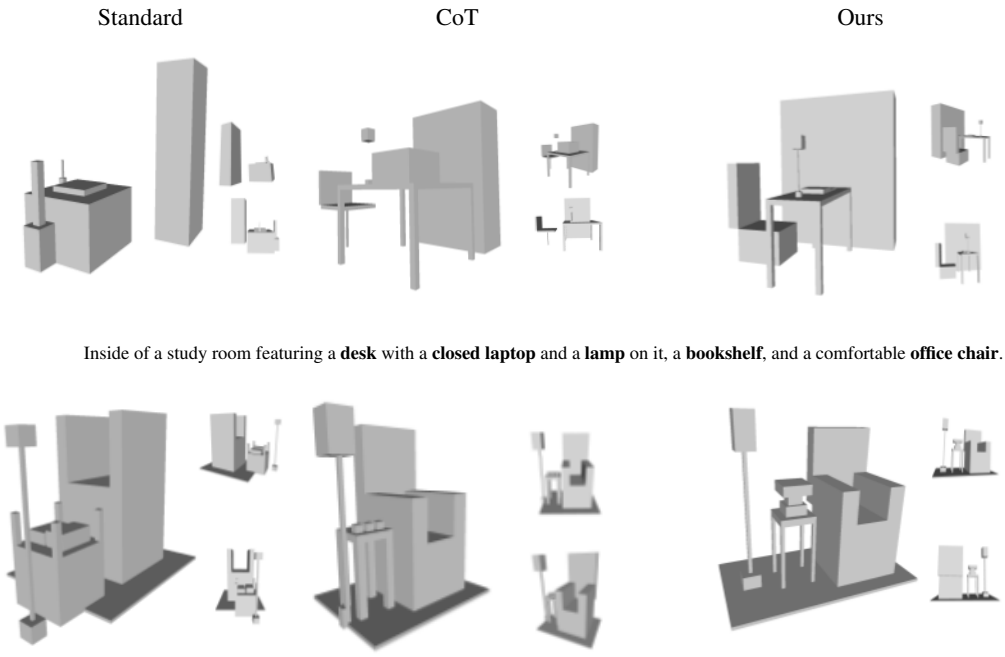


Figure 4: **Comparison of text-to-layout generation.** This figure demonstrates the layout generation outcomes from various prompting methods. Our approach yields a more coherent and physics-informed composition, maintaining consistency across different objects.

text to 3D scene generation? We generate another set of the layout use the bounding box the shape of the lounge chair is represented as a box-like shape from four different view angles, where as realistically the geometry of the lounge could not be a cube. As shown in Fig. 3 and Fig. 5, without the physical rules as the selection, the generated shape tends to have a box-like shape which is caused by a worse shape prior.

Effect of the physical prompt Through the collision and gravity metric we use, we quantitatively showcase our layout generated is able to be more adhere to the real physical world by adding the physical checks in Tab. 3. It demonstrates that employing physical-informed selection significantly enhances performance.

Limitations

Even though there have been notable advancements, there are instances where the background might still exhibit unclear, foggy shapes. Similarly, our approach, like other strategies that rely on optimization within the scope of SDS, requires several thousand iterations to achieve optimal outcomes. While ScenePlannerLLM excels at generating believable 3D layouts, it faces challenges in designing objects with rounded edges or those that deviate from square or rectangular shapes.

Conclusion

In this work, we introduce **PhysiBuilder**, an innovative framework for text to 3D scene generation. Our approach uniquely combines the adaptability of language models with the inherent property of the physics world. The cornerstone of PhysiBuilder is its ability to synthesize and edit highly detailed and contextually accurate 3D scenes directly from text prompts. Our method not only enhances the diversity and realism of the generated scenes but also provides user-friendly interactive editing performance. Our extensive user studies and qualitative assessments demonstrate the superiority of PhysiBuilder in terms of both generation quality and editing capability.

References

Bar-Tal, O.; Yariv, L.; Lipman, Y.; and Dekel, T. 2023. MultiDiffusion: Fusing Diffusion Paths for Controlled Image Generation. *arXiv preprint arXiv:2302.08113*.

Besta, M.; Blach, N.; Kubicek, A.; Gerstenberger, R.; Gianinazzi, L.; Gajda, J.; Lehmann, T.; Podstawski, M.; Niewiadomski, H.; Nyczyk, P.; et al. 2023. Graph of thoughts: Solving elaborate problems with large language models. *arXiv preprint arXiv:2308.09687*.

Chen, R.; Chen, Y.; Jiao, N.; and Jia, K. 2023a. Fantasia3D: Disentangling Geometry and Appearance for High-quality Text-to-3D Content Creation. In *ICCV*.

Chen, Y.; Pan, Y.; Li, Y.; Yao, T.; and Mei, T. 2023b. Control3d: Towards controllable text-to-3d generation. In *International Conference on Multimedia*.

Cheng, X.; Yang, T.; Wang, J.; Li, Y.; Zhang, L.; Zhang, J.; and Yuan, L. 2023. Progressive3D: Progressively Local



Figure 5: **Visualization of the bounding box generation results** All the objects in the scene tend to have a box-like shape. Highlighting the effectiveness of our mesh representation as compared to bounding boxes.

Editing for Text-to-3D Content Creation with Complex Semantic Prompts. *arXiv preprint arXiv:2310.11784*.

Chengoden, R.; Victor, N.; Huynh-The, T.; Yenduri, G.; Jhaveri, R. H.; Alazab, M.; Bhattacharya, S.; Hegde, P.; Maddikunta, P. K. R.; and Gadekallu, T. R. 2023. Metaverse for healthcare: A survey on potential applications, challenges and future directions. *IEEE Access*.

Cho, J.; Zala, A.; and Bansal, M. 2023. Visual Programming for Text-to-Image Generation and Evaluation. *arXiv preprint arXiv:2305.15328*.

Cohen-Bar, D.; Richardson, E.; Metzer, G.; Giryes, R.; and Cohen-Or, D. 2023. Set-the-Scene: Global-Local Training for Generating Controllable NeRF Scenes. In *ICCVW*.

Fang, C.; Hu, X.; Luo, K.; and Tan, P. 2023a. Ctrl-Room: Controllable Text-to-3D Room Meshes Generation with Layout Constraints. *arXiv preprint arXiv:2310.03602*.

Fang, S.; Wang, Y.; Yang, Y.; Tsai, Y.-H.; Ding, W.; Yang, M.-H.; and Zhou, S. 2023b. Text-driven Editing of 3D Scenes without Retraining. *arXiv preprint arXiv:2309.04917*.

Feng, W.; Zhu, W.; Fu, T.-j.; Jampani, V.; Akula, A.; He, X.; Basu, S.; Wang, X. E.; and Wang, W. Y. 2023. LayoutGPT: Compositional Visual Planning and Generation with Large Language Models. *arXiv preprint arXiv:2305.15393*.

Gafni, O.; Polyak, A.; Ashual, O.; Sheynin, S.; Parikh, D.; and Taigman, Y. 2022. Make-a-scene: Scene-based text-to-image generation with human priors. In *ECCV*.

Haque, A.; Tancik, M.; Efros, A. A.; Holynski, A.; and Kanazawa, A. 2023. Instruct-nerf2nerf: Editing 3d scenes with instructions. *arXiv preprint arXiv:2303.12789*.

Höllein, L.; Cao, A.; Owens, A.; Johnson, J.; and Nießner, M. 2023. Text2room: Extracting textured 3d meshes from 2d text-to-image models. *arXiv preprint arXiv:2303.11989*.

Huang, J.; Gu, S. S.; Hou, L.; Wu, Y.; Wang, X.; Yu, H.; and Han, J. 2022. Large language models can self-improve. *arXiv preprint arXiv:2210.11610*.

Huang, X.; Shao, R.; Zhang, Q.; Zhang, H.; Feng, Y.; Liu, Y.; and Wang, Q. 2023. Humannorm: Learning normal diffusion model for high-quality and realistic 3d human generation. *arXiv preprint arXiv:2310.01406*.

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

Katzir, O.; Patashnik, O.; Cohen-Or, D.; and Lischinski, D. 2023. Noise-free score distillation. *arXiv preprint arXiv:2310.17590*.

Kerbl, B.; Kopanas, G.; Leimkühler, T.; and Drettakis, G. 2023. 3d gaussian splatting for real-time radiance field rendering. *ToG*.

Li, M.; Zhou, P.; Liu, J.-W.; Keppo, J.; Lin, M.; Yan, S.; and Xu, X. 2023a. Instant3D: Instant Text-to-3D Generation. *arXiv preprint arXiv:2311.08403*.

Li, W.; Chen, R.; Chen, X.; and Tan, P. 2023b. SweetDreamer: Aligning Geometric Priors in 2D Diffusion for Consistent Text-to-3D. *arXiv preprint arXiv:2310.02596*.

Li, Y.; Lin, Z.; Zhang, S.; Fu, Q.; Chen, B.; Lou, J.-G.; and Chen, W. 2023c. Making Language Models Better Reasoners with Step-Aware Verifier. In Rogers, A.; Boyd-Graber, J.; and Okazaki, N., eds., *ACL*.

Li, Y.; Liu, H.; Wu, Q.; Mu, F.; Yang, J.; Gao, J.; Li, C.; and Lee, Y. J. 2023d. GLIGEN: Open-Set Grounded Text-to-Image Generation. *CVPR*.

Lin, C.-H.; Gao, J.; Tang, L.; Takikawa, T.; Zeng, X.; Huang, X.; Kreis, K.; Fidler, S.; Liu, M.-Y.; and Lin, T.-Y. 2023. Magic3D: High-Resolution Text-to-3D Content Creation. In *CVPR*.

Liu, M.; Shi, R.; Chen, L.; Zhang, Z.; Xu, C.; Wei, X.; Chen, H.; Zeng, C.; Gu, J.; and Su, H. 2023. One-2-3-45++: Fast single image to 3d objects with consistent multi-view generation and 3d diffusion. *arXiv preprint arXiv:2311.07885*.

Liu, N.; Li, S.; Du, Y.; Torralba, A.; and Tenenbaum, J. B. 2022. Compositional visual generation with composable diffusion models. In *ECCV*.

Metzer, G.; Richardson, E.; Patashnik, O.; Giryes, R.; and Cohen-Or, D. 2023. Latent-nerf for shape-guided generation of 3d shapes and textures. In *CVPR*.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*.

Mohammad Khalid, N.; Xie, T.; Belilovsky, E.; and Popa, T. 2022. Clip-mesh: Generating textured meshes from text using pretrained image-text models. In *SIGGRAPH Asia*.

Noghabaei, M.; Heydarian, A.; Balali, V.; and Han, K. 2020. A survey study to understand industry vision for virtual and augmented reality applications in design and construction. *arXiv preprint arXiv:2005.02795*.

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

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-resolution image synthesis with latent diffusion models. In *CVPR*.

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. 2022. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS*.

Sanghi, A.; Chu, H.; Lambourne, J. G.; Wang, Y.; Cheng, C.-Y.; Fumero, M.; and Malekshan, K. R. 2022. Clip-forge: Towards zero-shot text-to-shape generation. In *CVPR*.

Schult, J.; Tsai, S.; Höllein, L.; Wu, B.; Wang, J.; Ma, C.-Y.; Li, K.; Wang, X.; Wimbauer, F.; He, Z.; et al. 2023. Controlroom3d: Room generation using semantic proxy rooms. *arXiv preprint arXiv:2312.05208*.

Shen, T.; Gao, J.; Yin, K.; Liu, M.-Y.; and Fidler, S. 2021. Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. *NeurIPS*.

Shridhar, M.; Manuelli, L.; and Fox, D. 2022. Cliport: What and where pathways for robotic manipulation. In *CoRL*. PMLR.

Sun, J.; Gan, W.; Chen, Z.; Li, J.; and Yu, P. S. 2022. Big data meets metaverse: A survey. *arXiv preprint arXiv:2210.16282*.

Tang, J.; Ren, J.; Zhou, H.; Liu, Z.; and Zeng, G. 2023. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *arXiv preprint arXiv:2309.16653*.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; and Zhou, D. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Wang, Z.; Lu, C.; Wang, Y.; Bao, F.; Li, C.; Su, H.; and Zhu, J. 2023. ProlificDreamer: High-Fidelity and Diverse Text-to-3D Generation with Variational Score Distillation. *arXiv preprint arXiv:2305.16213*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *NeurIPS*.

Xu, X.; Lyu, Z.; Pan, X.; and Dai, B. 2023a. Matlaber: Material-aware text-to-3d via latent brdf auto-encoder. *arXiv preprint arXiv:2308.09278*.

Xu, X.; Wang, Z.; Zhang, G.; Wang, K.; and Shi, H. 2023b. Versatile diffusion: Text, images and variations all in one diffusion model. In *ICCV*.

Xu, Y.; Chai, M.; Shi, Z.; Peng, S.; Skorokhodov, I.; Siarohin, A.; Yang, C.; Shen, Y.; Lee, H.-Y.; Zhou, B.; et al. 2023c. DisCoScene: Spatially Disentangled Generative Radiance Fields for Controllable 3D-aware Scene Synthesis. In *CVPR*.

Yang, B.; Zhang, Y.; Xu, Y.; Li, Y.; Zhou, H.; Bao, H.; Zhang, G.; and Cui, Z. 2021. Learning object-compositional neural radiance field for editable scene rendering. In *ICCV*.

Yang, C.; Wang, X.; Lu, Y.; Liu, H.; Le, Q. V.; Zhou, D.; and Chen, X. 2023. Large language models as optimizers. *arXiv preprint arXiv:2309.03409*.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T. L.; Cao, Y.; and Narasimhan, K. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Yi, T.; Fang, J.; Wu, G.; Xie, L.; Zhang, X.; Liu, W.; Tian, Q.; and Wang, X. 2023. Gaussiandreamer: Fast generation from text to 3d gaussian splatting with point cloud priors. *arXiv preprint arXiv:2310.08529*.

Zhang, L.; Rao, A.; and Agrawala, M. 2023. Adding conditional control to text-to-image diffusion models. In *ICCV*, 3836–3847.

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

Zhou, D.; Schärli, N.; Hou, L.; Wei, J.; Scales, N.; Wang, X.; Schuurmans, D.; Cui, C.; Bousquet, O.; Le, Q.; et al. 2022a. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

Zhou, X.; Ran, X.; Xiong, Y.; He, J.; Lin, Z.; Wang, Y.; Sun, D.; and Yang, M.-H. 2024. GALA3D: Towards Text-to-3D Complex Scene Generation via Layout-guided Generative Gaussian Splatting. *arXiv preprint arXiv:2402.07207*.

Zhou, Y.; Muresanu, A. I.; Han, Z.; Paster, K.; Pitis, S.; Chan, H.; and Ba, J. 2022b. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*.

Zhu, J.; and Zhuang, P. 2023. HiFA: High-fidelity Text-to-3D with Advanced Diffusion Guidance. *arXiv preprint arXiv:2305.18766*.