UNIVERSITY OF AMSTERDAM

UvA-DARE (Digital Academic Repository)

SceneTeller: Language-to-3D Scene Generation

Öcal, B.M.; Tatarchenko, M.; Karaoğlu, S.; Gevers, T.

DOI
10.1007/978-3-031-73013-9_21

Publication date
2025

Document Version
Final published version

Published in
Computer Vision – ECCV 2024

License
Article 25fa Dutch Copyright Act (https://www.openaccess.nl/en/policies/open-access-in-dutch-copyright-law-taverne-amendment)

Link to publication

Citation for published version (APA):
Öcal, B. M., Tatarchenko, M., Karaoğlu, S., & Gevers, T. (2025). SceneTeller: Language-to-3D Scene Generation. In A. Leonardis, E. Ricci, S. Stefan, O. Russakovsky, T. Sattler, & G. Varol (Eds.), *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024 : proceedings* (Vol. LXXXV, pp. 362–378). (Lecture Notes in Computer Science; Vol. 15143). Springer. https://doi.org/10.1007/978-3-031-73013-9_21

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, P.O. Box 19185, 1000 GD Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (https://dare.uva.nl)

Download date:09 Nov 2026

SceneTeller: Language-to-3D Scene Generation

Başak Melis Öcal[1(✉)], Maxim Tatarchenko[2], Sezer Karaoğlu[1], and Theo Gevers[1]

[1] UvA-Bosch Delta Lab, University of Amsterdam, Amsterdam, The Netherlands
b.m.ocal@uva.nl

[2] Bosch Center for AI, Robert Bosch GmbH, Renningen, Germany

Abstract. Designing high-quality indoor 3D scenes is important in many practical applications, such as room planning or game development. Conventionally, this has been a time-consuming process which requires both artistic skill and familiarity with professional software, making it hardly accessible for layman users. However, recent advances in generative AI have established solid foundation for democratizing 3D design. In this paper, we propose a pioneering approach for text-based 3D room design. Given a prompt in natural language describing the object placement in the room, our method produces a high-quality 3D scene corresponding to it. With an additional text prompt the users can change the appearance of the entire scene or of individual objects in it. Built using in-context learning, CAD model retrieval and 3D-Gaussian-Splatting-based stylization, our turnkey pipeline produces state-of-the-art 3D scenes, while being easy to use even for novices. Our project page is available at https://sceneteller.github.io/.

Keywords: Text-to-3D scene generation · Language-guided generation · 3D Gaussian splatting scene stylization

1 Introduction

3D scene design has numerous applications in domains like architecture or game development. Designing high-quality 3D scenes, however, is a laborious and time-consuming process, challenging even for highly-skilled domain experts utilizing professional tools. This complexity often leaves novice users unable to customize their own spaces. Recent advancements in automatic text-to-image synthesis [22,41] have led to massive progress in text-to-3D creation [36–38], enabling the generation of 3D content only from text input. Such works create foundation for democratizing 3D design, supporting its accessibility for non-experts.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-73013-9_21.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Leonardis et al. (Eds.): ECCV 2024, LNCS 15143, pp. 362–378, 2025.
https://doi.org/10.1007/978-3-031-73013-9_21

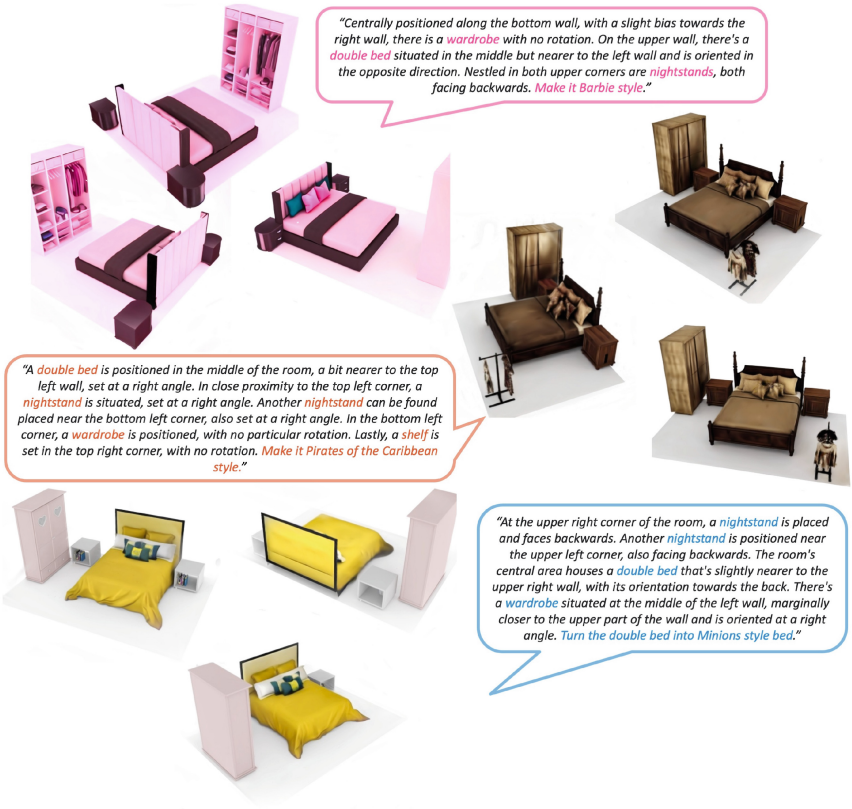

Fig. 1. With solely a textual prompt in natural language describing the desired spatial positions and orientations of objects within the scene, *SceneTeller* is able to generate realistic and high-quality 3D spaces. Additionally, by facilitating modifications to the style of the entire scene or individual objects within the scene through edit instructions, *SceneTeller* offers a practical and flexible framework for designing personalized rooms.

A pioneering work by Poole et al. [36] introduced object-centric text-to-3D generation based on 2D diffusion models coupled with NeRFs [31]. Several subsequent works proposed ways to improve the geometric accuracy, multi-view consistency, and appearance quality [6,27,30,50]. Despite their success in object-level generation, hallucinating complex large-scale scenes with multiple objects is still remaining a challenging problem. Some recent 3D scene generation methods [11,35,52] require a 3D layout as input, limiting their usability for novices. Others [16,23] produce results that violate global object arrangements and are not multi-view consistent.

Aiming to address both these shortcomings - insufficient user friendliness and unsatisfactory global consistency - we introduce *SceneTeller*, a framework for text-based 3D scene generation (Fig. 1). 1) We start with a premise that a 3D

scene generation framework should offer control over the placement and style of individual objects, allowing novice users to design their customized spaces through simple interactions with the system. Our approach achieves this by relying on text-based user interaction for scene and appearance specification. 2) In order to generate compositionally plausible 3D scenes, it is important to incorporate some degree of global information guidance. We implement this by conditioning the method on global layout descriptions.

To this end, we split the task of generating high-quality 3D scenes into three steps: 3D layout generation, object population and style editing. 1) When presented with a prompt describing the desired spatial positions and orientations of objects, *SceneTeller* generates an initial 3D layout of the scene by exploiting the Large Language Models (LLMs) through in-context learning, to ensure global consistency. 2) The generated layout is then used to assemble a 3D scene and populate it with CAD models from a database. 3) To facilitate scene editing, the assembled scene is fitted with a 3D Gaussian Splatting [24] representation. This representation is then used to stylize the scene according to the user-provided edit prompt targeting the entire scene or individual objects within it.

Extensive quantitative and qualitative evaluations show that *SceneTeller* performs significantly better than state-of-the-art methods, offering superior geometric fidelity and compositional plausibility. In brief, our contributions to the existing body of work can be summarized as follows:

- We introduce a turnkey pipeline for high-quality 3D scene generation yielding complex, realistic and globally consistent 3D scenes, given only scene descriptions in natural language. Our pipeline supports object- and scene-level appearance editing.
- We propose a novel LLM-based module for 3D layout generation trained through in-context learning, which provides control over individual object placement.
- We experimentally demonstrate that *SceneTeller* generates 3D scenes of higher quality and better consistency than state-of-the-art methods, while boasting unprecedented user friendliness and flexibility.

2 Related Work

2.1 3D Scene Representations

Rendering 3D scenes using implicit neural networks has seen remarkable success, notably with NeRF's [31] photo-realistic image synthesis. Extensions of NeRF improve reconstruction quality [3,25], training and rendering speed [5,17,32,39,43,55], and handling large-scale scenes [7,45]. Despite advancements, NeRF-based methods still struggle with slow rendering speeds. Recently, 3D Gaussian splatting (3DGS) was proposed for real-time, high-quality rendering using anisotropic 3D Gaussians and fast tile-based rasterization. Our framework employs 3DGS for scene representation and allows edits on these Gaussians, generating high-quality, 3D consistent scenes.

2.2 Text-to-3D Generation

Object-Centric Generation. 3D generation from textual prompts has recently grown in popularity. DreamFusion [36] introduces Score Distillation Sampling (SDS) to optimize NeRF representations by leveraging an off-the-shelf 2D diffusion model, paving the way towards text-driven 3D content creation [29,37,38,57]. Following [36], Magic3D [27] employs a coarse-to-fine optimization scheme, exploiting low-resolution diffusion priors, and then enhancing the texture. To achieve high-fidelity mesh generation, Fantasia3D [6] disentangles the appearance from geometry. Latent-NeRF [30] optimizes a NeRF in the latent space of StableDiffusion [41], and additionally proposes incorporating 3D priors to generate a target geometry. To address the over-smoothing, over-saturation and lack of diversity problems within the SDS, ProfilicDreamer [50] proposes a principled particle-based variational framework, namely Variational Score Distillation (VSD). Following the same motivation, [49] presents Perturb-and-Average Scoring to estimate the score for non-noisy images.

With the recent introduction of 3D Gaussian Splatting [24], several works attempt to combine 3DGS and diffusion models. GaussianDreamer [54] employs a 3D diffusion model to generate the initialized point clouds for 3D Gaussians, and then further optimize the Gaussians using SDS with a 2D diffusion model. [9,47] follows a two-stage pipeline for geometry optimization and appearance refinement. GaussianDiffusion [26] focuses on the challenge of achieving multi-view consistency in 3D generation and proposes combining structured noise with a variational 3DGS technique. Despite their success in generating object-level scenes, both NeRF-based and 3DGS-based methods fall short in hallucinating complex scenes with multiple objects.

Scene Generation. To represent multiple objects within the scenes, existing works [11,28,35,52] rely on compositional NeRFs that are initialized based on user-defined object proxies. However, they are usually restricted to produce only narrow inward-facing views of the rooms. Also, the requirement of having 3D layouts as input is restrictive for final users. Orthogonal to those, SceneScape [16] and Text2Room [23] generate scenes following the *render-refine-repeat* paradigm by iteratively warping and inpainting the previously generated image according to the specified camera motion. This strategy does not allow object-level control during scene generation, and the results tend to exhibit multi-view inconsistencies. LucidDreamer [10] employs the same paradigm for an initial point cloud reconstruction, and then further optimizes the scene using 3DGS. Recently, several works introduced techniques for 3DGS scene editing, by manipulating the training images using 2D diffusion models [8,13,48] and incorporating the edited images into 3DGS scenes for further optimization. Concurrent to our work, [56] initializes a 3DGS scene from the layouts created using LLMs and optimizes it using a diffusion model. However, [56] lacks control over the generated layout in terms of positions and orientations of objects.

2.3 LLMs for Vision

With the inclusion of grounding information in LLMs, they demonstrate enhanced reasoning capabilities for visual concepts. Recently, the visual reasoning abilities of LLMs have been utilized for 3D tasks such as avatar generation [40], instruction-driven 3D modeling [44], 3D editing [14]. Exploiting the spatial reasoning capabilities of LLMs, LayoutGPT [15] extends beyond its 2D counterparts [34] to generate random 3D layouts of specific room types through in-context learning. However, LayoutGPT lacks support for controlling the generated layout. The 3D layout generation module of *SceneTeller* is the first of its kind allowing control over individual object placement in accordance with the provided natural language prompts describing the desired positions of objects.

3 Method

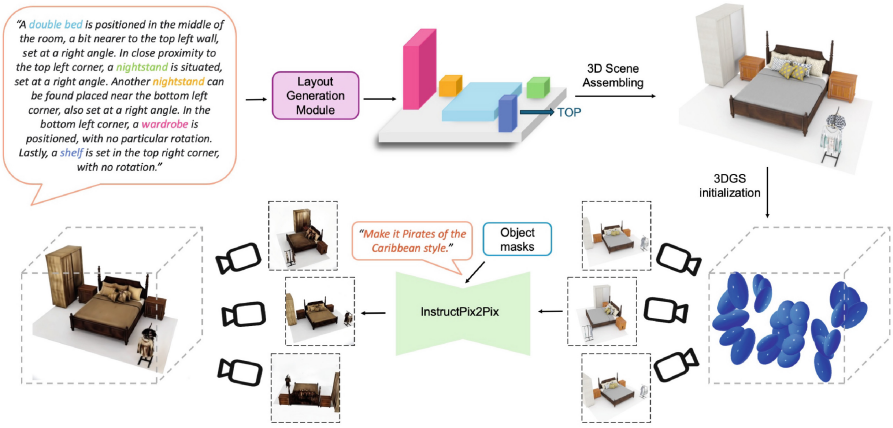

Fig. 2. Overview of our method. Given a textual prompt in natural language delineating the desired spatial positions and orientations of objects within the scene, a 3D scene layout is generated using in-context learning. An initial 3D scene is assembled for the predicted layout, which is then fitted with a 3D Gaussian Splatting representation. This representation is then used to stylize the scene according to the user-provided text prompt, and subsequently render the final images of the scene.

The pipeline of *SceneTeller* is divided into three main stages: language-driven 3D layout generation, 3D scene assembling from the layout and 3D scene stylization. In the first stage, a 3D scene layout is generated based on the given text prompt (Sect. 3.1). Then, an initial 3D scene is assembled for the predicted layout (Sect. 3.2). After that, a 3D Gaussian Splatting representation is fitted to the assembled scene. This representation is then used to stylize the scene according to the user-provided text prompt, and subsequently render the final images of the scene (Sect. 3.3. The scheme of the framework is illustrated in Fig. 2.

3.1 Language-Driven 3D Layout Generation

The task of this module is to produce a collection of object bounding boxes conditioned on a brief textural description of their spatial positions within the scene in plain language. More formally, the input is a condition $\mathcal{C}$, consisting of a textual description $\mathcal{Y}$, along with the scene type and scene dimensions information. The output is a 3D layout represented as a set of 3D bounding boxes $\mathbf{b}_i \in B = \{\mathbf{b}_i\}_{i=1}^{N}$ for each of the N referenced objects. Each 3D bounding box $\mathbf{b}_i = (c_i, \mathbf{t}_i, \mathbf{s}_i, o_i)$ consists of the following components: category name c_i, box center $\mathbf{t}_i = (x_i, y_i, z_i) \in \mathbb{R}^3$, box dimensions $\mathbf{s}_i = (w_i, h_i, d_i) \in \mathbb{R}^3$ and orientation $o_i \in \mathbb{R}$.

The textual description $\mathcal{Y}$ is comprised of multiple sentences, each specifying the position and orientation of a single object within a canonical coordinate system. The canonical system is chosen to represent the scene in a 2D perspective, to ensure that the descriptions closely resemble those produced by a human observer. $\mathcal{Y}$ is initially generated using a rule-based approach formulated in three steps as follows: 1) The canonical coordinate system is partitioned into a 3×3 grid, creating rectangular patches. 2) Each of the predicted objects is assigned to a patch based on the center position $\mathbf{t}_i$ and orientation o_i of its bounding box within these patches. 3) Per-object sentences are then formulated according to which patch the object is assigned to and object's orientation, *e.g.* for the object assigned to the top-left patch, with category name c_i and 90° orientation, we generate: *"A c_i is placed at the top-left corner of the room, with a perpendicular orientation."* Given the relatively limited linguistic variation inherent in rule-based descriptions, initial textual templates are provided to an LLM to serve as a basis for paraphrasing.

LayoutGPT [15] focused on improving an LLM's interpretation of the spatial knowledge by using a structured format featuring attribute-value pairs. We follow a similar strategy and adopt the standard CSS (Cascading Style Sheets) format to describe $\mathbf{b}_i$. Bounding box values are mapped to standard CSS format attributes and category name is employed as the selector.

Prompt Construction. Recent progress in the field has showcased the impressive in-context learning capabilities of LLMs, which entail conditioning these models with both natural language instructions and a select set of task demonstrations. Following this conditioning, the LLM is then tasked with completing further instances based on the provided demonstrations. In line with this, our prompts to LLM include three main parts: task specifications, in-context exemplars and the query condition:

- **Task specifications:** Following prior work [42,51] on enhancing LLMs instruction following capabilities, a task description is incorporated at the beginning of each prompt, which explains the goal of the task, establishes a standard for the 3D layout format in CSS style and provides unit information for attributes. Additionally, constraints are integrated into the prompts to guide the LLM and minimize errors (e.g., predicting overlapping boxes or placing objects out of bounds), during task completion.

- **In-context learning:** Supporting exemplars for the in-context learning are selected by adopting the retrieval-based approach used in [2,15,53]. When provided with a set of supporting exemplars $S = \{(\mathcal{C}_m^s, \mathbf{b}_m^s)\}_{m=1}^M$ and the queried condition $\mathcal{C}_q$, the function $f(\mathcal{C}_k^s, \mathcal{C}_q) = \|rl_k - rl_q\|^2 + \|rw_k - rw_q\|^2$ is computed between each element of the set and $\mathcal{C}_q$ following [15], where rl and rw are the length and width of the scenes. Top-k supporting exemplars with the shortest distance to $\mathcal{C}_q$ are selected for in-context learning, provided to LLM with the same format with $\mathcal{C}_q$.
- **Query condition:** The inference condition $\mathcal{C}_q$, for which we want to predict the layout.

3.2 3D Scene Assembling from Layouts

To assemble a scene based on the generated layouts, we rely on object retrieval. More specifically, for each of the bounding boxes, the nearest 3D model of the corresponding predicted category is retrieved from the furniture dataset based on the Euclidean distance of the bounding box dimensions as follows:

$$d(\mathbf{s}_i, \mathbf{s}_{cand}) = \sqrt{(w_i - w_{cand}))^2 + (h_i - h_{cand}))^2 + (d_i - d_{cand}))^2}, \quad (1)$$

where $\mathbf{s}_{cand}$ denotes the bounding box dimensions for the candidate object from the dataset. The retrieved 3D model is placed within the scene according to the predicted box center $\mathbf{t}_i$ and orientation angle o_i.

3.3 3D Scene Stylization

Given a generated 3D scene, the users should be able to modify either the overall appearance of the scene or specific objects within it. Our approach facilitates scene editing by training a 3D Gaussian splatting model representing the generated scene and refining this representation according to the provided edit instruction, offering a flexible framework for 3D scene generation.

3D Gaussian Splatting Scene Representation. 3D Gaussian splatting (3DGS) [24] is a recent pioneering work for novel view generation, which represents the underlying scene as a collection of anisotropic 3D Gaussians defined by their center positions $\mu \in \mathbb{R}^3$ and 3D covariance matrices $\mathbf{\Sigma}$ parameterized as:

$$\mathbf{\Sigma} = \mathbf{RSS}^{\mathrm{T}}\mathbf{R}^{\mathrm{T}}. \quad (2)$$

$\mathbf{R}$ denotes the rotation matrix and $\mathbf{S}$ is the scale matrix. Each 3D Gaussian is assigned a color c represented with spherical harmonics (SH) coefficients, to capture the view-dependent appearance. To allow α-blending of splats, Gaussians are associated with an opacity value $\alpha \in \mathbb{R}$.

Unlike NeRF [31] which relies on volume rendering, 3DGS enables faster training and rendering through differentiable rasterization. A set of 3D Gaussians is rendered by projecting into the camera's image plane as 2D Gaussians, and

assigned to individual image tiles. The color of each pixel $\mathbf{p}$ on the image plane is then determined as follows:

$$C(\mathbf{p}) = \sum_{i \in \mathcal{N}} c_i \sigma_i \prod_{j=1}^{i-1} (1 - \sigma_j), \qquad \sigma_i = \alpha_i e^{-\frac{1}{2}(\mathbf{p}-\mu_i)^T \Sigma_i^{-1} (\mathbf{p}-\mu_i)} \tag{3}$$

where $\mathcal{N}$ denotes the Gaussians in this tile, σ_i represents the influence of the Gaussian on the image pixel and μ_i, $\mathbf{\Sigma}_i$, c_i, α_i are the position, covariance, color and opacity of the i-th Gaussian respectively. For optimization, a weighted combination of $\mathcal{L}_1$ loss and SSIM loss is employed.

To initialize the centers of the 3D Gaussians, we utilize the mesh vertices of the retrieved objects and create a scene point cloud. This generated scene point cloud replaces the Structure-from-Motion (SfM) points used in the original work. To obtain the required training images for 3DGS optimization, we render the generated scene into RGB images I. We additionally render 2D segmentation masks $\mathcal{M}$ to be used for 3D Gaussian scene editing.

Scene Editing. For editing our 3DGS scenes based on the provided edit instruction, we build on top of Instruct-GS2GS [48] which in turn extends Instruct-NeRF2NeRF [20] for 3DGS scenes. Given the images used for training the 3DGS scene, Instruct-GS2GS updates the images individually using a diffusion model, namely Instruct-Pix2Pix [4]. The diffusion model takes three inputs: 1) an unedited conditioning image I_0^v, a text instruction c_T and the noisy version of the current render I_i^v at optimization step i, where v denotes the viewpoint from which the images are captured. Formally, the process of updating a single image is defined as:

$$I_{i+1}^v \leftarrow U_\theta(I_i^v, t; I_0^v, c_T), \tag{4}$$

where t is the noise level within the constant range $[t_{\min}, t_{\max}]$, U_θ is the sampling process of DDIM [41] and I_{i+1}^v is the edited image respecting the text instruction c_T and the unedited conditioning image I_0^v. The training is performed by editing all the training images during a dataset update at every 2.5k training iterations.

Despite achieving general scene-level stylization, [48] lacks of support for more fine-grained object-level editing. To enable object-level edits, we use the binary masks $m_{\{o\}_{k=1}^K}$. Each $m_{\{o\}_{k=1}^K}$ is obtained by binarizing 2D segmentation masks $\mathcal{M}$ for the set of objects to edit $\{o\}_{k=1}^K$. The set $\{o\}_{k=1}^K$ is defined by extracting the referenced category names from the text instruction c_T. Having the unedited conditioning image I_0^v, the edited image I_{i+1}^v, and the binary mask $m_{\{o\}_{k=1}^K}$, we keep only the edits at the pixels of the target object set:

$$I_{i+1}^v = m_{\{o\}_{k=1}^K} \odot I_{i+1}^v + (1 - m_{\{o\}_{k=1}^K}) \odot I_0^v, \tag{5}$$

where $\odot$ denotes element-wise multiplication of the image pixels. This way, the other pixels within I_{i+1}^v are set to their unedited versions, enabling an object-level editing of training images. While training the 3DGS scene with the edited images, the mask $m_{\{o\}_{k=1}^K}$ is also applied for $\mathcal{L}_1$ and SSIM losses, ensuring gradient propagation only to the target objects.

4 Experiments

Datasets. To create a set of supporting exemplars for the layout generation, we employ the 3D-FRONT dataset [18], a large-scale indoor scene dataset with 18,797 rooms furnished with 3D textured objects from 3D-FUTURE [19]. Following the same pre-processing steps as [33] and excluding non-rectangular floor plans from the test set similar to [15], we obtained 3397 train, 453 validation, 423 test bedroom and 690 train, 98 validation, 53 test living room scenes. During 3D scene assembling, 3D models are retrieved from the 3D-FUTURE dataset.

Implementation Details. We use ChatCompletions API[1], which enables a series of dialogue exchanges between the user and the LLM, to query GPT-4 [1]. BlenderProc[2] is employed to render 250 images per assembled scene with 512×512 resolution. After relocating the center of each scene to the origin of the coordinate system, we sample camera positions on the upper hemisphere at a distance of $1.5r$ from the center where r is the diagonal length of the room, with an elevation angle of 35°. We use the Splatfacto model from NeRFstudio [46] for 3DGS training, by optimizing each scene for a maximum of 20k iterations. For editing the 3DGS training images, we use InstructPix2Pix [4] as the diffusion model, with classifer guidance scales $s_T = [3.5, 12.5]$ for the text and $s_I = 1.5$ for the image. The dataset updates are performed every 2.5k iterations following Instruct-GS2GS [48]. The InstructGS2GS training takes approximately 15–20 minutes per scene on a single A6000, most of it spent on diffusion model sampling. The rendering speed is 101.21 fps.

4.1 3D Layout Generation

Evaluation Setup and Baselines. In this section we evaluate the first stage of our approach, therefore we only focus on assessing the generated layouts. To our knowledge, *SceneTeller* is the first approach that conducts conditional layout generation guided by text prompts to control the position and orientation of every object, targeting a user-friendly framework for novices. Hence, replicating an interaction mode that is natural for humans is pivotal for our framework. Based on this, our layout generation performance is compared with the performance of *humans* solving the same task. In particular, each participant of our user study is presented with a series of textual descriptions, each accompanied by an empty layout according to the room dimensions. For each text prompt, participants are requested to draw bounding boxes and label them with the object names. Each text prompt is given to at least 2, at most 3 participants. This study is conducted with 12 participants and 40 text prompts. To generate text prompts, 40 layouts are selected from the test set and their text prompts are generated by providing the initially rule-based formed sentences to GPT for paraphrasing, as described in Sect. 3.1.

[1] See https://platform.openai.com/docs/api-reference/chat.
[2] See https://github.com/DLR-RM/BlenderProc.

Metrics. Based on the predicted number of bounding boxes in each category, we report recall, precision and accuracy [15]. An accuracy score of 100 indicates that the number of bounding boxes and their corresponding categories precisely match the ground-truth annotations. Otherwise, a score of 0 is assigned to the scene. All values are in percentage (%). We additionally report mean Intersection over Union (mIoU) between the predicted and ground-truth bounding boxes and out-of-bound rates, indicating the percentage of scenes where furniture extends beyond the floor plan boundary. We refer to the supplementary material for detailed metric descriptions.

Results. Table 1 presents the quantitative evaluation results of 3D layout generation. Our in-context-learning-based 3D layout generation module performs better than human participants in terms of mIoU, showcasing the effectiveness of LLMs' visual reasoning capabilities. In-context exemplars further enhance this performance by aiding in reasoning about the current data distribution. However, human participants possess their own understanding of the physical world, leading to variations in their notions of object sizes and prompt reasoning, resulting in slightly poorer performance in this assessment. Both our layout generation module and human participants performed excellently in terms of predicting the number of bounding boxes per category. Humans sometimes forget or miss objects mentioned in the prompt, which explains their slightly imperfect recall score. The ground-truth layouts from the dataset involve objects slightly exceeding the floor plan boundary, leading to 15.15% OOB rate. Human participants are able to place the objects within the layout more accurately.

Table 1. Quantitative evaluation of 3D layout generation. The proposed in-context learning based 3D layout generation module performs better than human participants in terms of predicting object sizes, enhanced by in-context exemplars aiding in reasoning about the current data distribution. All values are in percentage (%).

Method	OOB ↓	rec.↑	prec.↑	acc.↑	mIoU ↑
Ours	56.78	100.00	100.00	100.00	44.60
Human	12.09	98.79	100.00	95.00	28.70
GT	15.15	100.00	100.00	100.00	100.00

4.2 3D Scene Generation

Baselines. Our approach is evaluated and compared with three state-of-the-art 3D scene generation methods. Set-the-scene [11] employs compositional NeRFs that are initialized based on user-defined object proxies to have more control over individual objects. To compare with [11], layouts generated by our method are provided as input to their pipeline. GSGEN [9] is a 3DGS-based approach,

following a two-stage pipeline for geometry optimization and appearance refinement. LucidDreamer [10] employs render-refine-repeat paradigm by iteratively warping and inpainting the previously generated image to create an initial point cloud, which is then used for 3DGS scene training.

Fig. 3. Qualitative comparison with state-of-the-art text-to-3D scene generation methods. The top/upper walls are marked with arrows for your reference, if this information is available within the generations. *SceneTeller* is able to generate high-quality scenes, with superior geometric fidelity and 3D consistency.

User Study. We conduct a user study with 30 participants to assess the performance of our method in generating high-quality text-consistent realistic 3D scenes. Human evaluators are presented with 10 scenes, each accompanied by a textual prompt describing its layout and style. For each scene, they are requested to rate it on a scale from 1 to 5 (5 being the best) based on four criteria: (a) Realism, (b) Text Alignment, (c) Geometric Fidelity and (d) Compositional Plausibility.

We present the results of the study in Table 2. Our method consistently outperforms the existing state-of-the-art 3D scene generation methods across all the four criteria by a significant margin, showcasing the merits of *SceneTeller* in generating realistic high-quality 3D scenes solely from text prompts. For additional details of the study and extensive analysis of the results, we refer to the supplementary material.

Table 2. User study results. Study conducted with 30 participants to assess the performance of state-of-the-art methods in generating 3D scenes based on four criteria with a scale from 1 to 5 (5 being the best and vice versa). Our method consistently outperforms the existing state-of-the-art 3D scene generation methods across all the four criteria by a significant margin, showcasing the merits of *SceneTeller* in generating realistic high-quality 3D scenes solely from text prompts.

Method	Input Type	User Study			
		Realism	Text Alignment	Geometric Fidelity	Compositional Plausibility
GSGEN [9]	T	1.313	1.375	1.439	1.392
LucidDreamer [10]	T	3.543	2.751	3.050	3.116
Set-the-scene [11]	L + T	1.963	1.900	2.112	2.181
Ours	T	**4.154**	**4.189**	**4.479**	**4.323**

Quantitative Evaluation of Stylization. To evaluate the consistency of the generated scene style with textual descriptions, we utilize the CLIP Score [21] as the metric. Given that the CLIP score cannot reason about the parts of the prompts describing the layout, we only provide style information to CLIP in the format: *'A Barbie style bedroom'* or *'A bedroom with Barbie style nightstands'* for all methods. As shown in Table 3, our method consistently generates scenes adhering to the given style description.

Qualitative Evaluation. Visual comparison of *SceneTeller* with state-of-the-art text-to-3D methods are provided in Fig. 3. *SceneTeller* yields high-quality 3D scene generations, surpassing the baselines in terms of text-alignment and 3D consistency by predicting an initial 3D layout conditioned on the textual descriptions. Unlike some of the state-of-the-art baselines producing unrealistic appearance (*e.g.* comic or cartoon-like characteristics), *SceneTeller* is able to produce more realistic object shapes and appearances.

Table 3. Quantitative comparison of stylization. We report the CLIP Score to evaluate the consistency of the generated scene style with textual descriptions. Our method consistently generates scenes adhering to the given style description.

	GSGEN [9]	LucidDreamer [10]	Set-the-scene [11]	Ours
CLIP Score ↑	26.478	30.715	27.188	**30.923**

Realism of the Generated Scenes. We evaluate all methods using FID score with a real indoor dataset ScanNet [12] on 10 generated scenes by rendering 50 images from each of them. As shown in Table 4, our method consistently generates more realistic scenes.

Table 4. FID score with the ScanNet dataset [12].

	GSGEN [9]	LucidDreamer [10]	Set-the-scene [11]	Ours
FID ↓	297.10	238.75	297.57	**167.27**

Applicability to Real-World. While current design tools enable professionals to create floor plans, visualize 3D models, and render photo-realistic images, they are not suitable for novice users unfamiliar with such software tools. By enabling users to interact solely through plain natural language prompts, our approach also empowers novice users to flexibly design their own rooms.

4.3 Ablation Study

Effect of Design Choices for 3D Layout Generation. To assess the effectiveness of individual components in our design, we conduct an ablation study. Specifically, we examine two aspects: 1) the prompt source for in-context exemplars, and 2) the technique for selecting in-context exemplars. Regarding the **prompt source**, we present exemplars where the textual description is generated either solely by the rule-based formulation (R) or by a combination of the rule-based formulation and GPT paraphrasing (R+GPT) to introduce linguistic variation as described in Sect. 3.1. For the **exemplar selection** technique, we experiment with a *random* and a *retrieval*-based exemplar selection following [15], $k = 8$ exemplars. Additionally, we experiment with a third set of exemplars, denoted as *pos+neg*, where 4 exemplars are selected by retrieval, and the other 4 are chosen from negative exemplars involving out-of-bound (OOB) or overlapping objects. The negative ones are from the filtered samples during the pre-processing of the dataset, and the LLM is specifically instructed to avoid them. Table 5 verifies that providing in-context exemplars with linguistic variation (R+GPT) selected with a *retrieval*-based approach performs the best, especially in terms of OOB rate by constraining the in-context learning of LLM to similar layouts. Using the *pos+neg* set also considerably improves the performance in terms of mIoU.

Table 5. Effect of design choices for 3D layout generation. We report the 3D layout generation performance w.r.t different design choices such as the prompt source and the exemplar selection. Providing in-context exemplars to LLM with linguistic variation (R+GPT) selected with a *retrieval*-based approach, performs the best.

	Prompt source	Exemplar Selection				Numerical Reasoning			Bbox
		random	retrieval	pos+neg	OOB(%) ↓	rec.↑	prec.↑	acc.↑	mIoU ↑
1	R	✓			74.12	100.00	100.00	100.00	31.10
2	R		✓		64.87	100.00	100.00	100.00	**45.50**
3	R			✓	74.10	100.00	100.00	100.00	41.90
4	R + GPT	✓			76.11	99.10	99.10	99.10	29.90
5	R + GPT		✓		**54.65**	100.00	100.00	100.00	44.20
6	R + GPT			✓	66.45	100.00	100.00	100.00	43.90

5 Conclusion

In this paper, we presented a novel pipeline for 3D room generation from a textual description. Our approach supports style editing either on the individual object or on the entire scene level. Rooms generated by our method were preferred by human evaluators over the baseline results almost in all cases, confirming its efficacy on this challenging task. Importantly, our approach can be used by novices, thus making a step towards democratizing 3D scene design.

Acknowledgements. We thank members of the Bosch-UvA Delta Lab and anonymous reviewers for helpful discussions and feedback. This project was generously supported by the Bosch Center for Artificial Intelligence.

References

1. Achiam, J., et al.: GPT-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
2. Alayrac, J.B., et al.: Flamingo: a visual language model for few-shot learning. In: Advances in Neural Information Processing Systems (NeurIPS), vol. 35, pp. 23716–23736 (2022)
3. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: MIP-NeRF: a multiscale representation for anti-aliasing neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR), pp. 5855–5864 (2021)
4. Brooks, T., Holynski, A., Efros, A.A.: Instructpix2pix: learning to follow image editing instructions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 18392–18402 (2023)
5. Chen, A., Xu, Z., Geiger, A., Yu, J., Su, H.: Tensorf: tensorial radiance fields. In: European Conference on Computer Vision (ECCV) (2022)
6. Chen, R., Chen, Y., Jiao, N., Jia, K.: Fantasia3D: disentangling geometry and appearance for high-quality text-to-3D content creation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2023)

7. Chen, X., et al.: Hallucinated neural radiance fields in the wild. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12943–12952 (2022)
8. Chen, Y., et al.: Gaussianeditor: swift and controllable 3D editing with gaussian splatting. arXiv preprint arXiv:2311.14521 (2023)
9. Chen, Z., Wang, F., Liu, H.: Text-to-3D using Gaussian splatting. arXiv preprint arXiv:2309.16585 (2023)
10. Chung, J., Lee, S., Nam, H., Lee, J., Lee, K.M.: Luciddreamer: domain-free generation of 3D Gaussian splatting scenes. arXiv preprint arXiv:2311.13384 (2023)
11. Cohen-Bar, D., Richardson, E., Metzer, G., Giryes, R., Cohen-Or, D.: Set-the-scene: global-local training for generating controllable nerf scenes. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops, pp. 2920–2929 (2023)
12. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: richly-annotated 3D reconstructions of indoor scenes. In: Proceedings of the Computer Vision and Pattern Recognition (CVPR). IEEE (2017)
13. Fang, J., Wang, J., Zhang, X., Xie, L., Tian, Q.: Gaussianeditor: editing 3D Gaussians delicately with text instructions. arXiv preprint arXiv:2311.16037 (2023)
14. Fang, J., Wang, J., Zhang, X., Xie, L., Tian, Q.: Gaussianeditor: editing 3D Gaussians delicately with text instructions. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2024)
15. Feng, W., et al.: LayoutGPT: compositional visual planning and generation with large language models. In: Advances in Neural Information Processing Systems (NeurIPS), vol. 36 (2024)
16. Fridman, R., Abecasis, A., Kasten, Y., Dekel, T.: Scenescape: text-driven consistent scene generation. In: Advances in Neural Information Processing Systems (NeurIPS), vol. 36 (2024)
17. Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: radiance fields without neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5501–5510 (2022)
18. Fu, H., et al.: 3D-front: 3D furnished rooms with layouts and semantics. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 10933–10942 (2021)
19. Fu, H., Jia, R., Gao, L., Gong, M., Zhao, B., Maybank, S., Tao, D.: 3D-future: 3D furniture shape with texture. Int. J. Comput. Vis. **129**, 3313–3337 (2021)
20. Haque, A., Tancik, M., Efros, A., Holynski, A., Kanazawa, A.: Instruct-nerf2nerf: editing 3D scenes with instructions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR) (2023)
21. Hessel, J., Holtzman, A., Forbes, M., Le Bras, R., Choi, Y.: CLIPScore: a reference-free evaluation metric for image captioning. In: Moens, M.F., Huang, X., Specia, L., Yih, S.W.t. (eds.) Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp. 7514–7528. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic (2021). https://doi.org/10.18653/v1/2021.emnlp-main.595. https://aclanthology.org/2021.emnlp-main.595
22. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: Advances in Neural Information Processing Systems, vol. 33, pp. 6840–6851 (2020)
23. Höllein, L., Cao, A., Owens, A., Johnson, J., Nießner, M.: Text2room: extracting textured 3D meshes from 2D text-to-image models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 7909–7920 (2023)

24. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3D Gaussian splatting for real-time radiance field rendering. ACM Trans. Graph. **42**(4) (2023). https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/
25. Li, Q., Li, F., Guo, J., Guo, Y.: UHDNeRF: ultra-high-definition neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR), pp. 23097–23108 (2023)
26. Li, X., Wang, H., Tseng, K.K.: Gaussiandiffusion: 3D Gaussian splatting for denoising diffusion probabilistic models with structured noise. arXiv preprint arXiv:2311.11221 (2023)
27. Lin, C.H., et al.: Magic3D: high-resolution text-to-3D content creation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 300–309 (2023)
28. Lin, Y., et al.: Componerf: text-guided multi-object compositional nerf with editable 3D scene layout. arXiv preprint arXiv:2303.13843 (2023)
29. Lorraine, J., et al.: Att3d: amortized text-to-3D object synthesis. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 17946–17956 (2023)
30. Metzer, G., Richardson, E., Patashnik, O., Giryes, R., Cohen-Or, D.: Latent-nerf for shape-guided generation of 3D shapes and textures. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12663–12673 (2023)
31. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: representing scenes as neural radiance fields for view synthesis. In: Proceedings of the European Conference on Computer Vision (ECCV) (2020)
32. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Trans. Graph. (ToG) **41**(4), 1–15 (2022)
33. Paschalidou, D., Kar, A., Shugrina, M., Kreis, K., Geiger, A., Fidler, S.: ATISS: autoregressive transformers for indoor scene synthesis. In: Advances in Neural Information Processing Systems, vol. 34, pp. 12013–12026 (2021)
34. Phung, Q., Ge, S., Huang, J.B.: Grounded text-to-image synthesis with attention refocusing. arXiv preprint arXiv:2306.05427 (2023)
35. Po, R., Wetzstein, G.: Compositional 3D scene generation using locally conditioned diffusion. arXiv abs/2303.12218 (2023). https://api.semanticscholar.org/CorpusID:257663283
36. Poole, B., Jain, A., Barron, J.T., Mildenhall, B.: Dreamfusion: text-to-3D using 2D diffusion. In: The Eleventh International Conference on Learning Representations (ICLR) (2023). https://openreview.net/forum?id=FjNys5c7VyY
37. Qian, G., et al.: Magic123: one image to high-quality 3d object generation using both 2D and 3D diffusion priors. In: The Twelfth International Conference on Learning Representations (ICLR) (2024). https://openreview.net/forum?id=0jHkUDyEO9
38. Raj, A., et al.: Dreambooth3D: subject-driven text-to-3D generation. In: The International Conference on Computer Vision (ICCV) (2023)
39. Reiser, C., et al.: MERF: memory-efficient radiance fields for real-time view synthesis in unbounded scenes. ACM Trans. Graph. (TOG) **42**(4), 1–12 (2023)
40. Ren, J., et al.: Make-a-character: high quality text-to-3D character generation within minutes. arXiv preprint arXiv:2312.15430 (2023)
41. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10684–10695 (2022)

42. Sanh, V., et al.: Multitask prompted training enables zero-shot task generalization. In: International Conference on Learning Representations (2022). https://openreview.net/forum?id=9Vrb9D0WI4
43. Sun, C., Sun, M., Chen, H.T.: Direct voxel grid optimization: super-fast convergence for radiance fields reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5459–5469 (2022)
44. Sun, C., Han, J., Deng, W., Wang, X., Qin, Z., Gould, S.: 3D-GPT: procedural 3D modeling with large language models. arXiv preprint arXiv:2310.12945 (2023)
45. Tancik, M., et al.: Block-nerf: scalable large scene neural view synthesis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8248–8258 (2022)
46. Tancik, M., et al.: Nerfstudio: a modular framework for neural radiance field development. In: ACM SIGGRAPH 2023 Conference Proceedings. SIGGRAPH 2023 (2023)
47. Tang, J., Ren, J., Zhou, H., Liu, Z., Zeng, G.: Dreamgaussian: generative gaussian splatting for efficient 3D content creation. arXiv preprint arXiv:2309.16653 (2023)
48. Vachha, C., Haque, A.: Instruct-gs2gs: editing 3D Gaussian splats with instructions (2024). https://instruct-gs2gs.github.io/
49. Wang, H., Du, X., Li, J., Yeh, R.A., Shakhnarovich, G.: Score Jacobian chaining: lifting pretrained 2D diffusion models for 3D generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 12619–12629 (2023)
50. Wang, Z., et al.: Prolificdreamer: high-fidelity and diverse text-to-3D generation with variational score distillation. In: Advances in Neural Information Processing Systems (NeurIPS) (2023)
51. Wei, J., et al.: Finetuned language models are zero-shot learners. In: International Conference on Learning Representations (2022). https://openreview.net/forum?id=gEZrGCozdqR
52. Xu, Y., et al.: Discoscene: spatially disentangled generative radiance fields for controllable 3D-aware scene synthesis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4402–4412 (2023)
53. Yang, Z., et al.: An empirical study of GPT-3 for few-shot knowledge-based VQA. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), vol. 36, pp. 3081–3089 (2022)
54. Yi, T., et al.: Gaussiandreamer: fast generation from text to 3d gaussian splatting with point cloud priors. arXiv preprint arXiv:2310.08529 (2023)
55. Yu, A., Li, R., Tancik, M., Li, H., Ng, R., Kanazawa, A.: Plenoctrees for real-time rendering of neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR), pp. 5752–5761 (2021)
56. Zhou, X., et al.: Gala3D: towards text-to-3D complex scene generation via layout-guided generative Gaussian splatting. arXiv preprint arXiv:2402.07207 (2024)
57. Zhu, J., Zhuang, P., Koyejo, S.: HIFA: high-fidelity text-to-3D generation with advanced diffusion guidance. In: The Twelfth International Conference on Learning Representations (ICLR) (2024). https://openreview.net/forum?id=IZMPWmcS3H