LandCraft: Designing the Structured 3D Landscapes via Text Guidance

Zhihao Liu[1,2*], **Fang Liu**[1*], **Weihao Xuan**[1,2], **Naoto Yokoya**[1,2†]

[1]The University of Tokyo
[2]RIKEN Center for Advanced Intelligence Project (AIP)
liuzh96@outlook.com, fangliu2896@gmail.com, xuan@ms.k.u-tokyo.ac.jp, yokoya@k.u-tokyo.ac.jp

Abstract

Modeling large-scale landscapes is a foundational yet time-consuming task in many 3D applications, typically requiring substantial expertise. Recently, Text-to-3D techniques have emerged as a promising, beginner-friendly prototyping approach for generating 3D content from textual input. However, existing methods either produce unusable, problematic geometries, or fail to fully capture the user's complex intent from the input text—making it difficult to generate high-quality landscape assets with controllable spatial and geographic features. In this paper, we present *LandCraft*, a novel AI-assisted authoring tool that enables the rapid creation of high-quality landscape scenes based on user descriptions. Our system employs a coarse-to-fine generation process: Initially, large language and deep generative models concretize textual ideas into abstract representations that capture essential landscape features, such as spatial and geographic characteristics. Then, we leverage a comprehensive procedural generation module to synthesize the detailed, structurally consistent 3D landscapes based on these inferred representations. Our *LandCraft* can effectively generate production-ready 3D scene assets that can be seamlessly exported to external game engines or modeling software, enabling immediate practical use.

Introduction

Landscapes are the foundation of our natural world. Their high-quality 3D models can greatly enhance the realism of a wide range of applications such as games, movies, and virtual reality. The conventional approach to 3D landscape creation relies heavily on manual modeling within commercial software platforms (e.g., *3ds Max*). This process is not only time-consuming and labor-intensive, but also requires professional expertise, posing a significant barrier for newcomers. Therefore, there is a growing need for efficient methods to create 3D landscape scenes.

With advances in generative AI, Text-based generation has emerged as a promising design paradigm, enabling even beginners to rapidly create various digital content such as images (Rombach et al. 2022; Podell et al. 2024). Building on this progress, recent research has extended text-based generation into the 3D domain. Methods like DreamFusion (Poole et al. 2023) and its successors (Sun et al. 2024b; Wang et al. 2024; Lin et al. 2023) achieve Text-to-3D generation by leveraging diffusion models to produce 3D content in implicit forms, such as neural radiance fields. However, these methods face several critical limitations: they primarily focus on small, isolated objects and tend to generate outputs with unstructured geometry and noise. Especially when dealing with complex scenes such as landscapes, the results are often low-resolution and topologically problematic meshes. Thus, generating high-quality, truly-usable 3D landscapes from text prompts is beyond their capabilities.

To tackle this problem, recent research has explored combining AI with procedural modeling techniques to produce more realistic 3D scenes. Procedural modeling (Smelik et al. 2014) is an important concept in computer graphics (CG) that enables the algorithmic generation of production-ready 3D models through predefined rules These rules programmatically simulate structural patterns found in nature, allowing for the automated creation of diverse 3D scenes such as forests (Niese et al. 2022) and terrains (Guérin et al. 2016). To enable text-driven control, recent 3D-GPT (Sun et al. 2024a) and its concurrent works (Hu et al. 2024; Zhou et al. 2025) have made preliminary attempts to leverage large language models (LLMs) to translate text into simple Blender commands for 3D scene generation. However, these methods only generate beginner-level scenes where objects are just randomly placed. And users are not allowed to control the spatial layouts or geographic features, leading to their inability of creating large-scale landscapes with controllable complex structures. Secondly, these methods are all built on Blender, a professional 3D software that requires expert knowledge to use, making it inaccessible for most ordinary users.

In this paper, we present *LandCraft*, a novel framework that enables even novice users to easily generate high-quality, personalized 3D landscapes from text descriptions. Our method employs a progressive synthesis scheme: (1) Given a text description, we first leverage a large language model (LLM) to generate an initial concretization of the target landscape. Specifically, the LLM functions a analytic planner to extract a set of coarse-grained features, including the approximate spatial locations of landscape elements and a collection of attributes that define both global and lo-

[*]These authors contributed equally.
[†]Corresponding author.

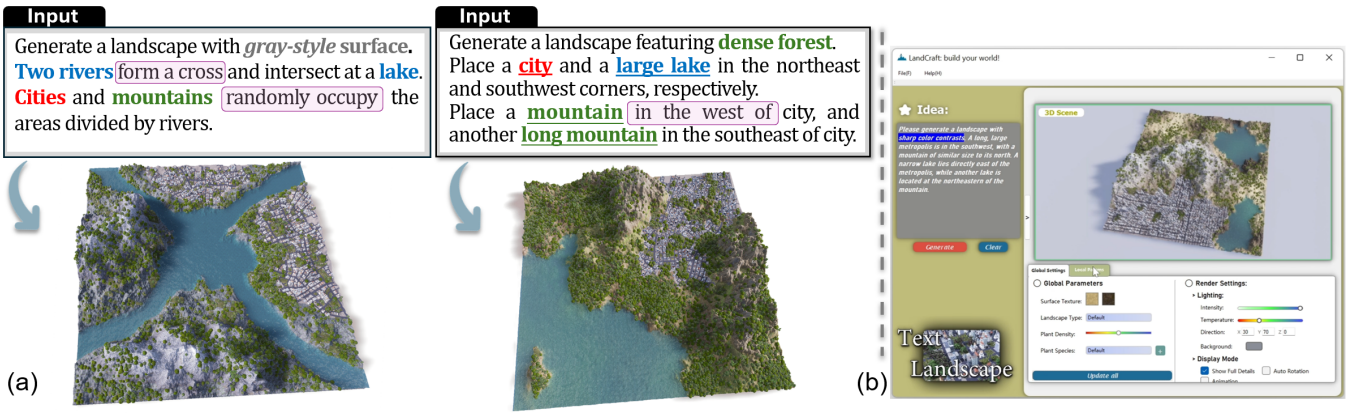


Figure 1: (a) We present LandCraft, a novel text-based generation framework that enables users to easily create detailed, large-scale 3D landscapes with correct spatial and geographic features. (b) The user interface of our LandCraft.

cal appearance settings. (2) Next, we employ a diffusion-based conditional map generator (MapGen) to refine this initial concretization into more detailed maps. These maps define heightfields and semantic layouts with fine-grained precision. (3) Finally, conditioned on the inferred maps and attributes, we leverage a comprehensive compositional 3D generation module to synthesize the final 3D landscape scenes. This module incorporates a suite of well-integrated, parametric procedural generators, and is capable of effectively producing realistic 3D assets for various terrain components, such as trees, grass, cities, and other natural or man-made elements. We specifically develop our procedural generators by C# and C++, rather than relying on inefficient Python wrappers of third-party tools like Blender, thereby achieving significantly higher geometry modeling efficiency. Figure 1 showcases two example modeling results with an average generation time of just 1.8 minutes. To summarize, this work makes the following contributions:

- A novel landscape design system that empowers users to rapidly create realistic, large-scale 3D landscape assets tailored to their textual descriptions. Our system yields exportable and editable 3D models, allowing seamless integration into external game engines for immediate use.
- We present a coarse-to-fine framework that systematically integrates LLM, generative network, and procedural modeling techniques, enabling the creation of structured landscape models with reliable user control over the *spatial layouts* and *appearance details*.
- A light-weight, intuitive software interface (Figure 1(b)) for users to perform the landscape generation simply with a single click. Extensive experiments demonstrate the effectiveness of our method in reasoning and planning for 3D landscape generation.

Related Work

3D Landscape Generation. Previous research on landscape modeling mainly focuses on its random generation, with *procedural modeling* techniques being the predominant approach (Smelik et al. 2014). Procedural modeling is a concept that generates 3D content by programmatically simulating its underlying structural patterns (e.g., self-similarity). Early studies primarily addressed the modeling of geological elevation characteristics of terrain surfaces, such as mountains (Stachniak and Stuerzlinger 2005; Guérin et al. 2016), wrinkles (Cordonnier et al. 2023; Génevaux et al. 2015), and river systems (Derzapf et al. 2011; Peytavie et al. 2019). More recently, with procedural modeling techniques extending to domains like forests (Niese et al. 2022; Palubicki et al. 2009; Liu, Cheng, and Yokoya 2025) and urban areas (Parish and Müller 2001), researchers also began to explore generating more realistic and complex landscape composites (Emilien et al. 2012; Grosbellet et al. 2016; Bulbul 2023; Pałubicki et al. 2022). However, these methods primarily focus on random generation without easy user control. Users must possess sufficient programming skills to produce satisfying results.

Text-based Generative Models. The introduction of latent diffusion model (LDM) (Rombach et al. 2022) has significantly accelerated the development in Text-to-Image (T2I) generation domain. A range of subsequent works were then proposed to improve the image quality (Podell et al. 2024; Saharia et al. 2022) or to address specific user requirements (Li et al. 2025; Ma et al. 2025). Moreover, researchers also explored incorporating multimodal inputs (e.g., images or sounds) as additional guidance in the T2I generation process to further enhance user control (Zhang, Rao, and Agrawala 2023; Sung-Bin et al. 2025).

The above success in the 2D has also influenced the development of 3D generation. Early Text-to-3D approaches are limited to producing low-resolution 3D representations, such as voxels (Chen et al. 2018), and point clouds (Nichol et al. 2022). Recently, DreamFusion (Poole et al. 2023) and its successors (Lin et al. 2023; Wang et al. 2024; Cao et al. 2024) have made substantial progress by leveraging diffusion models to help generate visually appealing 3D objects in the form of neural radiance fields (NeRFs) or Gaussian splatting. However, the outputs of these methods are often unstructured geometries with noticeable floaters and noises, which hinders their practical applicability. Generating high-

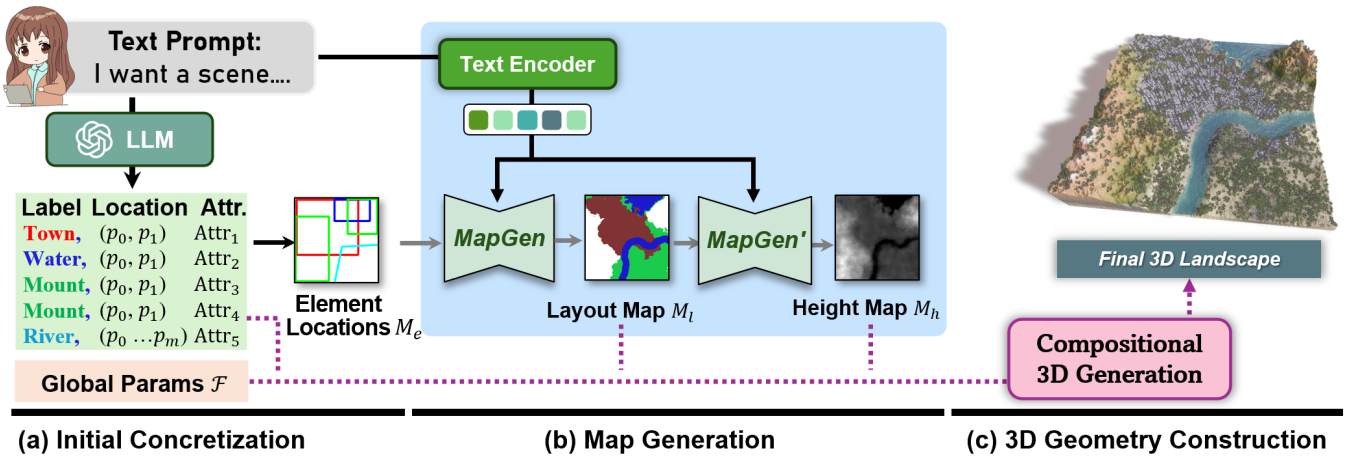


Figure 2: Overall pipeline of our proposed method. (a) Our process begins using an LLM to concretize plausible spatial distributions for landscape elements and appearance parameters. (b) The map generator modules (MapGen) then sequentially generate the detailed layout map M_l and a heightfield map M_h conditioned on the LLM output and the given text. (c) Finally, the 3D landscape models are synthesized using procedural modeling techniques, based on the maps M_l and M_h, as well as the inferred local/global appearance parameters.

quality, large-scale landscapes with complex topologies is beyond their capabilities.

Large Language Models. Recent advancements in large language models (LLMs) have demonstrated impressive capabilities in logical reasoning, significantly reshaping the paradigm of natural language processing (NLP) (Devlin et al. 2019; OpenAI 2023; Dubey et al. 2024) . By leveraging in-context learning (Wei et al. 2022) or various fine-tuning strategies (Hu et al. 2022), LLMs can be effectively adapted to a wide range of specialized downstream applications including robotics (Dalal et al. 2024), vision (Wang et al. 2023), art design (Qu et al. 2023; Liu et al. 2025), and beyond (Wang et al. 2025a). Building on this, several recent works also explored using LLMs to achieve the scene generation (Hu et al. 2024; Sun et al. 2024a; Zhou et al. 2025). These methods try to translate input text into instructions that are executable by commercial 3D software like Blender to construct simple 3D scene. However, these methods are limited to beginner-level operations—such as just placing pre-defined objects in a small area in a random pattern. None of the above methods can generate large 3D landscapes with controllable spatial layouts and geographic structures from the text descriptions.

Methodology: From Text to 3D Landscape

In this paper, we propose a coarse-to-fine paradigm for efficiently generating 3D landscape assets from text prompts, with offering robust control over spatial and visual features. Figure 2 illustrates the overall framework of our approach, and we introduce its three main steps in the following subsections:

LLM-based Initial Concretization

Given the input text, our first step is to generate an initial concretization for the entire landscape at a coarse-grained level, outlining plausible spatial arrangements of landscape elements along with the corresponding appearance parameters. Previous studies have demonstrated that LLMs exhibit strong potential in performing question-answering tasks related to layout understanding (Qu et al. 2023; Feng et al. 2024). Inspired by this, we leverage the GPT-4o module (OpenAI 2023) to plan the initial landscape layouts.

As shown in Figure 2(a), given the text prompts, the LLM extracts a group of landscape elements $\{\mathbf{E}_i | i \in \mathbb{N}\}$, and each element is described as the following triplet:

$$\mathbf{E}_i = (c_i, \boldsymbol{P}_i, \mathbf{Attr}_i). \tag{1}$$

Here, c_i denotes the element's category, $\boldsymbol{P}_i$ represents the spatial coordinates defining the location of $\mathbf{E}_i$, and $\mathbf{Attr}_i$ is the local appearance attributes. By default, the location $\boldsymbol{P}_i$ is represented as a 2D bounding box when the category c_i is *"town"*, *"mountain"*, or *"water body"*. In these cases, $\boldsymbol{P}_i = \{\boldsymbol{P}_i^0, \boldsymbol{P}_i^1\}$ represents the (x, y)-coordinates of the top-left and bottom-right vertices for the bounding box area, respectively. For *"river"* category, we use the polyline to describe its flow path, where $\boldsymbol{P}_i = \{\boldsymbol{P}_i^k\} (2 \leq k \leq 5)$ indicates the key sequential vertices along the river. In our settings, a river can have at most 5 vertices.

Each region $\mathbf{E}_i$ is also associated with a few local appearance attributes $\mathbf{Attr}_i$ (e.g., the residential density in a town area), which can be seamlessly used by the subsequent procedural modeling algorithms to generate 3D landscape scenes that align with the user-specified features. Beyond element-level arrangement, LLM also automatically extracts a set of global parameters $\boldsymbol{\mathcal{F}}$ that describes the features of the entire landscape, such as the base surface tint and vegetation types. We adapt the LLM to our task using in-context learning (ICL) (Brown 2020). The full ICL instructions and the global/local parameter list are summarized in the supplementary materials.

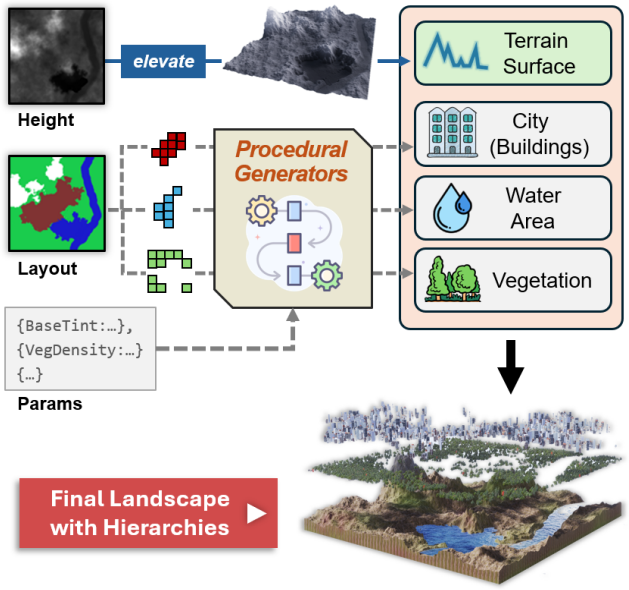


Figure 3: The procedural modeling module contains a series of modular sub-algorithms to synthesize details for final 3D landscape geometries with separable hierarchies.

Fine-Grained Map Generator

Given the intial concretization, we then synthesize more fine-grained landscape maps based on diffusion models. As shown in Figure 2(b), we first plot the numerical locations of all elements onto a 2D image M_e. Next, we train two map generators (MapGen) to sequentially produce a layout map M_l and a height map M_h for the entire terrain.

Simply put, this step has two image translation processes, both taking multimodal inputs (i.e., text prompt $\boldsymbol{\tau}$ and an image M) and producing a new image M' in a different domain. However, prior LDMs like ControlNet (Zhang, Rao, and Agrawala 2023) often struggle with multi-object control, which leads to redundant/missing elements in the resulting maps (please see supplementary material). Therefore, a more reliable map generator is needed for this step.

Architecture. In brief, our MapGen network adopts a chain of Diffusion Transformer (DiT) (Peebles and Xie 2023) as the base architecture. We use a CLIP text encoder to extract the text embeddings, which are then injected into the DiT blocks via multi-head cross-attention (Chen et al. 2024).

Diffusion Loss. One key of the network is the use of a Brownian Bridge diffusion model (BBDM) (Li et al. 2023). Unlike prior multimodal LDMs (Zhang, Rao, and Agrawala 2023) that take a random noise as initial state, the BBDM defines the T-step denoising process as a stochastic Brownian bridge, where the source and target images are directly treated as the initial and end states of diffusion process. Therefore, the intermediate state $\boldsymbol{x}_t$ at time $t \in [0, T]$ can be formulated as an interpolation between two image domains:

$$\boldsymbol{x}_t = (1 - \frac{t}{T})M + \frac{t}{T}M' + \boldsymbol{\gamma}, \tag{2}$$

where $\boldsymbol{\gamma} \sim \mathcal{N}(0, \sigma_t \mathbf{I})$ is a small noise term. This formulation ensures that all intermediate states $\boldsymbol{x}_t$ are explicitly constrained by both source and target domains, leading to improved spatial and semantic consistency throughout the generation process. Thus, the final Brownian bridge diffusion loss can be written as:

$$\mathcal{L}_{\text{bbdm}} = \|\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, \boldsymbol{\tau}) - \boldsymbol{\epsilon}_t\|^2, \boldsymbol{\epsilon}_t = \tfrac{t}{T}(M' - M) + \gamma \tag{3}$$

Here, the network is trained to predict $\boldsymbol{\epsilon}_t$, which is the difference between $\boldsymbol{x}_t$ and initial state M.

Apart from diffusion loss, we also add extra loss terms according to different map types, respectively.

Layout Consistency Loss. For the first MapGen generating the layout maps M_l, we additionally add a layout consistency loss to further enhance the spatial alignments between the predicted maps and the known element location coordinates. Specifically, the loss is computed as Soft-IoU score over all semantic categories c:

$$\mathcal{L}_{lc} = 1 - \frac{1}{|C|}\sum_c \frac{\sum(\hat{B}_c \cdot B_c)}{\sum(\hat{B}_c + B_c - \hat{B}_c \cdot B_c)} \tag{4}$$

where "$\cdot$" indicates element-wise multiplication. $\hat{B}_c$ is the binary mask for class c obtained from the predicted layout map $\hat{M}_l$, while B_c is another binary mask of class c derived from the known numerical element locations of $\mathbf{E}_i$: For box-type classes (e.g., *town*), the mask value in B_c is set to 1 inside bounding boxes and 0 elsewhere, while for path-type elements (i.e., *river*), we sparsely sample several keypoints along the path and assign them a value of 1 in the mask B_c.

Gradient Loss. Moreover, we also incorporate a gradient loss term for the second MapGen$'$ that generates height maps M_h. This loss penalizes discrepancies between the height gradients of maps, so as to produce more geometrically consistent elevation. The gradient loss is defined as:

$$\mathcal{L}_{\text{grad}} = \|\nabla_x \hat{M}_h - \nabla_x M_h\|_1 + \|\nabla_y \hat{M}_h - \nabla_y M_h\|_1, \tag{5}$$

where $\hat{M}_h$ and M_h denote the predicted and ground truth elevation maps, and ∇_x, ∇_y represent the horizontal and vertical gradient operators, respectively.

As a result, the complete loss functions used to train the two map generator modules can be defined as: $L_{M_l} = L_{bbdm} + \lambda_1 \mathcal{L}_{lc}$ and $L_{M_h} = L_{bbdm} + \lambda_2 \mathcal{L}_{grad}$, respectively. where λ_1 and λ_2 are hyperparameters.

Compositional 3D Landscape Constructions

After applying the above AI modules, we have obtained a series of intermediate representations that align with the user inputs, i.e., two fine-grained maps (M_l and M_h) and a set of global/local appearance parameters.

To instantiate these abstract representations, our final step employs a combination of parametric procedural modeling algorithms to synthesize realistic 3D landscape models with intricate geometric details automatically. As shown in Figure 3, we begin by constructing 3D triangular meshes for the bare terrain surface based on the height map M_h. Then, for each element area in the layout map M_l, we implemented specific algorithmic modules to generate the corresponding 3D details. The following briefly summarizes the techniques used for different landscape elements:

Towns. A rule-based road-map algorithm (Parish and Müller 2001) is utilized to generate the street networks and subdivide the city area into a series of smaller blocks. Subsequently, each block will have a certain probability to evolve into a *building* or *urban green space*. For the *building* blocks, we further generate 3D building models within the block boundaries using a component-based building modeling method (Schwarz and Müller 2015).

Vegetation. The 3D tree models with various species are botanically generated based on a self-organization modeling algorithm (Palubicki et al. 2009), and then are planted in wild areas and urban green spaces using a probabilistic placement strategy (Niese et al. 2022).

Water Areas. For *water* areas and *rivers*, we simulate real-time water surface animation in the GPU shader using the Trochoidal Gerstner wave algorithm (Tessendorf et al. 2001).

We implemented the above procedural generation modules based on C++. As shown in Figure 3, this modular generation workflow can make each landscape element to be independent and separable, resulting in 3D scenes with fine-grained hierarchical structures. This feature also enables our results to be directly used in external game engines or commercial modeling software.

Dataset Synthesis. The above procedural modeling is utilized not only in the design stage but also for automatically synthesizing the dataset to train the neural networks. The core of dataset generation is to obtain the inputs and outputs for MapGen, including the text prompts, element triplets $\mathbf{E}_i$, as well as two maps M_l and M_h.

Specifically, we first synthesize the semantic layout maps M_l based on a combination of random walk and cellular automata algorithms (Macedo and Chaimowicz 2017). Then, the terrain heightfields M_h are simulated based on a gradient field-based method (Guérin et al. 2022) for the mountain area marked in M_l, and perlin-noise (Perlin 1985) for the outside regions. The element triplet $\mathbf{E}_i = (l_i, \boldsymbol{\mathcal{P}}_i, \text{Attr}_i)$ and appearance attributes can be jointly collected during the generation of map M_l. Finally, to obtain the input text, we employ GPT-4o to construct the descriptions according to the extracted element triplets $\mathbf{E}_i$ and appearance attributes. We also manually proofread the text descriptions to prevent potential mislabeling. As a result, we synthesize 20K samples to train the map generators.

Experimental Evaluation

In this section, we conducted a series of experiments to validate the effectiveness of *LandCraft* both qualitatively and quantitatively. The software interface and procedural generators were developed in C++&C#, and we deploy the interface on a machine equipped with an Intel Core Ultra 9 CPU. The AI modules were implemented in PyTorch and trained on a Nvidia A100 GPU. For quantitative evaluation, we additionally synthesized 2k landscape samples as the test set.

Results

Modeling Results. To demonstrate the robustness of the proposed method, Figure 5 first presents several modeling results generated from challenging input scenarios. We test the prompts containing vague descriptions, stylized shapes, and complex multi-object spatial relationships, respectively. The results show that our output can closely align with the given requirements, demonstrating the system's high flexibility in understanding diverse landscape descriptions. Moreover, text-based design is also widely recognized for providing users with high flexibility in open-ended exploration of results. Figure 4 illustrates an example: given a textual input, our system can generate various yet semantically consistent outputs. Users can select their preferred version, thereby facilitating their ideation stage.

Figure 7 further provides a gallery of more generated landscapes, demonstrating the capability of our approach to yield structurally diverse results.

Map types	Loss setting	LPIPS ↓	SSIM ↑
Layout M_l	$\mathcal{L}_{bbdm}$	0.425	0.741
	$\mathcal{L}_{bbdm}$+$\mathcal{L}_{lc}$ (Full)	**0.398**	**0.786**
Height M_h	L_{bbdm}	0.497	0.726
	$\mathcal{L}_{bbdm}$+$\mathcal{L}_{grad}$ (Full)	**0.464**	**0.752**

Table 1: Ablation study of using different loss combinations for MapGen to generate two map types, respectively.

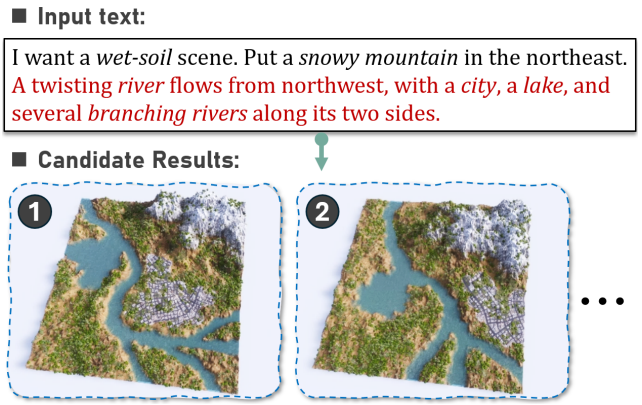
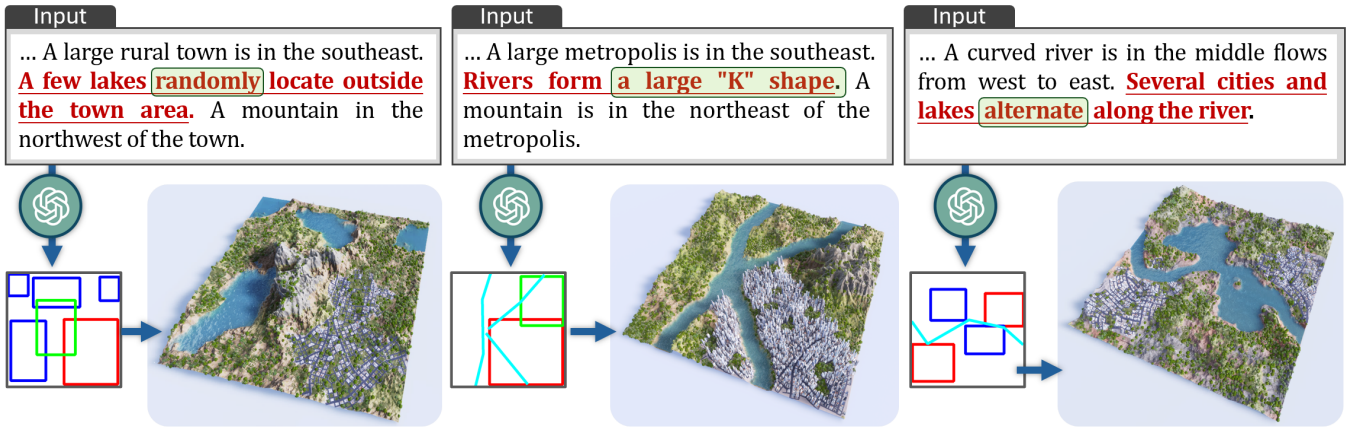


Figure 4: The use of LLM can yield various possible concretizations that all align with user inputs. This feature facilitates the open-ended exploration of results by users.

Network Performance. To quantitatively evaluate our approach, we also designed a series of ablation studies. Here, we first focus on the design of the map generator (MapGen) module. In Table 1, we quantitatively analyze the effectiveness of different loss function combinations for generating two types of maps. We adopt two evaluation metrics: LPIPS (Zhang et al. 2018) and SSIM (Wang et al. 2004), both of which are well-suited for assessing localized structural similarity and perceptual quality in images. The results in Table 1 demonstrate that incorporating the loss terms $\mathcal{L}_{lc}$ and $\mathcal{L}_{grad}$ can effectively enhances the performance of the base diffusion model for the two map generation tasks. Due to page limitations, more quantitative and qualitative analyses are included in the supplementary materials.



(a) Vague descriptions. (b) Stylized shapes. (c) Complex relative locations among multiple objects.

Figure 5: Robustness of layout induction in handling challenging text descriptions: (a) vague descriptions, (b) stylized shapes, and (c) multi-object spatial relations, demonstrating the high adaptability of our method to diverse user requirements.

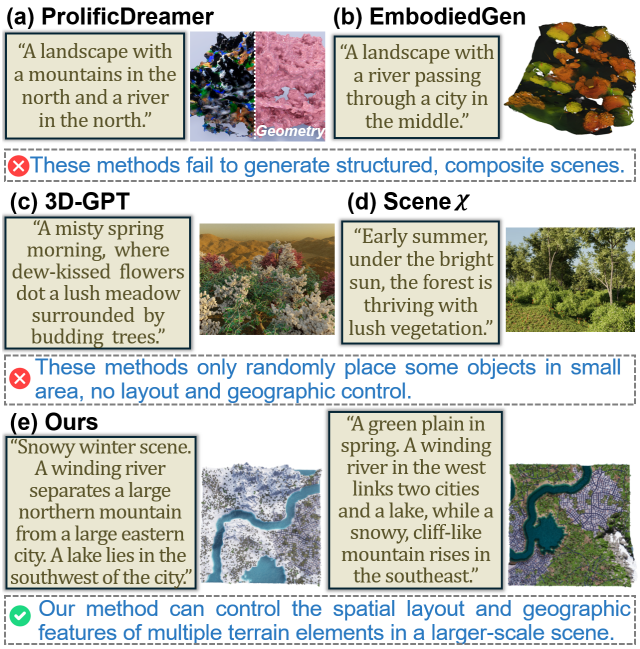


Figure 6: Qualitative comparisons with recent text-3D approaches: (a-b) the generic text-to-3D methods, (c-d) scene-specific methods, (e) our method.

Method	AE↑	CST↑
DreamFussion (Poole et al. 2023)	2.63	2.86
ProlificDreamer (Wang et al. 2024)	2.96	2.75
EmbodiedGen (Wang et al. 2025b)	4.24	3.65
3D-GPT (Sun et al. 2024a)	6.73	4.39
SceneCraft (Hu et al. 2024)	5.46	5.27
Ours	**8.62**	**7.51**

Table 2: Results of a user study comparing our method with recent Text-to-3D approaches. We mainly measures two aspects: aesthetics (AE) and text-scene consistency (CST).

Comparisons

In Figure 6, we further compare our *LandCraft* with a series of recent Text-to-3D approaches, and we categorize the previous works into two groups:

The first group is the generic text-to-3D methods, with ProlificDreamer(Wang et al. 2024) and EmbodiedGen(Wang et al. 2025b) representing recent advances in this area. These methods typically leverage neural networks to generate 3D shapes in implicit forms such as NeRFs. However, these methods are primarily suitable for isolated objects, and only yield "unstructured" geometries. Therefore, when applied to complex scenes, they tend to produce problematic, watertight meshes with low-resolution details (see Figure 6 (a-b)). As a result, these methods are unable to produce practically-usable 3D landscape assets.

The second group is the recent scene-specific methods like 3D-GPT (Sun et al. 2024a). These methods employ LLM to infer Blender commands for assembling 3D assets into 3D scenes. However, limited by Blender's functionality, these methods only randomly place objects without precise control. They cannot interpret users' more complex intentions, such as manipulating spatial layouts or geographic features. Therefore, they only generate homogeneous scenes with simple patterns (see Figure 6(c-d)). Moreover, Blender is a very complicated platform, and users must require a certain level of expertise to correctly operate their methods.

In contrast, by decomposing the generation process into a three-stage coarse-to-fine workflow, our system can effectively accommodate users' complex and diverse requirements, including robust control over spatial layouts, appearance features, etc. This enables the creation of large-scale terrain scenes with rich structural diversity. Meanwhile, our one-stop software interface also greatly facilitates users' interaction and use. Please also refer to the supplementary ma-

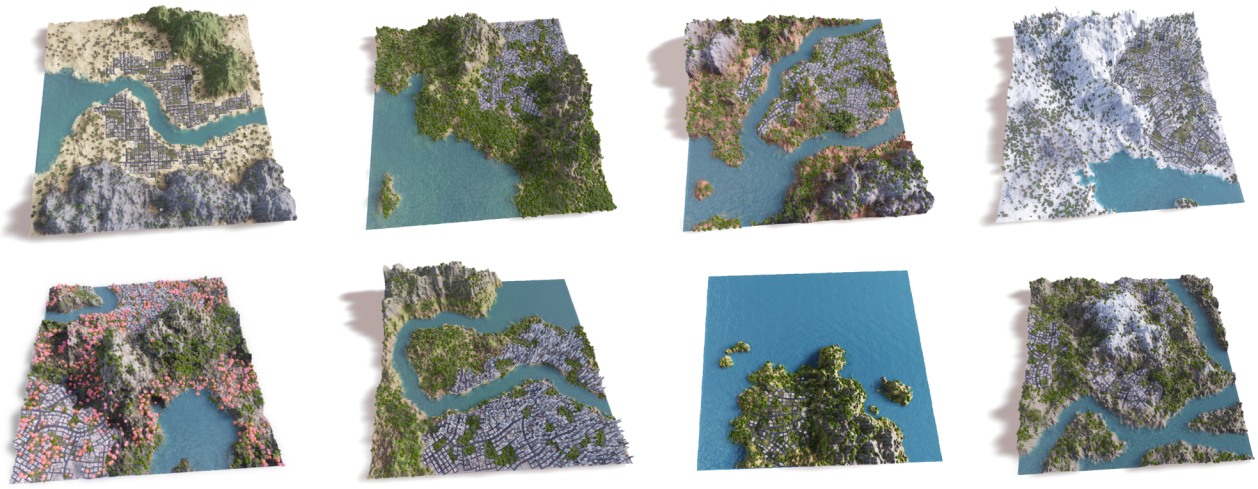


Figure 7: A Gallery of 3D Landscapes that are directly generated by our text-based generation method. Please see the Supplemental Material for the corresponding input text prompts and more results.

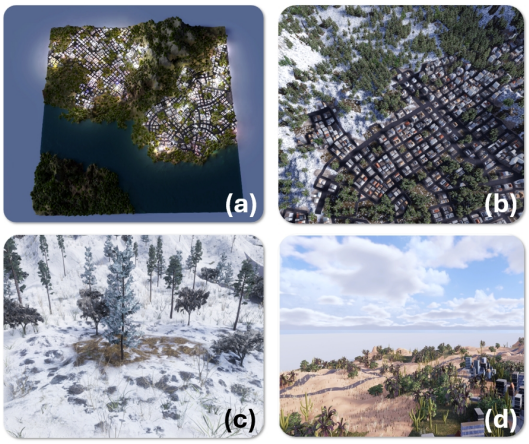


Figure 8: Practical application examples of our results. (a) City-light simulation. (b-d) Export into *Unity* game engine.

terial for a detailed quantitative comparison.

User Study. We also conducted a user study to further evaluate the effectiveness of different Text-to-3D approaches from user's perspective. The study involved 15 participants, including 4 experts in 3D modeling and 11 novice volunteers. Prior to the study, we configured recent methods for participants to use. Noted, since 3D-GPT (Sun et al. 2024a) and SceneCraft (Hu et al. 2024) are not fully open-source so far, we allow participants to assess by reviewing the results presented in their original papers. Each participant was asked to freely create 3D landscapes using the systems, and then rate them based on two criteria: aesthetic quality (AE) and text-scene consistency (CST). All ratings were collected using a 10-point Likert scale (higher scores indicate better performance). Table 2 summarizes the average scores. Overall, our system received relatively better ratings on both metrics, which is consistent with our earlier analyses.

Conclusion

In this paper, we presented *LandCraft*, a novel text-based prototyping system that reliably concretizes users' textual descriptions into high-quality, structured 3D landscape assets. Our system allows even novice users without professional design knowledge to create landscapes with minimal effort. We introduce an AI-CG collaborative pipeline that decouples the task into a coarse-to-fine progressive generation workflow. This workflow ensures robust control over terrain features, while satisfying the requirements of precise 3D geometric modeling. As a result, our approach can produce editable 3D scenes that are directly compatible with industrial 3D software. Fig. 8 showcases several examples of applying our generated models in practical applications. Furthermore, we developed an intuitive user interface that allows interactive refinement of generated results according to users' needs. Extensive experiments demonstrate our modeling system's usability, efficiency, and effectiveness in creating diverse landscape scenes.

Limitations. As a preliminary attempt, our system still has limitations. For instance, our current system fails to make response for the requirements of specific cultural styles (e.g., medieval European or Middle Eastern cities). To solve this problem, we plan to collaborate closely with industry partners in the future to incorporate industrial-grade assets into our procedural modeling modules—such as more facade textures for buildings, thereby enhancing the completeness and visual quality of the generated scenes.

Acknowledgments

We thank the anonymous reviewers for their valuable suggestions. This work is supported in part by the following programs: JST, FOREST under Grant Number JPMJFR206S; JST, NEXUS under Grant Number JPMJNX25CA; RIKEN Junior Research Associate (JRA) Program.

References

Brown, T. B. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Bulbul, A. 2023. Procedural generation of semantically plausible small-scale towns. *Graphical Models*, 126: 101170.

Cao, Y.; Cao, Y.-P.; Han, K.; Shan, Y.; and Wong, K.-Y. K. 2024. Dreamavatar: Text-and-shape guided 3d human avatar generation via diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Chen, J.; Jincheng, Y.; Chongjian, G.; Yao, L.; Xie, E.; Wang, Z.; Kwok, J.; Luo, P.; Lu, H.; and Li, Z. 2024. PixArt-α: Fast Training of Diffusion Transformer for Photorealistic Text-to-Image Synthesis. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Chen, K.; Choy, C. B.; Savva, M.; Chang, A. X.; Funkhouser, T.; and Savarese, S. 2018. Text2shape: Generating shapes from natural language by learning joint embeddings. In *Asian conference on computer vision (ACCV)*, 100–116.

Cordonnier, G.; Jouvet, G.; Peytavie, A.; Braun, J.; Cani, M.-P.; Benes, B.; Galin, E.; Guérin, E.; and Gain, J. 2023. Forming terrains by glacial erosion. *ACM Transactions on Graphics (TOG)*, 42(4): 1–14.

Dalal, M.; Chiruvolu, T.; Chaplot, D.; and Salakhutdinov, R. 2024. Plan-Seq-Learn: Language Model Guided RL for Solving Long Horizon Robotics Tasks. In *International Conference on Learning Representations (ICLR)*.

Derzapf, E.; Ganster, B.; Guthe, M.; and Klein, R. 2011. River networks for instant procedural planets. In *Computer Graphics Forum (CGF)*, volume 30, 2031–2040.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Burstein, J.; Doran, C.; and Solorio, T., eds., *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, 4171–4186.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Emilien, A.; Bernhardt, A.; Peytavie, A.; Cani, M.-P.; and Galin, E. 2012. Procedural generation of villages on arbitrary terrains. *The Visual Computer*.

Feng, W.; Zhu, W.; Fu, T.-j.; Jampani, V.; Akula, A.; He, X.; Basu, S.; Wang, X. E.; and Wang, W. Y. 2024. Layoutgpt: Compositional visual planning and generation with large language models. *Advances in Neural Information Processing Systems (NeurIPS)*, 36.

Génevaux, J.-D.; Galin, E.; Peytavie, A.; Guérin, E.; Briquet, C.; Grosbellet, F.; and Benes, B. 2015. Terrain modelling from feature primitives. In *Computer Graphics Forum (CGF)*, volume 34, 198–210.

Grosbellet, F.; Peytavie, A.; Guérin, É.; Galin, E.; Mérillou, S.; and Benes, B. 2016. Environmental objects for authoring procedural scenes. In *Computer Graphics Forum (CGF)*, 296–308.

Guérin, E.; Digne, J.; Galin, E.; and Peytavie, A. 2016. Sparse representation of terrains for procedural modeling. In *Computer Graphics Forum (CGF)*, volume 35, 177–187.

Guérin, E.; Peytavie, A.; Masnou, S.; Digne, J.; Sauvage, B.; Gain, J.; and Galin, E. 2022. Gradient terrain authoring. In *Computer Graphics Forum (CGF)*, volume 41, 85–95.

Hu, E. J.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W.; et al. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations (ICLR)*.

Hu, Z.; Iscen, A.; Jain, A.; Kipf, T.; Yue, Y.; Ross, D. A.; Schmid, C.; and Fathi, A. 2024. SceneCraft: An LLM Agent for Synthesizing 3D Scenes as Blender Code. In *Forty-first International Conference on Machine Learning (ICML)*.

Li, B.; Xue, K.; Liu, B.; and Lai, Y.-K. 2023. BBDM: Image-to-image translation with brownian bridge diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern Recognition (CVPR)*, 1952–1961.

Li, B.; Zhang, Z.; Nie, X.; Han, C.; Hu, Y.; Qiu, X.; and Guo, T. 2025. Styo: Stylize your face in only one-shot. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 4625–4633.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 300–309.

Liu, M.; Ma, Y.; Yang, Z.; Dan, J.; Yu, Y.; Zhao, Z.; Hu, Z.; Liu, B.; and Fan, C. 2025. Llm4gen: Leveraging semantic representation of llms for text-to-image generation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 5523–5531.

Liu, Z.; Cheng, Z.; and Yokoya, N. 2025. Neural Hierarchical Decomposition for Single Image Plant Modeling. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 733–742.

Ma, J.; Deng, Y.; Chen, C.; Du, N.; Lu, H.; and Yang, Z. 2025. Glyphdraw2: Automatic generation of complex glyph posters with diffusion models and large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 5955–5963.

Macedo, Y. P.; and Chaimowicz, L. 2017. Improving procedural 2D map Generation based on multi-layered cellular automata and Hilbert curves. In *16th Brazilian Symposium on Computer Games and Digital Entertainment*, 116–125. IEEE.

Nichol, A.; Jun, H.; Dhariwal, P.; Mishkin, P.; and Chen, M. 2022. Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*.

Niese, T.; Pirk, S.; Albrecht, M.; Benes, B.; and Deussen, O. 2022. Procedural urban forestry. *ACM Transactions on Graphics (TOG)*, 41(2): 1–18.

OpenAI. 2023. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*.

Palubicki, W.; Horel, K.; Longay, S.; Runions, A.; Lane, B.; Měch, R.; and Prusinkiewicz, P. 2009. Self-organizing tree models for image synthesis. *ACM Transactions On Graphics (TOG)*, 1–10.

Pałubicki, W.; Makowski, M.; Gajda, W.; Hädrich, T.; Michels, D. L.; and Pirk, S. 2022. Ecoclimates: Climate-response modeling of vegetation. *ACM Transactions on Graphics (TOG)*, 41(4): 1–19.

Parish, Y. I.; and Müller, P. 2001. Procedural modeling of cities. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques (SIGGRAPH)*, 301–308.

Peebles, W.; and Xie, S. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision (CVPR)*, 4195–4205.

Perlin, K. 1985. An image synthesizer. *ACM Siggraph Computer Graphics*, 19(3): 287–296.

Peytavie, A.; Dupont, T.; Guérin, E.; Cortial, Y.; Benes, B.; Gain, J.; and Galin, E. 2019. Procedural riverscapes. In *Computer Graphics Forum (CGF)*, 35–46.

Podell, D.; English, Z.; Lacey, K.; Blattmann, A.; Dockhorn, T.; Müller, J.; Penna, J.; and Rombach, R. 2024. SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Poole, B.; Jain, A.; Barron, J. T.; and Mildenhall, B. 2023. Dreamfusion: Text-to-3d using 2d diffusion. *The Eleventh International Conference on Learning Representations (ICLR)*.

Qu, L.; Wu, S.; Fei, H.; Nie, L.; and Chua, T.-S. 2023. Layoutllm-t2i: Eliciting layout guidance from llm for text-to-image generation. In *Proceedings of the 31st ACM International Conference on Multimedia (MM)*, 643–654.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 10684–10695.

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. *Advances in neural information processing systems (NeurIPS)*, 35: 36479–36494.

Schwarz, M.; and Müller, P. 2015. Advanced procedural modeling of architecture. *ACM Transactions on Graphics (TOG)*, 1–12.

Smelik, R. M.; Tutenel, T.; Bidarra, R.; and Benes, B. 2014. A survey on procedural modelling for virtual worlds. In *Computer graphics forum (CGF)*, 31–50.

Stachniak, S.; and Stuerzlinger, W. 2005. An algorithm for automated fractal terrain deformation. *Computer Graphics and Artificial Intelligence*, 1: 64–76.

Sun, C.; Han, J.; Deng, W.; Wang, X.; Qin, Z.; and Gould, S. 2024a. 3D-GPT: Procedural 3D Modeling with Large Language Models. arXiv:2310.12945.

Sun, J.; Zhang, B.; Shao, R.; Wang, L.; Liu, W.; Xie, Z.; and Liu, Y. 2024b. DreamCraft3D: Hierarchical 3D Generation with Bootstrapped Diffusion Prior. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Sung-Bin, K.; Jun-Seong, K.; Ko, J.; Kim, Y.; and Oh, T.-H. 2025. Soundbrush: Sound as a brush for visual scene editing. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 7167–7175.

Tessendorf, J.; et al. 2001. Simulating ocean water. *SIGGRAPH*, 1(2): 5.

Wang, J.; Xuan, W.; Qi, H.; Liu, Z.; Liu, K.; Wu, Y.; Chen, H.; Song, J.; Xia, J.; Zheng, Z.; and Yokoya, N. 2025a. DisasterM3: A Remote Sensing Vision-Language Dataset for Disaster Damage Assessment and Response. In *Proceedings of the Neural Information Processing Systems*.

Wang, W.; Chen, Z.; Chen, X.; Wu, J.; Zhu, X.; Zeng, G.; Luo, P.; Lu, T.; Zhou, J.; Qiao, Y.; et al. 2023. Visionllm: Large language model is also an open-ended decoder for vision-centric tasks. *Advances in Neural Information Processing Systems (NeurIPS)*, 36: 61501–61513.

Wang, X.; Liu, L.; Cao, Y.; Wu, R.; Qin, W.; Wang, D.; Sui, W.; and Su, Z. 2025b. EmbodiedGen: Towards a Generative 3D World Engine for Embodied Intelligence. *arXiv preprint arXiv:2506.10600*.

Wang, Z.; Bovik, A. C.; Sheikh, H. R.; and Simoncelli, E. P. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing (TIP)*, 600–612.

Wang, Z.; Lu, C.; Wang, Y.; Bao, F.; Li, C.; Su, H.; and Zhu, J. 2024. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *Advances in Neural Information Processing Systems (NeurIPS)*, 36.

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. *Advances in neural information processing systems (NeurIPS)*, 35: 24824–24837.

Zhang, L.; Rao, A.; and Agrawala, M. 2023. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 3836–3847.

Zhang, R.; Isola, P.; Efros, A. A.; Shechtman, E.; and Wang, O. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 586–595.

Zhou, M.; Wang, Y.; Hou, J.; Zhang, S.; Li, Y.; Luo, C.; Peng, J.; and Zhang, Z. 2025. SceneX: Procedural Controllable Large-Scale Scene Generation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 10806–10814.