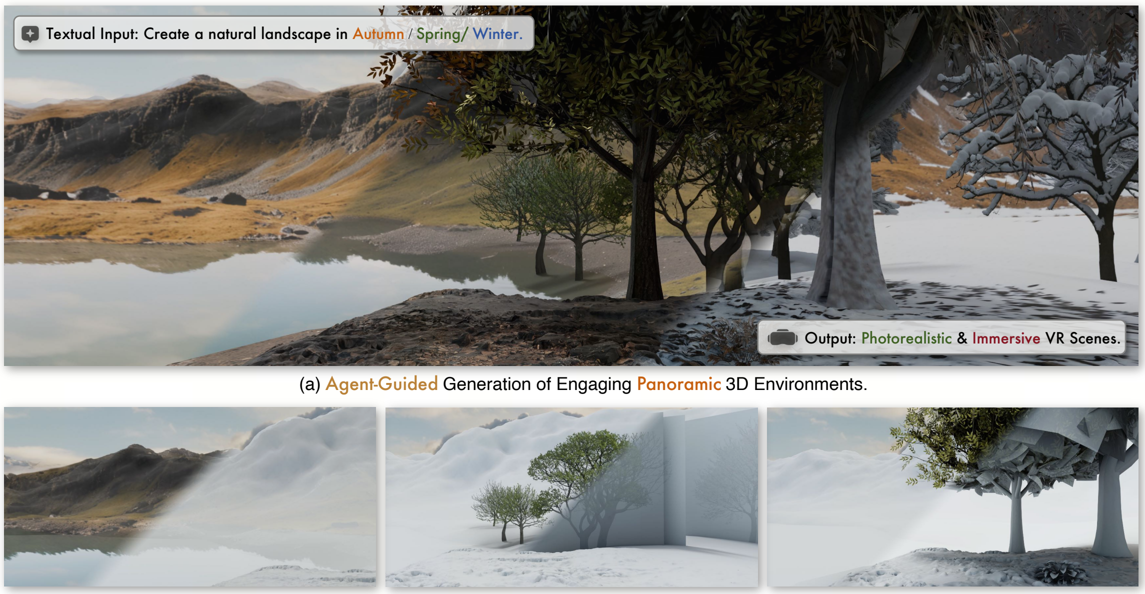
ImmerseGen: Agent-Guided Immersive World Generation with Alpha-Textured Proxies

Jinyan Yuan, Bangbang Yang, Keke Wang, Panwang Pan, Lin Ma, Xuehai Zhang, Xiao Liu, Zhaopeng Cui, and Yuewen Ma


Textual Input: Create a natural landscape in Autumn / Spring/ Winter.
Output: Photorealistic & Immersive VR Scenes.

(a) Agent-Guided Generation of Engaging Panoramic 3D Environments.

Base World: Skybox and Alpha-Textured Terrain — **Midground Scenery:** Plane-based Proxies — **Foreground Scenery:** Template-based Proxies

(b) Generated Compact Scenery in Alpha-Textured Proxies, Supporting Real-Time Rendering on VR Headsets.

Fig. 1: Tailored for immersive VR experiences, ImmerseGen generates photorealistic worlds from text prompts by synthesizing compact alpha-textured proxies through agent-guided design and arrangement, obviating the need for complex assets while ensuring diversity.

Abstract—Automating immersive VR scene creation remains a primary research challenge. Existing methods typically rely on complex geometry with post-simplification, resulting in inefficient pipelines or limited realism. In this paper, we introduce ImmerseGen, a novel agent-guided framework for compact and photorealistic world generation that decouples realism from exhaustive geometric modeling. ImmerseGen represents scenes as hierarchical compositions of lightweight geometric proxies with synthesized RGBA textures, facilitating real-time rendering on mobile VR headsets. We propose terrain-conditioned texturing for base world generation, combined with context-aware texturing for scenery, to produce diverse and visually coherent worlds. VLM-based agents employ semantic grid-based analysis for precise asset placement and enrich scenes with multimodal enhancements such as visual dynamics and ambient sound. Experiments and real-time VR applications demonstrate that ImmerseGen achieves superior photorealism, spatial coherence, and rendering efficiency compared to existing methods.

Index Terms—Scene generation, agents, texture synthesis, virtual reality.

1 Introduction

Humans have an innate desire to create and inhabit personalized worlds, whether it's children building sandcastles or artists designing landscapes. This creative drive extends to digital spaces, especially in VR/XR applications, where users expect to be immersed in custom environments with panoramic views, high-fidelity visuals, and real-time interactions. However, building such immersive 3D scenes remains challenging. Handcrafted 3D modeling requires specialized skills and considerable effort, while recent generative methods like object-compositional generation [9, 20, 64], LLM-powered modeling tools [1] and frameworks [31, 35, 78], and approximating through 3D Gaussians [60, 65, 79] often struggle to balance photorealism with computational efficiency. These approaches prioritize fully detailed geometry or massive Gaussians to achieve realism, but often result in overly complex scene representations that hinder real-time performance on VR headsets, or require handcrafted and time-consuming decimation and compression to make them usable. This raises a fundamental issue regarding the necessity of exhaustive 3D modeling for immersive VR. We argue that starting with complex geometry is not a prerequisite, especially when considering the limited explorable areas and finite

- *Jinyan Yuan, Bangbang Yang, Keke Wang, Panwang Pan, Lin Ma, Xuehai Zhang, Xiao Liu and Yuewen Ma (corresponding author) are with Bytedance. E-mail: {yuanjinyan | yangbangbang | wangkeke.1014 | panpanwang | malin.ml | xuehai.zhang | liuxiao.ai | mayuewen}@bytedance.com*
- *Zhaopeng Cui is with Zhejiang University. E-mail: zhpcui@zju.edu.cn*
- *Jinyan Yuan and Bangbang Yang contributed equally to this work.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

computational budgets.

In this paper, we propose ImmerseGen, a novel agent-guided framework that models immersive scenes as hierarchical compositions of lightweight RGBA-textured geometric proxies, including simplified terrain meshes and alpha-textured billboard meshes.

The formulation offers several important advantages:

1) Rather than modeling the scene with complex geometry and then simplifying it, our approach bypasses this process by generating photorealistic texture directly on lightweight geometric proxies leveraging SOTA image generators, alleviating reliance on detailed asset creation and preserving the texture quality without artifacts introduced in decimation or Gaussian approximations.

2) Such modeling paradigm enables agents to flexibly guide generative models in synthesizing coherent, context-aware textures that integrate seamlessly with the panoramic world;

3) It delivers VR-ready scene representations that allow real-time rendering at smooth frame rates.

To establish this hierarchical paradigm, ImmerseGen first creates the base layer world, which employs a terrain-conditioned RGBA texturing scheme on a simplified terrain mesh with user-centric UV mapping. More specifically, it employs a user-centric texturing and mapping scheme that synthesizes and allocates higher texture resolution based on central camera origin, prioritizing the primary viewing area, rather than uniformly covering the entire scene with limited quality [9, 45]. Then, ImmerseGen automatically enriches the environment with generative scenery assets, which are clearly separated into distinct depth levels. Midground assets, such as distant trees or vegetation, are efficiently created using planar billboard textures, while foreground assets, closer to the user, are generated with alpha-textured cards placed over retrieved low-poly 3D template meshes. This mechanism smartly allocates representation detail, maintaining both visual fidelity and rendering efficiency at every scale.

While RGBA-textured proxies simplify asset modeling, assembling coherent 3D scenes still requires manual adjustment and expert knowledge. To simplify this process, we develop a Visual-Language Models (VLMs)-based agentic system that interprets user text prompts into immersive environments. However, directly using VLMs often faces challenges in spatial understanding that hinder layout accuracy. To address this, we introduce a grid-based semantic analysis strategy, enhancing the spatial comprehension with coarse-to-fine visual prompt and raycasting-based validation, thus mitigating placement errors and inconsistencies existing in naïve VLMs. Moreover, ImmerseGen enriches the immersive experience by incorporating modular dynamics (e.g., flowing water, drifting clouds) and ambient audio (e.g., wind, birdcalls), delivering a fully multisensory environment.

In summary, our contributions are as follows:

1) We propose ImmerseGen, a novel agent-guided 3D environment generation framework that uses simplified geometric proxies with alpha-textured meshes to produce compact, photorealistic worlds ready for real-time mobile VR rendering.

2) We propose a novel RGBA texturing paradigm that first synthesizes 8K terrain textures using a geometry-conditioned panorama generator via user-centric mapping, and then directly generates alpha-textured proxy assets, avoiding fidelity loss inherent in mesh decimation.

3) To automate scene creation from user prompts, we introduce VLM-based modeling agents equipped with a novel grid-based semantic analysis, enabling 3D spatial reasoning from 2D observations and ensuring accurate asset placement. ImmerseGen further enhances immersion with dynamic effects and ambient audio for a multisensory experience.

4) Experiments on multiple scene-generation scenarios and live mobile VR applications show that ImmerseGen outperforms previous methods in visual quality, realism, spatial coherence, and rendering efficiency for immersive real-time VR experiences.

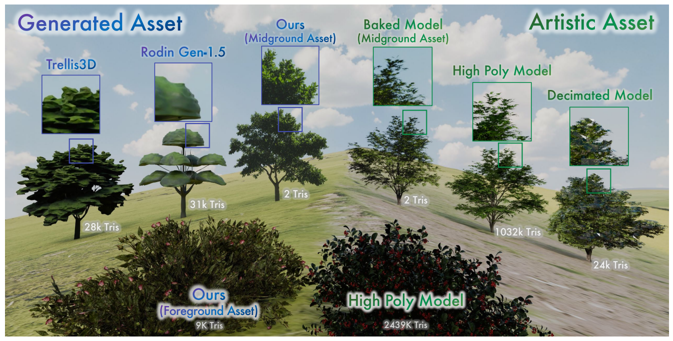


Fig. 2: **Asset comparison from different sources.** We compare assets created by learning-based generative methods (blue labels), artists (green labels), and ours. Our generative RGBA-textured proxy assets achieve better visual details than existing models [55, 74] with fewer triangles, delivering photorealistic appearance comparable to artist-created high-poly or baked assets.

2 Related works

2.1 Agentic Scene Generation

Early efforts in procedural content generation (PCG) for immersive environments primarily rely on rule-based systems [13, 32, 40, 71], where spatial relationships and asset placements are meticulously defined through handcrafted rules. Infinigen [44] advances this process by leveraging Blender scripts to orchestrate multiple procedural generators, enabling the creation of larger and more complex scenes. However, PCG methods inherently limit adaptability to novel scenarios and user-driven instructions. The advent of LLMs and VLMs introduces a paradigm shift in scene generation, enabling more intuitive, instruction-based workflows. Recent methods like BlenderMCP [1] increasingly harness the capabilities of LLMs to automate the generation process, employing function-calling agents to interpret text prompts [39, 62, 80], design scene layouts [29, 49], and populate environments [31, 78] with assets retrieved from pre-built libraries [1, 23, 33, 35, 48, 78]. These systems demonstrate significant potential in generating diverse, large-scale scenes from high-level descriptions, streamlining the content creation pipeline. However, existing LLM/VLM-based approaches rely heavily on asset libraries, often requiring a trade-off between quality and efficiency. Moreover, the precision of VLM-guided asset placement often proves insufficient in complex scenarios. In contrast, ImmerseGen addresses these limitations by introducing lightweight proxy assets and semantic grid-based arrangement by agents, enabling the creation of compact, photorealistic worlds.

2.2 Learning-based Generation

Recently, learning-based generation methods have shown promising results in creating 2D and 3D content [18, 46, 74, 82]. However, unlike 3D object generation that benefits from diverse object datasets [8, 67] for model training, 3D scene generation still faces challenges [17, 20, 37, 54, 57] due to the lack of comprehensive scene-level data and unified representations. Early methods either learned a generative neural field with GAN [4, 16, 30, 56] or 2D diffusion priors [6, 70, 75], but failed to produce detailed appearance. Recently, other lines of work tend to generate images and lift them to 3D space through depth prediction, combined with outpainting techniques to expand the scene [5, 11, 65, 66]. However, these methods typically produce incomplete 3D worlds (e.g., missing 360-degree views or geometry under the feet), thus failing to meet the demands of immersive VR applications. To create a complete surrounding world, some methods lift the generated panoramic images [52, 68] to 3D space with depth estimation and inpainting [27, 60, 77, 79], but still faces challenges in producing spatially coherent worlds due to the inconsistency of novel view inpainting. More recent approaches utilize video models for 3D scene creation [12, 14, 28, 50], which either suffer from blurry backgrounds or fail to guarantee fully explorable 360-degree environments. Addition-

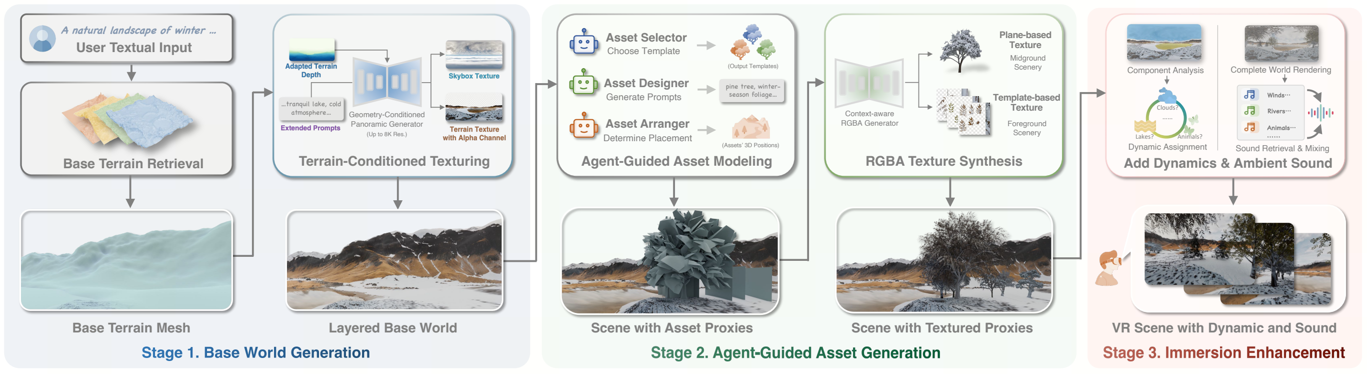


Fig. 3: **Overview.** Given a user's text input, the agent first retrieves a base terrain. Conditioned on the terrain depth and an extended prompt, panoramic textures for the terrain and sky are generated to form a layered base world. Next, VLM-based asset agents enrich the scene by selecting asset proxies as foreground or midground scenery, designing detailed asset prompts, and determining optimal asset placement. Each asset is instantiated via RGBA texture synthesis. Finally, the agent incorporates dynamic visual effects and synthesized ambient sound, producing a lightweight and photorealistic world.

ally, these methods often produce a large number of point clouds or 3D Gaussians for scene representation, making it challenging to achieve high-quality rendering while maintaining reasonable computational costs.

2.3 Traditional Asset Creation

Conventional asset creation pipelines typically follow a two-stage process: detailed geometric modeling followed by texture mapping. This modeling-first paradigm is prevalent in CG content production where artists craft complex meshes and apply high-resolution textures to achieve visual realism. However, when deploying such assets in real-time rendering applications like VR or games, these models require simplification via decimation techniques, such as mesh simplification [26, 34], billboard generation [7, 22], or level-of-detail (LOD) hierarchies [19, 76], along with baked textures. For natural scenes, many works on terrain generation and vegetation modeling [24, 25] have been proposed, yet they often lack diversity and realism. While effective, this conventional workflow incurs significant manual effort or computational cost, as it first generates excessively detailed primitives only to later reduce their complexity for efficiency. In contrast, ImmerseGen eliminates the need for post-hoc simplification by directly synthesizing alpha-textured assets tailored for efficient rendering, enabling photorealistic scene generation optimized for immersive applications.

3 Method

We introduce ImmerseGen, an agent-guided framework for generating immersive 3D scenes from textual prompts. As shown in Fig. 3, we construct the scene hierarchically from base terrain guided by VLM-based agents. First, we generate a layered base world via terrain-conditioned texturing, where panoramic sky and RGBA terrain textures are synthesized upon a retrieved terrain mesh (Sec. 3.1). Next, we enrich the scene through placing lightweight asset proxies by agents with semantic grid-based analysis. The selected assets are then instantiated using a context-aware RGBA texture synthesis scheme (Sec. 3.2). Finally, we augment the scene with dynamic effects guided by agents, such as flowing water and ambient sound, delivering a multisensory experience (Sec. 3.3).

3.1 Base World Generation

From textual prompts to base terrain. Given a user's textual prompt describing the world, a suitable base terrain mesh is first retrieved from a pre-generated template library. These templates are created using procedural content generation tools, followed by post-processing steps including remeshing, visibility culling, and captioning to support effective retrieval. We deploy an LLM agent that retrieves a suitable terrain template and enhances prompts with imaginative and

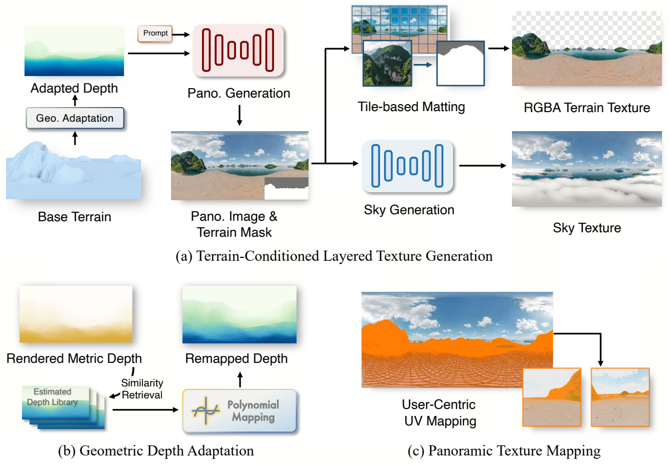


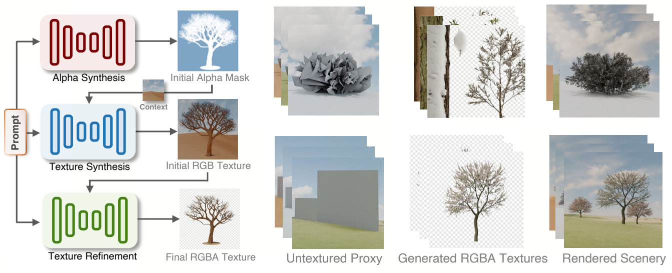
Fig. 4: **Workflow of base world generation.** Panoramic textures for terrain mesh and sky are generated for the base world. To tame the diffusion model for terrain texturing, we propose geometric adaptation (b) for depth control and user-centric texture mapping (c).

contextually relevant details from user's textual input, to improve scene diversity and ensure coherence between terrain characteristics and the input prompt. Since visual diversity is primarily introduced through subsequent generative texturing, this strategy strikes a practical balance between efficiency and variety.

Terrain-conditioned texturing. As demonstrated in Fig 4 (a), given a base terrain mesh and text prompts, we first generate panoramic sky textures and alpha ground textures upon the mesh. To support terrain texture synthesis in equirectangular projection (ERP), we adopt a two-stage training pipeline. We first train a panoramic diffusion model built upon Stable Diffusion XL [42] on ERP data conditioned on textual prompts [46]. Then, we extend this model by training a depth-conditioned ControlNet [73], which takes as input a panoramic depth map $\mathbf{D}_{\mathscr{M}}$ estimated from a neural depth estimator [59]. During inference, we combine both modules to generate a panoramic texture $\mathbf{I}_t$ that aligns with the terrain mesh $\mathscr{M}$, formulated as:

$$\mathbf{I}_t = \mathscr{U}(\mathscr{G}(\mathbf{D}_{\mathscr{M}}; \mathscr{C}_{\text{Global}}, \mathscr{C}_{\text{Region}})), \tag{1}$$

where $\mathbf{D}_{\mathscr{M}}$ is the conditioning panoramic depth map rendered from the terrain mesh, $\mathscr{G}$ is the conditional diffusion model, $\mathscr{C}_{\text{Global}}$ is the text prompt for global geographic description, $\mathscr{C}_{\text{Region}}$ is the optional regional prompts for generating designated geographic features (such



(a) RGBA Texture Synthesis (b) Diverse Scenery in Different Contexts from Single Proxy

Fig. 5: The proposed **context-aware texture synthesis** (a) generates diverse, contextually coherent RGBA textures directly on lightweight proxies for both foreground and midground scenery (b).

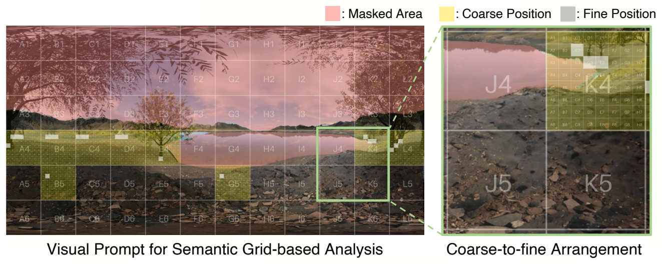

🟥 : Masked Area
🟨 : Coarse Position
⬜ : Fine Position
J4
K4
J5
K5
Visual Prompt for Semantic Grid-based Analysis
Coarse-to-fine Arrangement

Fig. 6: The proposed **semantic grid-based analysis** overlays a labeled grid and masks unsuitable regions as visual prompts. This enables the VLM agent to progressively select grid cells in a coarse-to-fine manner, improving the accuracy and semantic coherence of asset arrangement.

as water body, see supp. Sec. 1.1), and $\mathscr{U}$ is the conditioned upscaling model that produces 8K textures to enhance details utilizing a tile-based generation approach inspired by [2].

To separate the terrain texture and sky texture while maintaining high resolution, we perform tile-based matting and sky outpainting on the panorama, which yields an 8K fine-grained alpha matte and pure sky texture guided on the terrain mask. This detailed alpha matte allows low-poly terrain meshes to exhibit highly detailed landscape silhouettes (e.g., trees and houses against the sky).

Depth control with geometric adaptation. While it is technically plausible to apply conditional diffusion for mesh texturing, we find it non-trivial to produce 3D-coherent textures that align well with the terrain and meet immersive standards, i.e., degraded quality as shown in Fig. 9. This difficulty arises primarily from the domain gap between the estimated relative depth for ControlNet training and rendered metric depth maps for inference-time conditioning. To tackle this issue, we propose a geometric adaptation scheme that remaps the rendered metric depth to better match the domain of training-time estimated depth. Specifically, we retrieve the most similar depth map $\mathbf{D}_{\text{Retrieve}}$ from a sampled training set $\mathscr{L}$ using cosine similarity, and apply a polynomial remapping function:

$$\hat{\mathbf{D}}_{\mathscr{M}} = \mathscr{P}(\mathbf{D}_{\mathscr{M}}; \mathbf{D}_{\text{Retrieve}}), \tag{2}$$

where $\hat{\mathbf{D}}_{\mathscr{M}}$ is the remapped depth, and $\mathscr{P}$ is a third-degree polynomial mapping function. Practically, we downsample both $\mathbf{D}_{\mathscr{M}}$ and $\mathbf{D}_{\text{Retrieve}}$ to 32×16 resolution to estimate the polynomial coefficients, which are then applied to the full-resolution depth map $\mathbf{D}_{\mathscr{M}}$. By conditioning on the remapped depth, the panoramic diffusion model generates textures that are well aligned with the terrain.

Terrain texture mapping. To efficiently texture the terrain with the generated panoramic texture while preserving visual fidelity, we precompute user-centric panoramic UV coordinates for the terrain mesh, as illustrated in Fig. 4 (c). Thus, the texture can be directly sampled during the rendering without back-projection or baking procedures. Specifically, the UV coordinate for each mesh vertex can be calculated by transforming the coordinates from object space to camera space. Given a mesh vertex position in camera space $\mathbf{p} = (x, y, z)^\top$, the corresponding UV coordinate $\mathbf{u} = (u, v)^\top$ on the panoramic texture $\mathbf{I}_t$ can be calculated as:

$$\mathbf{u} = \left(\frac{1}{2\pi} \arctan(\frac{x}{-z}) + \frac{1}{2}, \frac{1}{\pi} \arcsin(\frac{y}{\|\mathbf{p}\|}) + \frac{1}{2} \right)^\top, \tag{3}$$

where $\|\mathbf{p}\|$ denotes the L2-norm of the vertex position. To prevent texture stretching at horizontal seams, we detect UVs crossing the panoramic boundary and offset them for correct wrapping, then set the texture wrapping mode to *repeat* for seamless interpolation of panoramic texture sampling.

To further improve visual fidelity around the user's viewpoint, particularly in the polar region where the ERP exhibits stretching, we first adopt an ERP-to-cubemap refinement scheme, using an image-to-image diffusion method [36] to repaint the bottom area. Then, we partition the mesh by cropping its bottom area and then reassign UV coordinates of this mesh to directly sample textures from the bottom map. Additionally, to achieve better geometric realism, we incorporate a displacement map obtained from an adapted depth estimation model [59] (see supp. Sec. 1.5.).

3.2 Agent-Guided Asset Generation

To enrich the base world with photorealistic scenery, we then add more generative 3D assets (such as vegetation) to the scene. Unlike prior methods that rely on complex modeling pipelines [7] or off-the-shelf asset retrieval, our framework dynamically generates unique, alpha-textured asset proxies from coarse templates using generative texture synthesis, thus simplifying asset creation and enabling more flexible agent-driven design.

Defining proxies by distance. We employ distinct proxy types based on the distance between the user and the asset to balance rendering quality and performance, which delivers a realistic appearance comparable to the artists' baked models while alleviating the cost of baking or decimation. As demonstrated in Fig. 1 (b) and Fig. 2, for midground objects, since users cannot perceive detailed depth changes of object surfaces, we synthesize RGBA textures on distant planar mesh (see Fig. 5 (c), a.k.a. billboard texture). For foreground objects that require depth perception, we generate alpha textures for each group of shared materials in a template mesh with alpha cards (such as tree leaves and trunks, see Fig. 5 (b)).

Asset selection and designing. To create diverse and contextually coherent scenery assets, we develop VLM-based agents to guide the asset design pipeline. First, the **asset selector** analyzes the rendered base world image and user's textual description to retrieve suitable foreground asset templates from an offline-generated library indexed by description, e.g., pine trees for mountainous regions or bushes for arid deserts. Next, the **asset designer** crafts detailed textual prompts to guide generative models in synthesizing these scenery assets. In practice, the designer examines both the generated base-world image and selected texture templates, and produces detailed descriptions for each scenery asset (such as categories, season, styles, etc.).

Asset arrangement with semantic grid-based analysis. To ensure that generative assets are placed in semantically appropriate and visually plausible locations, we introduce an **asset arranger** that analyzes the base world image to produce 2D position candidates, which are then back-projected to determine 3D positions through raycasting and validation. One primary challenge for the asset arranger is to generate reasonable 3D placements based solely on image-based observation. A naïve approach is to let the agent directly output the coordinate, which generally results in inaccurate positions and meaningless layout (see Sec. 4.3) due to the limited spatial understanding ability of existing models [58]. To address this, we propose a semantic grid-based position proposal scheme, which significantly improves the asset arrangement quality. As shown in Fig. 6, we overlay the base world image with a labeled grid and mask out unsuitable regions (e.g., water, sky), forming a structured visual prompt for the VLM agent. The agent first selects coarse grid cells given this visual prompt. Then, for finer placement, each selected cell is zoomed in and subdivided into sub-grids, from which the agent will select a more precise sub-cell. The final positions are determined by randomly selecting a point within the sub-cell.

Context-aware RGBA texture synthesis. Once the agents have determined the per-asset placement and textual descriptions, we proceed to instantiate each asset by synthesizing its RGBA texture in context with the base world. To facilitate seamless integration, we propose a context-aware cascaded RGBA texture synthesis model conditioned on base world background textures, which is inspired by the layered diffusion model [72].

Given a scenery prompt $\mathscr{C}_s$, the alpha synthesis module $\mathscr{G}_a$ first generates an alpha mask $\mathbf{M}_c = \mathscr{G}_a(\mathscr{C}_s) \in \mathbb{R}^{H \times W}$, serving as a sketch for subsequent texturing. To incorporate contextual information from the base world, the RGB base texture reference $\mathbf{I}_b \in \mathbb{R}^{H \times W \times 3}$ is injected into an initially empty RGBA canvas through alpha blending guided by $\mathbf{M}_c$. Then the texture synthesis module $\mathscr{G}_t$ generates an initial scenery texture from the alpha-blended reference with the alpha mask $\mathbf{M}_c$. Note that the generated texture usually produces boundaries that is not perfectly aligned with the given alpha mask. Thus, the alpha channel of the initial texture is further refined through a diffusion-based refinement module $\mathscr{R}$. The full process to generate final scenery texture $\mathbf{I}_s \in \mathbb{R}^{H \times W \times 4}$ is formulated as:

$$\mathbf{I_s} = \mathscr{R}\left(\mathscr{G}_t\left(\mathbf{M}_c, \mathbf{I}_b; \mathscr{C}_s\right)\right). \tag{4}$$

For foreground scenery that already contains an alpha channel in its template model, we directly reuse its alpha as $\mathbf{M}_c$ to ensure the correct structure.

3.3 Multi-Modal Immersion Enhancement

To further enhance immersion beyond static 3D visuals, we introduce agent-guided multi-modal enhancement in visual dynamics and sounds (see the right part of Fig. 3).

Dynamic shader-based effects. The immersive enhancer analyzes the scenery component of the generated scene, and adds shader-based dynamic effects for natural elements such as flowing water, drifting clouds, and falling rain. These effects are implemented using customizable shader parameters, including procedural flow maps, noise-based motion textures, and screen-space animations, which bring liveliness to the scene while maintaining real-time performance (see supp. Sec. 1.3 for details).

Ambient sound synthesis. The immersive enhancer then synthesizes ambient sounds using a library of natural soundtracks tagged by content. Specifically, the agent analyzes the rendered panorama of the complete scene and retrieves suitable natural soundtracks (such as birds, winds, and water) from the library. To support uninterrupted playback, we apply crossfading to seamlessly mix tracks for audio looping (see supp. Sec. 1.4 for details).

4 Experiments

4.1 Implementation Details

We utilize Blender [51] as our core scene modeling framework, integrating terrain texture projection, asset placement, scene rendering, and VR-ready scene export. We develop world modeling agents powered by GPT-4o, each configured with distinct system prompts (see supp. Sec. 1.2). The terrain library is primarily generated with Blender's A.N.T. Landscape add-on, resulting in a collection of about 10 initial templates for texturing. We build an asset library of about 50 foreground object templates, each modeled using alpha cards. The number of asset types when generating scenes ranges from 0 to 10, determined adaptively by the agents. The distance ranges for foreground and midground objects are configured as 2–10 meters and 20–50 meters, respectively. The terrain-conditional diffusion model is fine-tuned from SDXL [42] on 10K equirectangular terrain images collected from

Table 1: We perform quantitative comparison on the generated 3D scenes, and compare the complexity of representation (primitive count) and runtime performance (FPS) on VR devices.

Methods	Quantitative Metrics			Complexity & Perform.	
	CLIP-Score ↑	CLIP-Aesthetic ↑	QA-Quality ↑	Prim. Count ↓	FPS ↑
Infinigen	-	4.9546	3.0426	1276k	∼7
WonderWorld	27.0417	5.0116	2.6298	1632k	∼14
DreamScene360	29.3556	4.8283	2.1446	2097k	∼8
LayerPano3D	**29.4633**	5.1513	3.4812	14577k	N/A
Ours	28.8933	**5.4834**	**3.5445**	**223k**	**∼79**

Unreal Engine (UE) scene rendering and Internet sources, with a learning rate of 1×10^{-5} for 30K steps and batch size 4. To further constrain the model when generating ERP images, we apply a circular padding to ensure continuity between the leftmost and rightmost edges of the panorama [10, 69]. For training the depth ControlNet [73], we generate mixed depth maps using Depth-Anything V2 [59] and MiDaS [3] with perspective-to-panorama fusion, applying random scale and shift augmentations to improve robustness. We adapt VitMatte [63] with tile-based matting for high-resolution sky segmentation and PowerPaint [81] for sky outpainting. We leverage pre-trained diffusion models [72, 81] for training-free RGBA texture synthesis. To balance realism and performance, we pre-bake high-resolution panoramic maps under global illumination as run-time unlit materials, enabling photorealism without the need for real-time lighting for VR applications (see supp. Sec. 1.6). All models are trained on a single NVIDIA A100-80G GPU. The entire generation pipeline takes about 10 minutes deployed on a single NVIDIA RTX 4090 GPU (see supp. Sec. 1.7).

4.2 Comparison on Scene Generation

Baselines. We compare our method with recent scene generation methods across different categories: (1) Infinigen [45], which uses procedural generation with physics-based modeling; (2) DreamScene360 [79], which lifts panoramic images to 3D space; (3) WonderWorld [65], which generates scenes through perspective outpainting. (4) LayerPano3D [61], similar to DreamScene360, but adopts a layered representation. For a fair comparison, we use Infinigen's scene configurations that match the same category with our generated scenes, adopt the same enhanced text prompts as our method for DreamScene360 and LayerPano3D, and use the cropped perspective images from our generated panorama as the image condition for WonderWorld.

Metrics. For comprehensive comparison with the above methods, we use metrics for evaluating both prompt-scene consistency and aesthetic quality, including CLIP similarity score (CLIP-Score) [43], aesthetic score (CLIP-Aesthetic) [47] and the VLM-based visual scorer Q-Align (QA-Quality) [53].

Quantitative results. We present the quantitative comparison of our method with the baselines in Tab. 1. We generate 18 scenes for each method and render video sequences along circular camera trajectories, following the protocol of previous work [79]. Evaluation metrics are reported as the average scores across all frames and scenes. The generation prompts encompass a wide spectrum of natural landscape settings (e.g., glaciers, beaches, and forests) to evaluate the generalizability across diverse environments. (as illustrated in Fig. 7). As shown in Tab. 1, our method outperforms all baselines in CLIP-Aesthetic score and QA-Quality, demonstrating the superior visual quality of our generated scenes. For CLIP-Score, DreamScene360 and LayerPano3D also achieve competitive scores since they prioritize semantic alignment during training, while our method generates diverse textures(e.g., various geographic features instead of bare ground, see Fig. 7).

Qualitative results. We present the qualitative comparisons in Fig. 7, evaluating the results based on several criteria: visual quality (absence of blurring or artifacts), diversity (the range of landscape features and scenery textures), and coherence (spatial consistency across the scene). Infinigen, relying primarily on limited procedural generators, produces scenes lacking diversity (e.g., monotonous ice, row 1) and

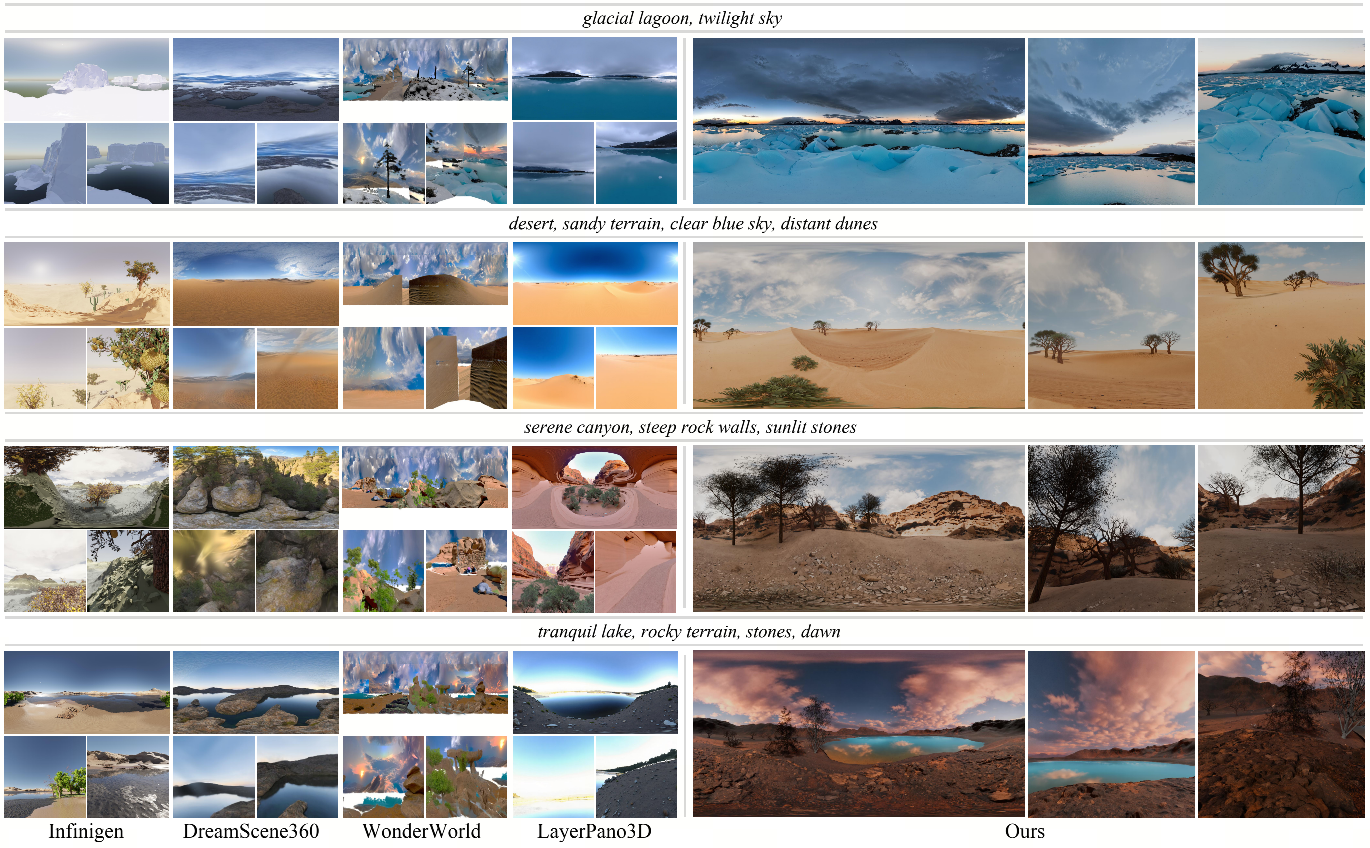

glacial lagoon, twilight sky
desert, sandy terrain, clear blue sky, distant dunes
serene canyon, steep rock walls, sunlit stones
tranquil lake, rocky terrain, stones, dawn
Infinigen
DreamScene360
WonderWorld
LayerPano3D
Ours

Fig. 7: We compare our method with Infinigen [44], DreamScene360 [79], WonderWorld [65] and LayerPano3D [61] based on the generated 3D scenes using identical text prompts, visualizing both panoramic and perspective views of the generated scenes.

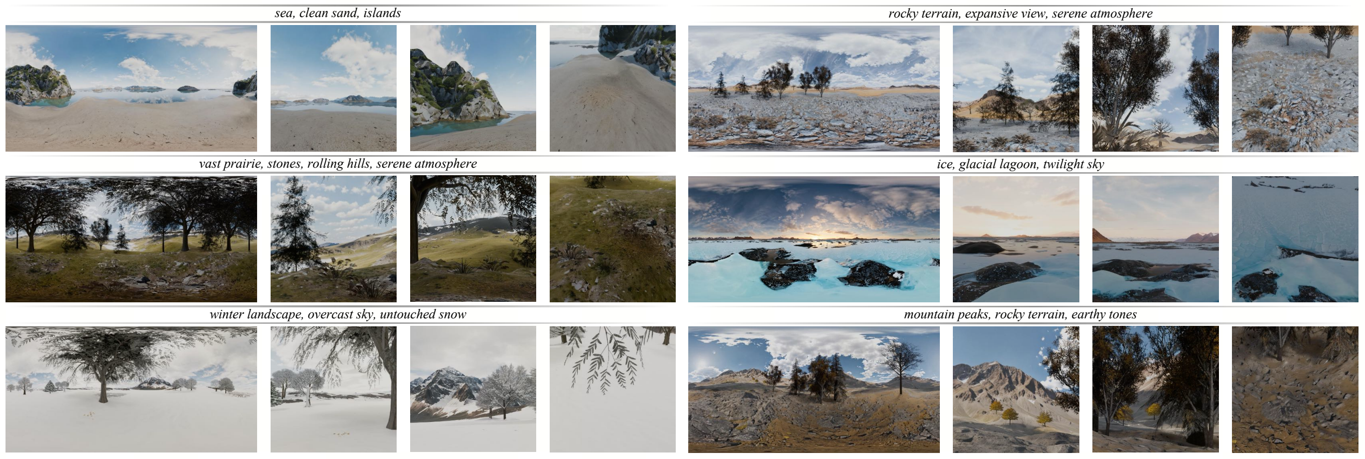

sea, clean sand, islands
rocky terrain, expansive view, serene atmosphere
vast prairie, stones, rolling hills, serene atmosphere
ice, glacial lagoon, twilight sky
winter landscape, overcast sky, untouched snow
mountain peaks, rocky terrain, earthy tones

Fig. 8: We present more examples of generated environments in panoramic and perspective views.

coherence (e.g., contextually incompatible trees, row 4). For DreamScene360, although it achieves consistent views with a panoramic lifting strategy, it lacks diverse scenery contents and also shows blurry artifacts (see the slanting floaters in the perspective views from the second and third row in Fig. 7) due to the instability of inpainting-based optimization and the limited resolution of 3D Gaussians. For WonderWorld, since it relies on outpainting to generate a complete world, it cannot ensure view consistency across different views and results in fragmented scenes. LayerPano3D produces aesthetic and consistent results with DiT-based panorama generator, but is prone to blurry artifacts and visible gaps at the layer boundary. By contrast, our method builds up the world with hierarchical alpha-textured proxies while considering the 3D coherence with agent-guided modeling, preserving consistent quality across views and delivering immersive scenery content. We provide more examples of generated realistic scenes in Fig. 8 and examples of scenes in a variety of styles and environments, demonstrating the generalizability of our approach.

User study. We conducted a user study involving 50 participants (33 of whom had expertise in 3D or computer graphics) to evaluate the 18 generated scenes. Using the PICO 4 Ultra VR headset, participants viewed randomized scene groups without time constraints and performed a forced-choice preference task across three aspects: Perceptual Quality (visual aesthetics and the absence of artifacts), Realism & Coherence (environmental plausibility and spatial consistency), and

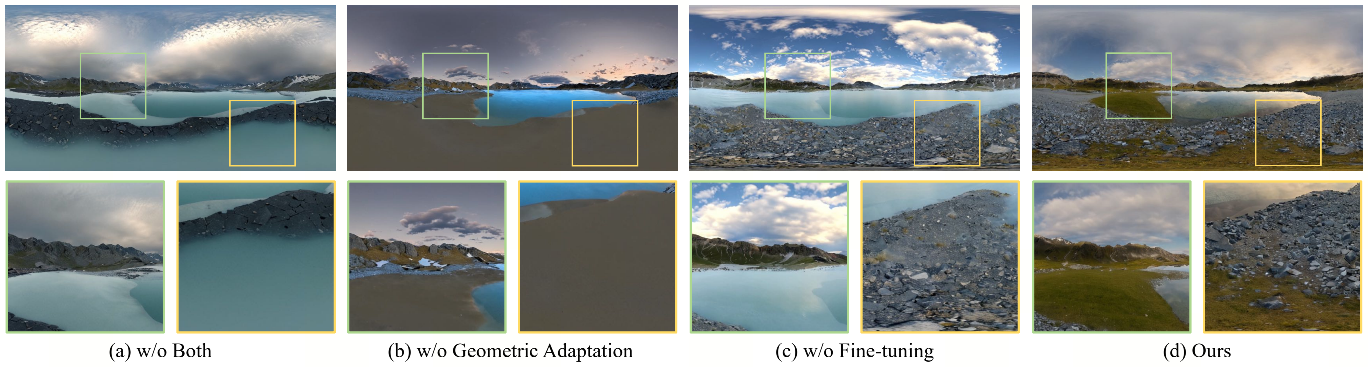


Fig. 9: We analyze the geometric adaptation and fine-tuning of the conditioning network for terrain-conditioned texture generation.

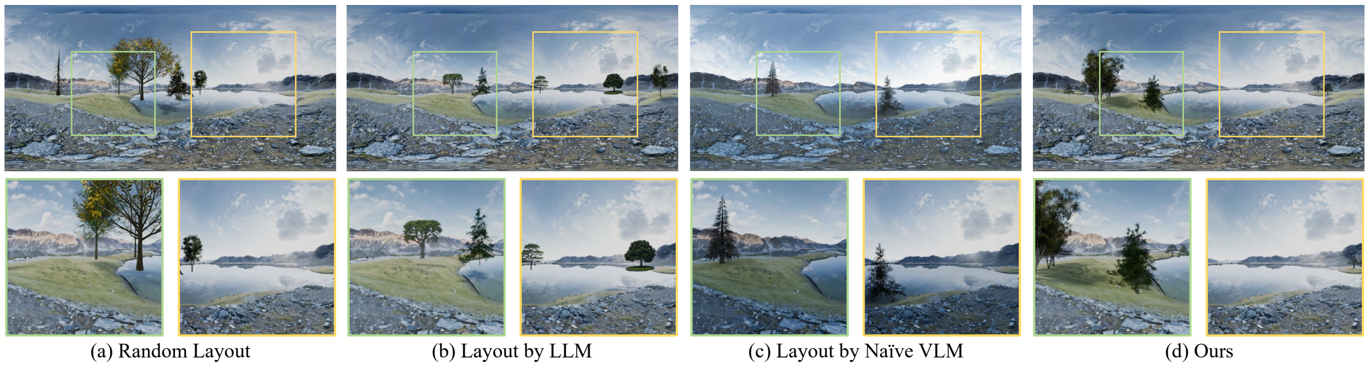


Fig. 10: We compare our semantic grid-based analysis for asset layout with different layout approaches.

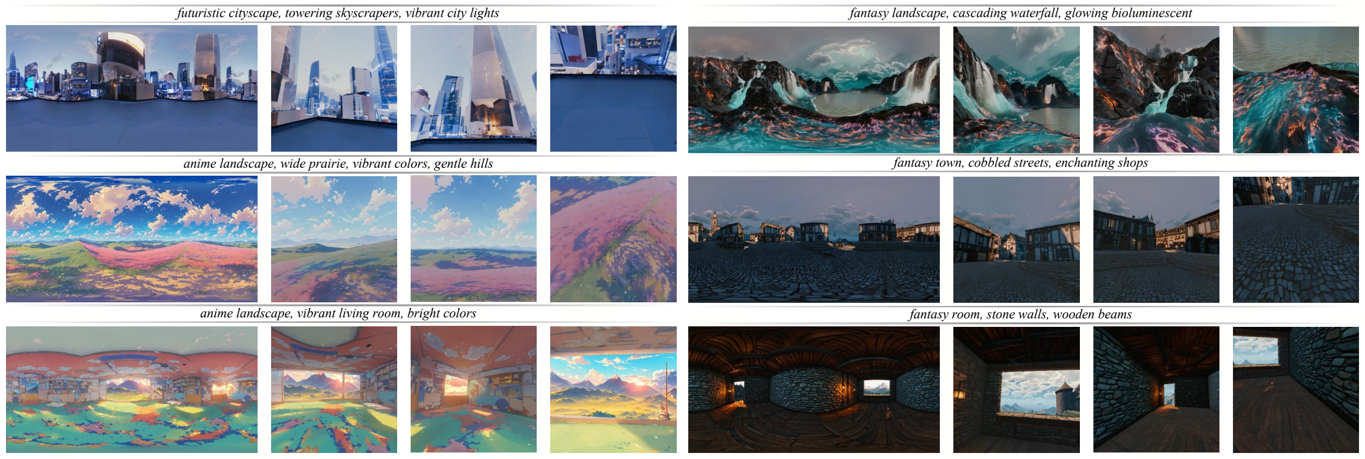

futuristic cityscape, towering skyscrapers, vibrant city lights
fantasy landscape, cascading waterfall, glowing bioluminescent
anime landscape, wide prairie, vibrant colors, gentle hills
fantasy town, cobbled streets, enchanting shops
anime landscape, vibrant living room, bright colors
fantasy room, stone walls, wooden beams

Fig. 11: We present examples of generated scenes in various styles and urban or indoor environments beyond outdoor natural settings.

Textual Alignment (adherence to the input prompt). LayerPano3D was excluded as its primitive count precluded VR rendering. As shown in Tab. 2, our method was significantly preferred over baselines, demonstrating superior visual fidelity and alignment. This research was conducted in accordance with institutional policies, which did not require an ethical review. Informed consent was obtained from all participants before they attended the study.

Table 2: We perform user studies on the generated 3D scenes.

Method	Perceptual Qual. ↑	Realism & Coherence ↑	Textual Align. ↑
Infinigen	7.12%	5.83%	6.04%
WonderWorld	13.46%	7.50%	11.49%
DreamScene360	<u>24.01%</u>	<u>33.89%</u>	<u>38.22%</u>
Ours	**55.41%**	**52.78%**	**44.25%**

Complexity of representation and performance. We compare the complexity of scene representation and runtime performance on VR devices (Snapdragon XR2 Gen 2 platform). We calculate the average primitive counts and FPS of all scenes for each method. As shown in Tab. 1, methods using 3D Gaussians as representation (DreamScene360 [79] and WonderWorld [65]) either generally achieve only 8-14 FPS even with foveated rendering, or fail to launch on VR devices [60]. For Infinigen [44], since it generates a detailed world with intricate procedural geometry and materials from generators, it remains computationally expensive for real-time rendering. In contrast, our method maintains a compact representation while preserving scene

Table 3: Ablation study of the proposed geometric adaptation in terrain-conditioned texturing.

	w/o both	w/o Geo. Ada.	w/o Fine-tuning	Ours
QA-Quality ↑	3.6938	3.7630	3.7614	**3.9057**

Table 4: Ablation study of the semantic grid-based analysis.

Method	Random layout	LLM	Naïve VLM	Ours
CLIP-Aesthetic ↑	5.4963	5.5212	5.5350	**5.5739**

Table 5: Ablation study on the aesthetic improvement of adding proxy scenery.

	w/o Both scenery	w/o Midground scenery	w/o Foreground scenery	Ours
QA-Aesthetic ↑	2.1143	2.3287	2.3562	**2.4408**
CLIP-Aesthetic ↑	5.1540	5.3307	5.2453	**5.3634**

quality, achieving an average FPS of **79+** on VR devices.

4.3 Ablation Studies

Geometric adaptation. We first analyze the geometric adaptation for projected terrain depth and fine-tuning of the conditioning network in terrain-conditioned texturing (Sec. 3.1). As shown in Tab. 3, we evaluate the QA-Quality for rendered sequences of 10 generated base world in different configurations. By ablating both strategies, the generated terrain texture fails to produce a plausible texture (water area on the bottom in Fig. 9 (a)). By enabling fine-tuning, the terrain texture precisely reflects the ground but with a monotonous appearance (see Fig. 9 (b)). By enabling geometric adaptation, the ground texture shows more detail (rocks on the bottom in Fig. 9 (c)). With all the strategies, we achieve terrain texture with fine-level details and realistic structure (see Fig. 9 (d)).

Semantic grid-based analysis. We then evaluate the efficacy of the proposed semantic grid-based analysis for the asset arranger (Sec. 3.2). Specifically, we compare our method with different strategies, including random layout generation, LLM-based generation that outputs object coordinates directly, and naïve VLM-based generator that receives unmodified base world images. As shown in Fig. 10, the output of random layout incorrectly places trees on the lake (Fig. 10 (a). The layout generated by generic LLM and naïve VLM improves the coherence by providing compatible texture descriptions and plausible coordinates, but still suffers from inappropriate placements. By using semantic grid-based visual prompts as input for the VLM, our method generates a pleasant scene composition while addressing the placement issue. As shown in Tab. 4, our layout generator achieves higher CLIP-Aesthetic scores comparing rendered panoramas from scenes generated by different layout generation strategies, demonstrating the efficacy of the semantic grid-based visual prompt for improving the placement quality.

Aesthetic contribution with proxy scenery. We also investigate the aesthetic contribution when adding generated proxy scenery into the base world. Specifically, we evaluate the QA-Aesthetic [53] and CLIP-Aesthetic score of 10 randomly selected generated scenes in absence of midground or foreground assets. As shown in Tab. 5 and Fig. 12, the added scenery significantly improves the visual quality by enriching the base world with diverse elements and improving the sense of depth.

5 Discussion

5.1 Design Choices

Panoramic terrain texturing. We observe that existing methods often struggle to synthesize photorealistic terrain, typically producing overly smooth or blurry textures, such as the procedural generation framework Infinigen [45], generative texturing model Easi-Tex [41]

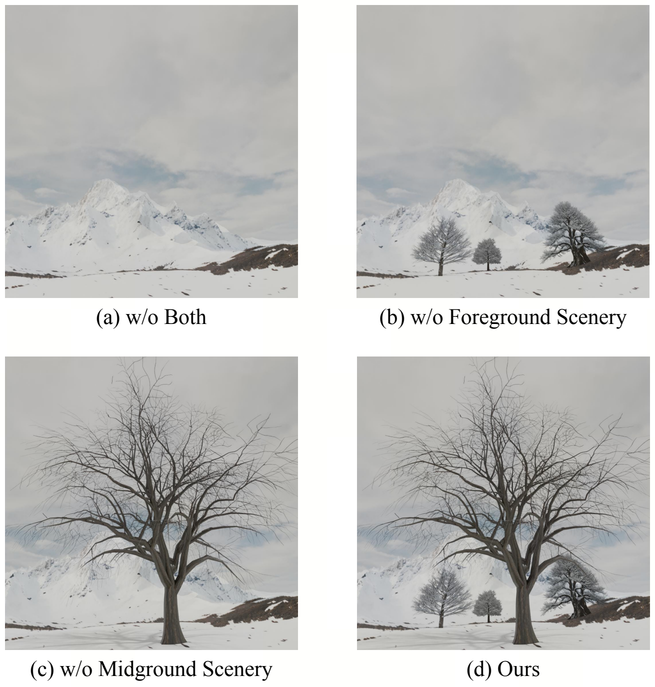


Fig. 12: We visualize the contribution of different scenery by ablating proxy scenery of different types.

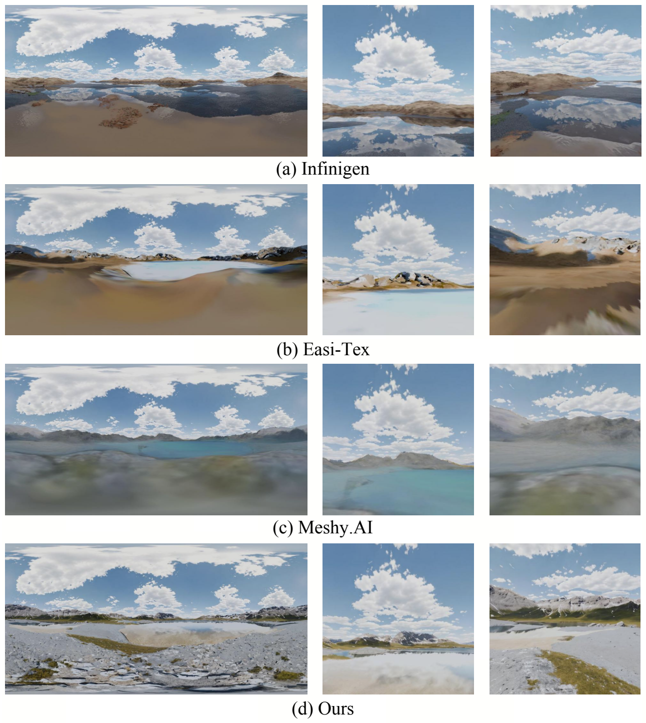


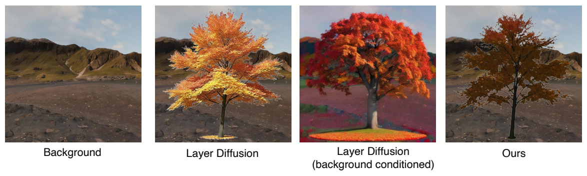
Fig. 13: Our method outperforms other works on terrain texturing.

and commercial texturing tools Meshy.AI [38]. As shown in Fig.13 and Tab.6, our method achieves better texture quality benefiting from the proposed terrain-conditioned panoramic texturing and user-centric UV mapping.

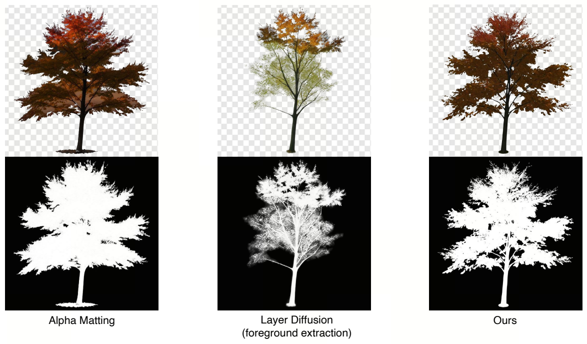
Context-aware asset texturing. While off-the-shelf generative models can produce asset textures, achieving precise and coherent integration of these assets into diverse backgrounds remains challenging.

Table 6: Quantitative Comparison on Terrain Texturing.

Method	Type	CLIP-Aesthetic ↑	QA-Quality ↑
Infinigen	Procedural	5.2937	2.9091
Easi-Tex	Generative	4.8599	2.9242
Meshy.AI	Commercial	4.8750	3.2685
Ours	Generative	**5.4317**	**3.4860**



(a) Comparison on texture coherence.

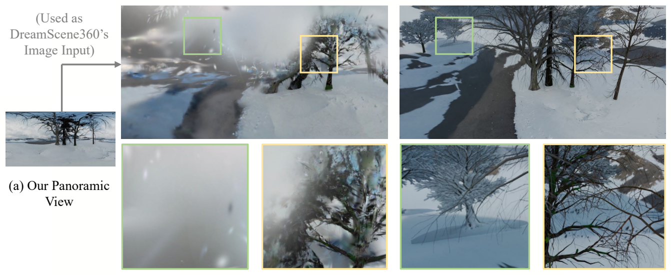


(b) Comparison on alpha quality.

Fig. 14: Our RGBA texture synthesis achieves better coherence and quality comparing with the baseline.

Table 7: Comparison with 3D Gaussians using the same panorama.

Method	CLIP-Score ↑	CLIP-Aesthetic ↑	QA-Quality ↑
DreamScene360	28.2111	4.8748	2.2975
Ours	**28.7806**	**5.3289**	**3.2066**



(a) Our Panoramic View (b) DreamScene360 (c) Ours method

Fig. 15: Our method produces scenes with higher visual fidelity compared to methods based on 3D Gaussians.

We visualize the texture generated from the proposed context-aware RGBA texture synthesis and the original layered diffusion model [72]. As shown in Fig. 14, our generated assets demonstrate greater consistency with the background and alpha quality, attributed to our disentangled generation of color and the alpha channel.

Proxy-based representation. To further evaluate the fidelity of our proxy-based representation relative to 3D Gaussians, we adapt DreamScene360 [79] to generate scenes on the same panorama image from our method. As shown in Fig.15, the generated 3D Gaussians scenes exhibit notable artifacts and reduced visual fidelity, which is further reflected in lower metrics as reported in Tab. 7, indicating its limitations in representing high-quality scenes compared to our approach.

5.2 Limitations and Future Work

First, our method focuses on outdoor scenes rather than indoor scenes with detailed furniture. Second, the scenes are currently restricted to a limited exploration range (typically around $50m^2$) due to the fixed hierarchy of generation levels relative to viewing distance. This could be addressed by incorporating novel view synthesis or video generation techniques [15] to create extensible scenes in future work. Third, our approach relies on pre-built templates for foreground assets, which could be enhanced by integrating procedural generators [21] to enable the creation of more diverse templates.

6 Conclusions

We have presented ImmerseGen, a novel framework for generating photorealistic 3D environments from lightweight geometric proxies tailored for immersive experiences. The proposed generative terrain-conditioned texturing and alpha-textured asset synthesis eliminate the need for dense modeling to create diverse scenes. VLM-based agents orchestrate the entire pipeline—ranging from asset selection, design, and arrangement to multi-modal immersion enhancement. Our method creates coherent worlds while maintaining real-time rendering on mobile platforms.

References

[1] S. Ahuja. BlenderMCP. `https://github.com/ahujasid/blender-mcp`, 2025. 1, 2
[2] O. Bar-Tal, L. Yariv, Y. Lipman, and T. Dekel. Multidiffusion: Fusing diffusion paths for controlled image generation. *arXiv preprint arXiv:2302.08113*, 2023. 4
[3] R. Birkl, D. Wofk, and M. Müller. Midas v3.1 – a model zoo for robust monocular relative depth estimation. *arXiv preprint arXiv:2307.14460*, 2023. 5
[4] Z. Chen, G. Wang, and Z. Liu. Scenedreamer: Unbounded 3d scene generation from 2d image collections. *IEEE transactions on pattern analysis and machine intelligence*, 2023. 2
[5] J. Chung, S. Lee, H. Nam, J. Lee, and K. M. Lee. Luciddreamer: Domain-free generation of 3d gaussian splatting scenes. *arXiv preprint arXiv:2311.13384*, 2023. 2
[6] D. Cohen-Bar, E. Richardson, G. Metzer, R. Giryes, and D. Cohen-Or. Set-the-scene: Global-local training for generating controllable nerf scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2920–2929, 2023. 2
[7] X. Décoret, F. Durand, F. X. Sillion, and J. Dorsey. Billboard clouds for extreme model simplification. In *ACM SIGGRAPH 2003 Papers*, pp. 689–696. 2003. 3, 4
[8] M. Deitke, R. Liu, M. Wallingford, H. Ngo, O. Michel, A. Kusupati, A. Fan, C. Laforte, V. Voleti, S. Y. Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 2
[9] P. Engstler, A. Shtedritski, I. Laina, C. Rupprecht, and A. Vedaldi. Syncity: Training-free generation of 3d worlds. 2025. 1, 2
[10] M. Feng, J. Liu, M. Cui, and X. Xie. Diffusion360: Seamless 360 degree panoramic image generation based on diffusion models. *arXiv preprint arXiv:2311.13141*, 2023. 5
[11] R. Fridman, A. Abecasis, Y. Kasten, and T. Dekel. Scenescape: Text-driven consistent scene generation. *Advances in Neural Information Processing Systems*, 36, 2024. 2
[12] R. Gao, A. Holynski, P. Henzler, A. Brussee, R. Martin-Brualla, P. Srinivasan, J. T. Barron, and B. Poole. Cat3d: Create anything in 3d with multi-view diffusion models. *arXiv preprint arXiv:2405.10314*, 2024. 2
[13] C. Gasch, J. M. Sotoca, M. Chover, I. Remolar, and C. Rebollo. Procedural modeling of plant ecosystems maximizing vegetation cover. *Multimedia Tools and Applications*, 81(12):16195–16217, 2022. 2
[14] H. Go, B. Park, J. Jang, J.-Y. Kim, S. Kwon, and C. Kim. Splatflow: Multi-view rectified flow model for 3d gaussian splatting synthesis. *arXiv preprint arXiv:2411.16443*, 2024. 2
[15] Z. Gu, R. Yan, J. Lu, P. Li, Z. Dou, C. Si, Z. Dong, Q. Liu, C. Lin, Z. Liu, W. Wang, and Y. Liu. Diffusion as shader: 3d-aware video diffusion for versatile video generation control. *arXiv preprint arXiv:2501.03847*, 2025. 9

[16] Z. Hao, A. Mallya, S. Belongie, and M.-Y. Liu. Gancraft: Unsupervised 3d neural rendering of minecraft worlds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14072–14082, 2021. 2

[17] L. Höllein, A. Cao, A. Owens, J. Johnson, and M. Nießner. Text2room: Extracting textured 3d meshes from 2d text-to-image models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7909–7920, 2023. 2

[18] Y. Hong, K. Zhang, J. Gu, S. Bi, Y. Zhou, D. Liu, F. Liu, K. Sunkavalli, T. Bui, and H. Tan. Lrm: Large reconstruction model for single image to 3d. *arXiv preprint arXiv:2311.04400*, 2023. 2

[19] Q. Huang, R. Zhang, K. Liu, M. Gong, H. Zhang, and H. Huang. Arcpro: Architectural programs for structured 3d abstraction of sparse points. *arXiv preprint arXiv:2503.02745*, 2025. 3

[20] Z. Huang, Y.-C. Guo, X. An, Y. Yang, Y. Li, Z.-X. Zou, D. Liang, X. Liu, Y.-P. Cao, and L. Sheng. Midi: Multi-instance diffusion for single image to 3d scene generation. *arXiv preprint arXiv:2412.03558*, 2024. 1, 2

[21] I. Inc. SpeedTree. https://store.speedtree.com/, 2024. 9

[22] J. Kratt, L. Coconu, T. Dapper, J. W. Schliep, P. Paar, and O. Deussen. Adaptive billboard clouds for botanical tree models. 2014. 3

[23] V. Kumaran, J. Rowe, B. Mott, and J. Lester. Scenecraft: Automating interactive narrative scene generation in digital games with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 19, pp. 86–96, 2023. 2

[24] J. J. Lee, B. Li, and B. Benes. Latent l-systems: transformer-based tree generator. *ACM Transactions on Graphics*, 43(1):1–16, 2023. 3

[25] B. Li, J. Kałużny, J. Klein, D. L. Michels, W. Pałubicki, B. Benes, and S. Pirk. Learning to reconstruct botanical trees from single images. *ACM Transactions on Graphics (TOG)*, 40(6):1–15, 2021. 3

[26] M. Li, D. M. Kaufman, V. G. Kim, J. Solomon, and A. Sheffer. Optcuts: Joint optimization of surface cuts and parameterization. *ACM transactions on graphics (TOG)*, 37(6):1–13, 2018. 3

[27] R. Li, P. Pan, B. Yang, D. Xu, S. Zhou, X. Zhang, Z. Li, A. Kadambi, Z. Wang, Z. Tu, and Z. Fan. 4k4dgen: Panoramic 4d generation at 4k resolution, 2024. 2

[28] H. Liang, J. Cao, V. Goel, G. Qian, S. Korolev, D. Terzopoulos, K. Plataniotis, S. Tulyakov, and J. Ren. Wonderland: Navigating 3d scenes from a single image. *arXiv preprint arXiv:2412.12091*, 2024. 2

[29] C. Lin and Y. Mu. Instructscene: Instruction-driven 3d indoor scene synthesis with semantic graph prior. *The International Conference on Learning Representations*, 2024. 2

[30] C. H. Lin, H.-Y. Lee, W. Menapace, M. Chai, A. Siarohin, M.-H. Yang, and S. Tulyakov. Infinicity: Infinite-scale city synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22808–22818, 2023. 2

[31] L. Ling, C.-H. Lin, T.-Y. Lin, Y. Ding, Y. Zeng, Y. Sheng, Y. Ge, M.-Y. Liu, A. Bera, and Z. Li. Scenethesis: A language and vision agentic framework for 3d scene generation. *arXiv preprint arXiv:2505.02836*, 2025. 1, 2

[32] M. Lipp, D. Scherzer, P. Wonka, and M. Wimmer. Interactive modeling of city layouts using layers of procedural content. In *Computer Graphics Forum*, vol. 30, pp. 345–354. Wiley Online Library, 2011. 2

[33] J.-H. Liu, S.-K. Zhang, C. Zhang, and S.-H. Zhang. Controllable procedural generation of landscapes. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 6394–6403, 2024. 2

[34] S. Liu, Z. Ferguson, A. Jacobson, and Y. I. Gingold. Seamless: seam erasure and seam-aware decoupling of shape from mesh resolution. *ACM Trans. Graph.*, 36(6):216–1, 2017. 3

[35] X. Liu, C.-K. Tang, and Y.-W. Tai. Worldcraft: Photo-realistic 3d world creation and customization via llm agents. *arXiv preprint arXiv:2502.15601*, 2025. 1, 2

[36] C. Meng, Y. He, Y. Song, J. Song, J. Wu, J.-Y. Zhu, and S. Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021. 4

[37] Q. Meng, L. Li, M. Nießner, and A. Dai. Lt3sd: Latent trees for 3d scene diffusion. *arXiv preprint arXiv:2409.08215*, 2024. 2

[38] Meshy. Meshy ai - the no. 1 ai 3d model generator for creators. https://www.meshy.ai/, 2025. 8

[39] B. M. Öcal, M. Tatarchenko, S. Karaoğlu, and T. Gevers. Sceneteller: Language-to-3d scene generation. In *European Conference on Computer Vision*, pp. 362–378. Springer, 2024. 2

[40] Y. I. Parish and P. Müller. Procedural modeling of cities. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pp. 301–308, 2001. 2

[41] S. R. K. Perla, Y. Wang, A. Mahdavi-Amiri, and H. Zhang. Easi-tex: Edge-aware mesh texturing from single image. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 43(4), article no. 40, 2024. doi: 10.1145/3658222 8

[42] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. In *The Twelfth International Conference on Learning Representations*, 2023. 3, 5

[43] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021. 5

[44] A. Raistrick, L. Lipson, Z. Ma, L. Mei, M. Wang, Y. Zuo, K. Kayan, H. Wen, B. Han, Y. Wang, et al. Infinite photorealistic worlds using procedural generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12630–12641, 2023. 2, 6, 7

[45] A. Raistrick, L. Lipson, Z. Ma, L. Mei, M. Wang, Y. Zuo, K. Kayan, H. Wen, B. Han, Y. Wang, A. Newell, H. Law, A. Goyal, K. Yang, and J. Deng. Infinite photorealistic worlds using procedural generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12630–12641, 2023. 2, 5, 8

[46] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. 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. 2, 3

[47] C. Schuhmann. CLIP+MLP Aesthetic Score Predictor. https://github.com/christophschuhmann/improved-aesthetic-predictor, 2023. 5

[48] C. Sun, J. Han, W. Deng, X. Wang, Z. Qin, and S. Gould. 3d-gpt: Procedural 3d modeling with large language models. *arXiv preprint arXiv:2310.12945*, 2023. 2

[49] F.-Y. Sun, W. Liu, S. Gu, D. Lim, G. Bhat, F. Tombari, M. Li, N. Haber, and J. Wu. Layoutvlm: Differentiable optimization of 3d layout via vision-language models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 2

[50] W. Sun, S. Chen, F. Liu, Z. Chen, Y. Duan, J. Zhang, and Y. Wang. Dimensionx: Create any 3d and 4d scenes from a single image with controllable video diffusion. *arXiv preprint arXiv:2411.04928*, 2024. 2

[51] C. Ton Roosendaal, Blender Foundation. Blender. https://www.blender.org/, 2024. 5

[52] H. Wang, X. Xiang, Y. Fan, and J.-H. Xue. Customizing 360-degree panoramas through text-to-image diffusion models. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 4933–4943, 2024. 2

[53] H. Wu, Z. Zhang, W. Zhang, C. Chen, C. Li, L. Liao, A. Wang, E. Zhang, W. Sun, Q. Yan, X. Min, G. Zhai, and W. Lin. Q-align: Teaching lmms for visual scoring via discrete text-defined levels. *arXiv preprint arXiv:2312.17090*, 2023. 5, 8

[54] Z. Wu, Y. Li, H. Yan, T. Shang, W. Sun, S. Wang, R. Cui, W. Liu, H. Sato, H. Li, et al. Blockfusion: Expandable 3d scene generation using latent tri-plane extrapolation. *ACM Transactions on Graphics (TOG)*, 43(4):1–17, 2024. 2

[55] J. Xiang, Z. Lv, S. Xu, Y. Deng, R. Wang, B. Zhang, D. Chen, X. Tong, and J. Yang. Structured 3d latents for scalable and versatile 3d generation. *arXiv preprint arXiv:2412.01506*, 2024. 2

[56] H. Xie, Z. Chen, F. Hong, and Z. Liu. Citydreamer: Compositional generative model of unbounded 3d cities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9666–9675, 2024. 2

[57] Y. Xu, Y. Ng, Y. Wang, I. Sa, Y. Duan, Y. Li, P. Ji, and H. Li. Sketch2scene: Automatic generation of interactive 3d game scenes from user's casual sketches. *arXiv preprint arXiv:2408.04567*, 2024. 2

[58] J. Yang, S. Yang, A. W. Gupta, R. Han, L. Fei-Fei, and S. Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. *arXiv preprint arXiv:2412.14171*, 2024. 4

[59] L. Yang, B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao. Depth anything v2. *arXiv:2406.09414*, 2024. 3, 4, 5

[60] S. Yang, J. Tan, M. Zhang, T. Wu, Y. Li, G. Wetzstein, Z. Liu, and D. Lin. Layerpano3d: Layered 3d panorama for hyper-immersive scene generation. *arXiv preprint arXiv:2408.13252*, 2024. 1, 2, 7

[61] S. Yang, J. Tan, M. Zhang, T. Wu, Y. Li, G. Wetzstein, Z. Liu, and D. Lin. Layerpano3d: Layered 3d panorama for hyper-immersive scene generation. *arXiv preprint arXiv:2408.13252*, 2024. 5, 6

[62] Y. Yang, F.-Y. Sun, L. Weihs, E. VanderBilt, A. Herrasti, W. Han, J. Wu,

N. Haber, R. Krishna, L. Liu, et al. Holodeck: Language guided generation of 3d embodied ai environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16227–16237, 2024. 2

[63] J. Yao, X. Wang, S. Yang, and B. Wang. Vitmatte: Boosting image matting with pre-trained plain vision transformers. *Information Fusion*, 103:102091, 2024. 5

[64] K. Yao, L. Zhang, X. Yan, Y. Zeng, Q. Zhang, L. Xu, W. Yang, J. Gu, and J. Yu. Cast: Component-aligned 3d scene reconstruction from an rgb image. *arXiv preprint arXiv:2502.12894*, 2025. 1

[65] H.-X. Yu, H. Duan, C. Herrmann, W. T. Freeman, and J. Wu. Wonderworld: Interactive 3d scene generation from a single image. In *CVPR*, 2025. 1, 2, 5, 6, 7

[66] H.-X. Yu, H. Duan, J. Hur, K. Sargent, M. Rubinstein, W. T. Freeman, F. Cole, D. Sun, N. Snavely, J. Wu, et al. Wonderjourney: Going from anywhere to everywhere. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6658–6667, 2024. 2

[67] X. Yu, M. Xu, Y. Zhang, H. Liu, C. Ye, Y. Wu, Z. Yan, C. Zhu, Z. Xiong, T. Liang, et al. Mvimgnet: A large-scale dataset of multi-view images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[68] C. Zhang, Q. Wu, C. C. Gambardella, X. Huang, D. Phung, W. Ouyang, and J. Cai. Taming stable diffusion for text to 360 {\deg} panorama image generation. *arXiv preprint arXiv:2404.07949*, 2024. 2

[69] C. Zhang, Q. Wu, C. C. Gambardella, X. Huang, D. Phung, W. Ouyang, and J. Cai. Taming stable diffusion for text to 360 panorama image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6347–6357, 2024. 5

[70] J. Zhang, X. Li, Z. Wan, C. Wang, and J. Liao. Text2nerf: Text-driven 3d scene generation with neural radiance fields. *IEEE Transactions on Visualization and Computer Graphics*, 2024. 2

[71] J. Zhang, C.-b. Wang, H. Qin, Y. Chen, and Y. Gao. Procedural modeling of rivers from single image toward natural scene production. *The Visual Computer*, 35:223–237, 2019. 2

[72] L. Zhang and M. Agrawala. Transparent image layer diffusion using latent transparency. In *ACM Transactions on Graphics (SIGGRAPH 2024)*, vol. 43, July 2024. 5, 9

[73] L. Zhang, A. Rao, and M. Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3836–3847, 2023. 3, 5

[74] L. Zhang, Z. Wang, Q. Zhang, Q. Qiu, A. Pang, H. Jiang, W. Yang, L. Xu, and J. Yu. Clay: A controllable large-scale generative model for creating high-quality 3d assets. *ACM Transactions on Graphics (TOG)*, 43(4):1–20, 2024. 2

[75] Q. Zhang, C. Wang, A. Siarohin, P. Zhuang, Y. Xu, C. Yang, D. Lin, B. Dai, B. Zhou, S. Tulyakov, and H.-Y. Lee. SceneWiz3D: Towards text-guided 3D scene composition. In *arXiv*, 2023. 2

[76] R. Zhang, S. Pan, C. Lv, M. Gong, and H. Huang. Architectural co-lod generation. *ACM Transactions on Graphics (TOG)*, 43(6):1–16, 2024. 3

[77] H. Zhou, X. Cheng, W. Yu, Y. Tian, and L. Yuan. Holodreamer: Holistic 3d panoramic world generation from text descriptions. *arXiv preprint arXiv:2407.15187*, 2024. 2

[78] M. Zhou, Y. Wang, J. Hou, C. Luo, Z. Zhang, and J. Peng. Scenex: Procedural controllable large-scale scene generation via large-language models. *arXiv preprint arXiv:2403.15698*, 2024. 1, 2

[79] S. Zhou, Z. Fan, D. Xu, H. Chang, P. Chari, T. Bharadwaj, S. You, Z. Wang, and A. Kadambi. Dreamscene360: Unconstrained text-to-3d scene generation with panoramic gaussian splatting. In *European Conference on Computer Vision*, pp. 324–342. Springer, 2025. 1, 2, 5, 6, 7, 9

[80] X. Zhou, X. Ran, Y. Xiong, J. He, Z. Lin, Y. Wang, D. Sun, and M.-H. Yang. Gala3d: Towards text-to-3d complex scene generation via layout-guided generative gaussian splatting. *arXiv preprint arXiv:2402.07207*, 2024. 2

[81] J. Zhuang, Y. Zeng, W. Liu, C. Yuan, and K. Chen. A task is worth one word: Learning with task prompts for high-quality versatile image inpainting, 2023. 5

[82] Z.-X. Zou, Z. Yu, Y.-C. Guo, Y. Li, D. Liang, Y.-P. Cao, and S.-H. Zhang. Triplane meets gaussian splatting: Fast and generalizable single-view 3d reconstruction with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10324–10335, 2024. 2