

WorldGen: From Text to Traversable and Interactive 3D Worlds

Dilin Wang^{1→} Hyunyoung Jung¹ Tom Monnier¹ Kihyuk Sohn¹ Chuhan Zou¹ Xiaoyu Xiang¹
 Yu-Ying Yeh¹ Di Liu¹ Zixuan Huang¹ Thu Nguyen-Phuoc¹ Yuchen Fan¹ Sergiu Oprea¹
 Ziyang Wang¹ Roman Shapovalov¹ Nikolaos Sarafianos¹ Thibault Groueix¹ Antoine Toisoul¹
 Prithviraj Dhar¹ Xiao Chu¹ Minghao Chen^{1,2} Geon Yeong Park^{1,3} Rakesh Ranjan^{1→} Andrea Vedaldi^{1→}
¹Reality Labs, Meta ²VGG, University of Oxford ³KAIST → Project lead



Figure 1. **World snapshots generated by WorldGen.** Each scene consists of individually editable objects represented as fully textured 3D meshes. As explicit geometry, these worlds naturally support collision and navigation, allowing characters to climb, jump, and interact. The resulting assets are immediately deployable in game engines. We encourage readers to view the [videos](#) for a clearer sense of the scale, structure, and immersiveness of the generated worlds.

Abstract

We introduce WorldGen, a method for generating large, fully formed, navigable 3D worlds from a single text prompt. Existing approaches to 3D scene generation often trade off scene diversity, completeness, and correctness in different ways. We push this envelope by producing large scenes explicitly decomposed into individual, high-quality 3D meshes, making them compatible with standard game engines. Our approach first uses a language-driven procedural generator to lay out the scene’s basic volumes and navigable regions. An image generator then estab-

lishes the scene’s theme, style, and details. Next, we obtain a high-quality, compositional 3D reconstruction of the planned scene. This step first uses an image-to-3D model to perform a holistic reconstruction that implicitly determines the shape and location of all scene objects, accounting for context and navigability. The reconstruction is then decomposed into individual entities, which are regenerated at higher resolution, synthesizing additional details with guidance from the image generator. We ablate key design choices and compare qualitatively against existing scene generators, showing that our design addresses their gaps.

1. Introduction

Creating 3D content for games and other interactive experiences requires significant effort and skill, particularly for 3D modeling of entire game worlds. 3D generative AI promises to dramatically simplify the creation of such content, allowing even laypeople to type a text prompt and obtain high-quality, textured 3D meshes in response [36, 38, 86, 88, 92]. However, these models are usually limited to creating individual 3D objects. Designing complete game levels that are also functional (e.g., traversable end-to-end by a character without getting stuck) remains challenging and largely an open problem.

In this work, we present *WorldGen*, an end-to-end pipeline that generates complete 3D worlds from a single text prompt (Fig. 1). We address several open challenges that have, so far, limited the quality, extent, and functionality of automatically generated 3D worlds.

A major challenge is the lack of large-scale datasets of 3D scenes for training world generators. Inspired by image-to-3D models, we use a powerful image generator to imagine the scene structure and content, and then reconstruct the 3D scene using this image as guidance. However, these models are not designed to ensure that the resulting scenes are fully functional—for example, traversable. We therefore guide image generation with a layout produced by a procedural generator (PG), which can easily enforce such structural constraints. We further condition these PGs on the user prompt via large language models (LLMs).

While PGs are narrow, in our case they define the main volumes of the scene and its navigable surfaces, captured by a so-called navmesh. This rough layout is provided to the image generator to establish the scene’s details, theme, and aesthetics, ensuring diversity. This image and the PG output thus form a *scene plan*.

Our second contribution is extracting a compositional, high-resolution 3D reconstruction from the scene plan. One approach is to segment and reconstruct objects, but the lack of contextual reasoning tends to produce poor results; for instance, objects may not fit together, particularly in occluded regions. Instead, we employ a *holistic* 3D latent diffusion model [38, 86, 88]. By conditioning this model on the navmesh, we ensure structural coherence and navigability, even in regions not visible in the guidance image.

Holistic reconstruction is a simple yet effective way of reconstructing objects in context. However, on its own it does not yield a production-quality 3D world, as the resolution of the resulting geometry is insufficient. Moreover, representing the scene as a single mesh is inconvenient for editing, animation, and efficiency in applications. To improve quality and modularity, we build on recent advances in compositional 3D generation [9] to decompose the reconstructed scene into its constituent objects. We propose a variant of this model to efficiently handle the large number

of components typically present in complete scenes. Then, we enhance the individual components to achieve higher visual fidelity while preserving the overall structure of the world. First, an image generator produces a high-resolution view of each object, enriching it with fine-grained visual details. Second, we use a mesh enhancer [57], fine-tuned from our image-to-3D model, to regenerate each object’s geometry and texture from the low-quality geometry and the visual cues in the generated images.

Because prior methods for 3D world generation differ substantially in their assumptions, representations, and outputs, we evaluate WorldGen through both ablations and targeted comparisons. We analyze key design choices qualitatively and quantitatively, and we compare against state-of-the-art image-to-3D methods to motivate our design. We also include qualitative comparisons with prior scene-generation systems, including Marble [76]. Our **main contributions** are as follows:

- We present WorldGen, an end-to-end pipeline that generates functional, traversable 3D worlds from text by synergizing LLM-driven procedural layouts with powerful image-to-3D priors.
- We introduce a navmesh-guided holistic reconstruction strategy that conditions 3D latent diffusion on structural constraints, ensuring global coherence and navigability even in occluded regions.
- We propose a compositional refinement framework that decomposes the holistic mesh into constituent objects for individual geometry and texture enhancement, yielding high-fidelity, editable, and game-ready assets.

2. Related Work

Many works have considered the problem of generating 3D worlds, often under very different assumptions and goals, making the resulting classes of approaches comparable only qualitatively. A popular line of work is *view-based* generation. Broadly speaking, the idea is to use 2D image or video generators to create views of the scene incrementally while also reconstructing its 3D structure. An early example with implicit geometry is SynSin [75]. Explicit geometry is recovered in works such as PixelSynth [65], CompNVS [39], Text2NeRF [87], and Text2Room [23]. Many follow-ups have extended these ideas in various ways [1, 7, 8, 11, 12, 15, 16, 19, 20, 22, 24, 28, 29, 33–35, 37, 43, 44, 46, 49, 50, 54–56, 61, 66–68, 70, 71, 73, 74, 78, 80, 82, 83, 85, 89–91, 94, 96]. However, the incremental and monolithic nature of the resulting 3D scenes limits their quality and applicability.

Others focus instead on *compositional scene generation*, arranging objects that are either retrieved from a database [41, 84] or generated ad hoc [42, 52, 95] under layout constraints derived from physics simulation or guided by LLMs. Some arrange objects to match a given im-

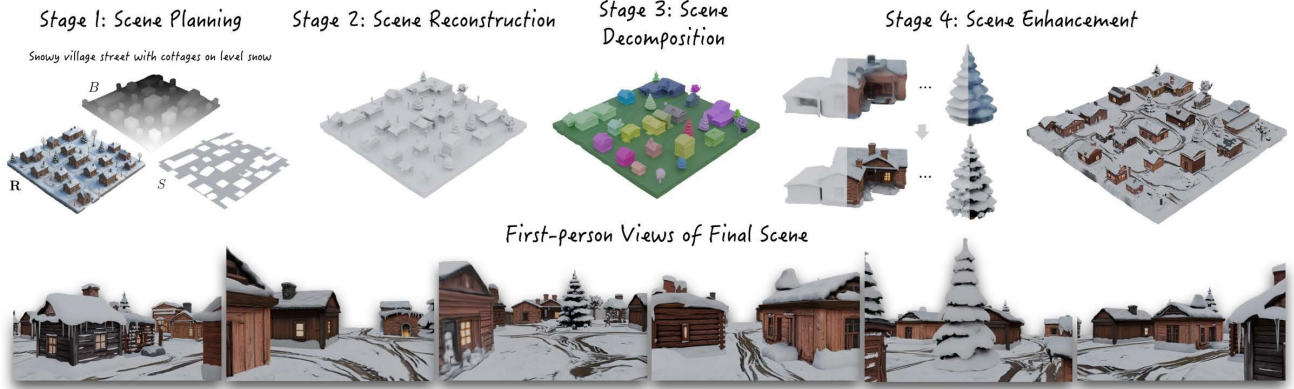


Figure 2. **WorldGen overview.** The pipeline has four stages: (1) Scene Planning, (2) Scene Reconstruction, (3) Scene Decomposition, and (4) Scene Enhancement. The final scene is traversable and highly detailed.

age [13, 18, 21, 26, 31, 52], which makes them closer to our approach. Usually, these methods do not scale beyond small arrangements of a handful of objects.

Procedural generators build worlds from hand-crafted procedures or rule systems. The computer graphics community has developed many such generators. Some, like InfiniGen and InfiniGen Indoors [43, 63], have used them to create computer vision datasets. Others, like SceneCraft [25] and SceneX [93], translate text or graphs into executable programs that build scenes using LLM agents. A limitation of procedural generation is that, because it is based on formal systems such as code and rules, it is narrow in practice. Our method leverages procedural generation to create the initial layout structure of the scene, and then relies on 2D and 3D generators to add diversity and detail.

Recent progress in 3D generation largely stems from *3D latent-space models* extended to scene generation. SceneFactor [6] generates *multi-room* indoor scenes. BlockFusion [77] and LT3SD [51] grow worlds *tile-by-tile*. NuiScene [32] scales VecSet models [86] from objects to *scenes* for unbounded outdoor generation, but without object structure or textures. By using latent spaces, these models generate complete scene meshes but, due to the limited availability of 3D scene data for training, lack diversity. Our approach compensates for this lack of training data by building on text-to-image generators.

SynCity [14] builds worlds using pre-trained text-to-image and image-to-3D models, *tile-by-tile*. Like us, they leverage an image generator to promote diversity; however, their results lack semantic compositionality, exhibit tile-like artifacts, and are comparatively low resolution. Some works have attempted to *bridge view-based and 3D latent-space* methods [2, 3, 30, 40, 64, 72, 81], but struggle to generate large, complete scenes. Finally, interactive video generators like Genie [58] promise to replace the entire gaming stack with real-time interactive video generators, subsuming also world generation. While compelling, this solution remains a distant future. Our approach generates 3D assets that are

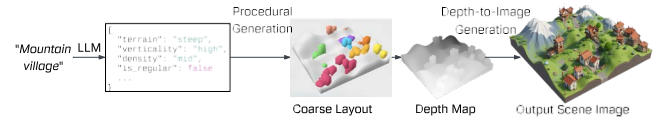


Figure 3. **3D Layout Generation.** An LLM parses the input prompt into structured parameters (JSON) to drive a procedural generator, producing a coarse 3D blockout. This blockout is then rendered to depth, which conditions the generation of the final scene reference image.

compatible with *today's* hardware, tools, and workflows.

3. WorldGen

We describe the stages of WorldGen: scene planning, reconstruction, decomposition, and enhancement.

3.1. Stage I: Scene Planning

The goal of the scene planning stage is to transform a user prompt y into a rough but functionally correct scene plan $L = (B, R, S)$, consisting of a *blockout layout* B , a *reference image* R , and a *navigational mesh* (navmesh) S [69]. An overview is illustrated in Fig. 3. This stage ensures the scene layout remains functional and traversable.

3.1.1. Procedural Blockout Generation

We generate the blockout using a text-conditioned procedural generation (PG) pipeline inspired by [48, 62]. Our PG system includes a language interface: an LLM converts the text prompt y into a JSON specification of terrain and layout parameters, which directly configure the PG pipeline to produce a blockout B aligned with the user's intent.

In more detail, our PG pipeline constructs the blockout in three steps: *terrain generation*, *spatial partitioning*, and *hierarchical asset placement*. First, *terrain generation* constructs a base landscape that defines the large-scale geometry of the scene—such as elevation, slopes, and flat regions—providing a base for where structures and traversal

paths can exist. Next, *spatial partitioning* divides the terrain into distinct regions that serve different scene purposes (open areas, clusters of structures, or transition zones). This step provides a high-level organizational layout, ensuring that the scene has variation in density and structure while maintaining navigability. Finally, *asset placement* populates each region with 3D assets in multiple passes: large landmark assets are placed first to establish structure and focal points, followed by smaller objects and decorative elements that add realism and detail. This multi-level placement strategy produces consistent yet varied scenes, maintaining both functional organization and visual diversity.

3.1.2. Navmesh and Reference Image

The resulting blockout B above is a 3D mesh composed of simple primitives (ground plane and boxes) that encode the essential geometry of the scene. It serves as an editable structural scaffold from which we subsequently derive the navmesh S and reference image R .

The navmesh S is extracted directly from the solid blockout geometry B using a standard algorithm such as Recast [53]. To generate the reference image R , we render the blockout geometry B into an isometrically projected depth map, which is then used as a condition to our depth-conditioned image generator. We adopt a camera elevation of approximately 45° to maximize visible coverage of the scene within the frame. Additionally, to reduce the appearance of overly rectilinear shapes, we apply a small Gaussian perturbation to the non-terrain depth values, scaled proportionally to depth, which encourages more natural, visually varied structural outlines in the resulting image.

3.2. Stage II: Scene Reconstruction

Given the scene plan $L = (B, R, S)$ obtained in Stage I, Stage II generates a 3D scene mesh M that faithfully aligns with the plan. In particular, the mesh should respect the navigable regions encoded in the navmesh S while also matching the overall composition and appearance suggested by the reference image R .

3.2.1. Navmesh-conditioned scene generation

Navmesh-conditioned image-to-3D must be learned from triplets (M, R, S) comprising the ground-truth scene mesh and the corresponding reference image and navmesh. Unfortunately, there are no available datasets of 3D scenes that can be used for this purpose. We address this problem with a two-stage training strategy. First, we pre-train an image-to-3D model on a broad set of generic object categories, allowing the model to learn a robust 3D reconstruction prior. We then fine-tune the model on a dataset of scene triplets that we have collected. Our image-to-3D model follows the VecSet design [10, 86] that uses a latent 3D diffusion transformer (see supp. mat.). Below, we explain how this base architecture is extended with navmesh conditioning.

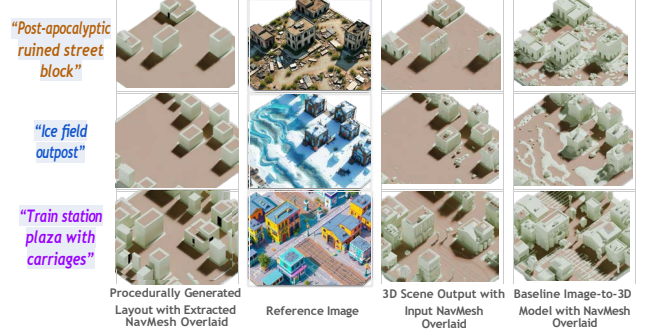


Figure 4. **Navmesh-conditioned scene generation.** The procedural layout with extracted navmesh, the reference image, the final scene from our finetuned model and from the baseline image-to-3D model, and the final scene with the navmesh overlay.

Navmesh Encoder and Conditioning. We condition the 3D diffusion transformer with the navmesh using cross-attention. To tokenize the navmesh S , we use an encoder similar to the VecSet [86] architecture. First, we uniformly sample points from the surface of S to form a dense point cloud $P \uparrow \mathbb{R}^{M \times 3}$ and then downsample it to a smaller set $P' = \text{FPS}(P | K) \uparrow \mathbb{R}^{K \times 3}$ using farthest point sampling. Both point sets are independently embedded using a coordinate positional encoder that maps 3D coordinates into D -dimensional feature vectors. Next, the encoded sparse points attend to the dense points through a cross-attention layer, allowing the sparse tokens to capture fine-grained geometric details of the navmesh. The resulting sparse navmesh embeddings are then injected into the image-to-3D denoising diffusion transformer backbone through additional cross-attention layers.

Training Strategy. We start from the pretrained weights and fine-tune them using our data. We compare updating only the weights of the newly added cross-attention layers against fine-tuning the entire transformer model end-to-end. Empirically, the former approach results in a 25% degradation in navmesh alignment, as measured by Chamfer distance. This suggests that producing a mesh that aligns faithfully with the navmesh requires joint adaptation across the entire generator network, as the task involves non-trivial geometric alignment and scene-level completion beyond the capacity of the newly introduced conditioning layers alone.

Data Normalization. Our VecSet-based image-to-3D model operates in a normalized space, where all meshes are rescaled within a $[\downarrow 1, 1]^3$ cube. During training, we rescale each navmesh using the scale factor computed from its corresponding scene mesh to ensure spatial alignment. We also jointly translate the navmesh and the scene mesh so that the navmesh ground plane is centered at $(0, 0, 0)$. This trick provides a stable spatial reference, leading to better alignment between the conditioned navmesh and the generated

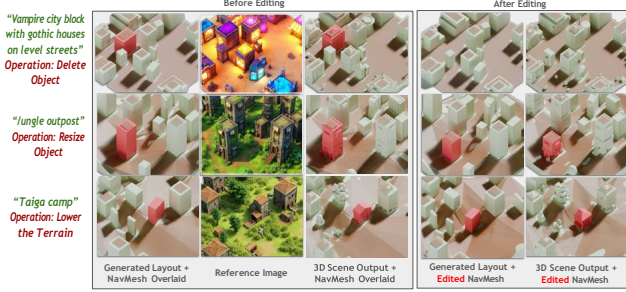


Figure 5. **Navmesh editing capabilities.** Each row shows results after manually editing the procedurally generated layout and its navigable path. Left to right: original layout with navmesh overlay, reference image, generated 3D scene with input navmesh overlay, manually edited navmesh, and the final 3D scene conditioned on the edited navmesh.

scene meshes. At inference time, where ground-truth scene meshes are unavailable, we normalize the navmesh using the scale derived from the procedurally generated blockout B and apply the translation to the navmesh.

3.2.2. Scene Reconstruction Results

As shown in Fig. 4, our navmesh-conditioned model produces smoother and cleaner terrain surfaces that adhere closely to the input navigation constraints—key for ensuring smooth traversability in the final scene. Although some fine terrain details may be visually approximated through texturing, accurate geometric grounding is critical for gameplay functionality. Moreover, the navmesh-conditioned model achieves better alignment with the reference image in structural regions such as buildings, particularly in more complex scenes.

To quantify this improvement, we compare our fine-tuned model with a baseline image-to-3D model and other state-of-the-art image-to-3D models, denoted as Models A, B, and C, on a curated scene benchmark, as detailed in Tab. 1. This benchmark comprises 50 procedurally generated scenes, each featuring terrain with moderate verticality and 10–30 densely sampled objects. We extract navmesh from our generated scenes and those produced by the baseline models. After normalizing all geometries to a $[\downarrow 1, 1]^3$ cube, we extract the navmesh and align it to the ground-truth navmesh using ICP. We then compute the Chamfer distance between these aligned meshes. Our navmesh-conditioned model achieves Chamfer distances that are 40–50% lower compared to the baselines. The quantitative comparisons clearly demonstrate stronger spatial alignment with the input navigation conditions. More detailed evaluations are provided in the supplementary material.

In Fig. 5, we further evaluate the model’s generalization under slight misalignments between the navmesh and the reference image. This scenario is particularly important, as editing a 2D reference image to accurately reflect spatial

Table 1. **Quantitative evaluation of navmesh alignment**, measured via Chamfer distance (CD) between the input navmesh and the navmesh extracted from the generated scene (lower is better). Baseline* denotes baseline shape generator trained on our curated dataset of scene triplets.

Model	Model A	Model B	Model C	Baseline	Baseline*	Ours
NavMesh CD	0.038	0.049	0.048	0.042	0.038	0.022

intent is often cumbersome—or even infeasible—due to inherent ambiguities in projecting 3D structure onto a single view. In contrast, navmesh edits can be easily and precisely specified in 3D by modifying the layout directly. For example, in the “Jungle outpost” scene, we remove a structure to create additional walkable space; in “Vampire city block”, we reduce the height of a building; and in “Taiga camp”, we slightly lower the terrain to form a shallow concave dip, adding subtle variation to the landscape. These examples illustrate that our navmesh-conditioned model does not merely replicate the reference image but instead learns to reason about spatial organization and navigability, maintaining structural and stylistic coherence even when the layout and image conditions diverge.

3.3. Stage III: Scene Decomposition

The coarse, monolithic textured mesh M produced by Stage II represents the entire scene and fuses all objects into a single geometry. This makes it difficult to edit or refine individual assets. To address this limitation, we first decompose M into semantically meaningful objects and parts, and then enhance them individually in the subsequent stage.

Our approach builds upon the AutoPartGen [9] model, which decomposes a mesh into parts sequentially in an autoregressive manner, where each part is generated conditioned on the holistic scene mesh and all previously generated parts. However, two key limitations make the original AutoPartGen unsuitable for large-scale scene decomposition. First, its autoregressive nature leads to slow inference, making it computationally expensive for complex scenes with many objects and parts. Second, the model was trained primarily on generic object-level datasets and therefore fails to generalize effectively to full scenes containing diverse, spatially interacting assets.

3.3.1. Accelerating AutoPartGen for Scenes

While AutoPartGen generates parts in a fixed lexicographical order (z - x - y), we instead generate parts according to their *connectivity degree*, defined as the number of other parts each collides with. By generating parts in decreasing order of connectivity, we prioritize *pivot parts* that serve as structural anchors connecting many other components. Once pivots are generated, the remaining parts can be efficiently recovered through spatial connectivity analysis of the residual geometry. For example, in outdoor scenes with

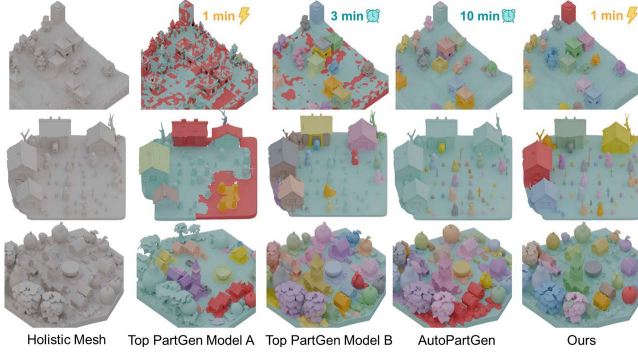


Figure 6. **Scene decomposition comparison.** Our method produces cleaner decompositions with less noise than prior methods while retaining fast inference.

many buildings and trees, the ground often exhibits the highest connectivity degree. Once the ground is extracted, the decomposition of other objects becomes easier through connected-component analysis.

To support this strategy, we extend AutoPartGen to explicitly generate the *remainder geometry* as a special part. We introduce a binary *flag token* that, when activated, instructs the model to produce all remaining geometry in a single forward pass. In practice, we use a five-step schedule: the model first generates four pivot parts, followed by the remainder part, which is further decomposed via connected-component analysis. This design accelerates decomposition significantly, even for complex scenes, and reduces overall generation time from ten to about one minute.

3.3.2. Scene Decomposition Data

We find it important to fine-tune AutoPartGen on scene-level data. However, there is no readily available dataset of 3D scenes with part annotations. We address this gap by creating a dataset of compositional 3D scenes.

First, we mine 3D scenes from a large internal 3D asset repository. To do this, we use a vision-language model (VLM) and identify assets that represent whole scenes. Specifically, we prompt the VLM to assess rendered images and determine whether they exhibit the characteristics of multi-object environments (e.g., sufficient object diversity, plausible spatial layout, and visible ground context).

Once initial scene-like assets are selected, we apply heuristics to convert the raw geometry into assets with meaningful object and part decompositions. Our processing pipeline combines connectivity-based part splitting with ground-aware reasoning. The pipeline includes four major steps: (1) we detect topologically connected components after vertex welding as minimal parts, (2) we detect the ground and merge thin overlays (e.g., traffic lines) onto the ground as a standalone part, (3) we de-duplicate and iteratively merge small parts into their nearest spatial neighbors while ensuring the ground remains separate, and (4) we fil-

Table 2. **Quantitative evaluation of scene decomposition methods.** Our model achieves significantly better results than previous state-of-the-art methods across all metrics.

Model	Chamfer	F-score@0.01	F-score@0.05	Time
Top PartGen Model A	0.171	0.090	0.443	1 min
Top PartGen Model B	0.136	0.155	0.633	3 min
AutoPartGen	0.144	0.281	0.683	10 min
Ours	0.061	0.322	0.853	1 min

ter decomposed assets with several quality constraints, including part count, part imbalance, and ground confidence.

3.3.3. Decomposition Results

To evaluate the decomposition capability of our model, we perform qualitative and quantitative comparisons with other state-of-the-art 3D part generation methods, denoted as Top PartGen Model A and Model B. Existing part generation models often fail to generalize to complex scenes (see Fig. 6). For instance, Top PartGen Model A struggles to produce stable decompositions for large outdoor scenes. Top PartGen Model B erroneously segments terrain regions, sometimes fragmenting buildings into excessively small components or merging ground and object geometry into a single part. AutoPartGen [9] also occasionally fails to decompose major objects, and its high latency makes it impractical for handling large scenes.

We further curate a diverse synthetic evaluation dataset by placing objects on various terrains. The holistic mesh is obtained by water-tightening all objects and terrains jointly. This provides us with pairs of holistic meshes and ground-truth part annotations. We follow an evaluation protocol of prior work [9, 45]: for each ground-truth part, we find its nearest neighbor in predictions and compute their Chamfer distance and F-score at various thresholds. We show results in Tab. 2, where our model outperforms other methods while maintaining the top inference speed.

3.4. Stage IV: Scene Enhancement

This final stage enhances the geometry quality of each object produced in the previous stage and generates its final texture. This process involves three steps. First, we generate a high-resolution image for each object to recover missing visual details, as the scene-level reference image provides limited resolution and coverage for each object. Second, we refine the object geometry using a mesh-enhancement model [57] that improves shape fidelity while remaining aligned with the original geometry so that enhanced objects still fit correctly in the scene. Finally, we generate a high-quality texture for each object, guided by both the refined geometry and the high-resolution per-object image.

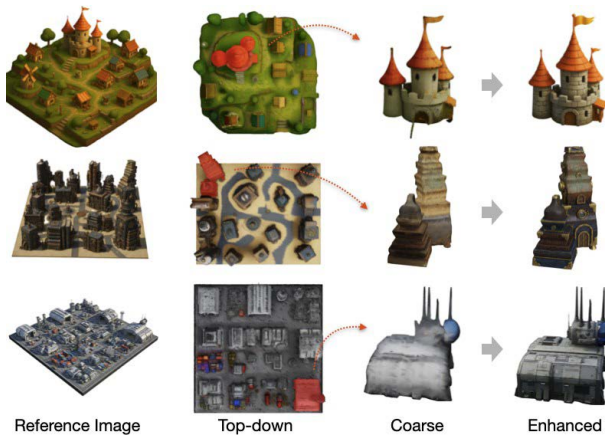


Figure 7. **Per-object image enhancement.** The reference image, the top-down view with target object highlighted with orange and the coarse textured rendering are sent to an LLM-VLM that outputs the final per-object enhanced image.

3.4.1. Per-Object Image Enhancement

We first apply TRELLIS [79] to texture the entire coarse mesh, giving each decomposed object an initial coarse texture. Although TRELLIS typically produces lower-resolution texture than multi-view generation methods, it operates directly in 3D and is therefore more robust to self-occlusion. At this stage, the texture does not need to be high-quality; it only needs to capture broad material and style cues for subsequent enhancement. Because TRELLIS is trained mainly on object-level assets and generalizes less effectively to full scenes, we retrain it on our in-house dataset containing both object- and scene-scale data.

Our main idea is to leverage a large language-vision model (LLM-VLM) to enhance each object’s appearance by conditioning on its coarse textured rendering. The primary challenge is maintaining style consistency across all objects, since independently enhanced textures can easily drift in color, material, or artistic style. To provide the image generator with global scene context, for each object, we further render a top-down view of the textured scene and highlight the target object in orange, explicitly indicating its location within the overall scene. We also supply the global reference image to the LLM-VLM.

In Fig. 7, the left three columns show the inputs to the LLM-VLM model, and the rightmost column shows the enhanced object image. Because generative enhancement can occasionally hallucinate unwanted changes, we apply a verification step: the enhanced image is compared against the coarse rendering, and results that deviate too far in silhouette are rejected. We repeat the enhancement process until a valid, well-aligned result is obtained. See Fig. 8 for examples of the initial input, rejected cases, and final results after verification and iterative refinement.



Figure 8. **Object image verification.** The verification step rejects enhanced images that do not align with the coarse input.

3.4.2. Per-Object Mesh Enhancement

We employ a mesh refinement model [57] to enhance geometric detail using the high-fidelity object image as guidance. The refinement model extends our baseline image-to-3D architecture by additionally conditioning on the latents of the coarse mesh obtained from its VAE encoder. Additional details can be found in the supplementary material and the original paper [57].

3.4.3. Per-Object Texture Enhancement

We finally generate a high-resolution texture for each object based on its enhanced image and geometry. Following an established recipe for texture generation [4, 27], we fine-tune a pretrained text-to-image latent diffusion model to produce 3D-consistent multi-view renderings of the object conditioned on normal and position maps for the target views. We backproject the generated multi-view images to UV map to get the final texture image. We provide more technical details in the supplementary materials.

4. Results

WorldGen Result Overview. In Figs. 1 and 9 and the supplementary material, we provide a gallery of several scenes generated by WorldGen end-to-end from user prompts. Overall, our method is capable of generating interesting, diverse, and high-fidelity scenes that can be navigated freely by characters, and are thus suitable for use in game engines. Each scene contains multiple semantically consistent objects, coherent textures, and a valid navmesh that enables real-time exploration. Despite the multi-stage nature of our approach, the entire pipeline from a text prompt to a fully textured, navigable 3D scene completes in $\downarrow 5$ minutes with many submodules (e.g., object enhancement and texture generation) running in parallel across GPUs. This enables rapid prototyping of interactive worlds with minimal user intervention.

Qualitative Comparison. Prior scene-generation systems vary substantially in their assumptions, scene representations, and target outputs. We therefore focus on qualitative comparisons that highlight differences in the resulting worlds and their intended use cases.



Figure 9. **WorldGen results.** We use first-person views with a camera randomly sampled from the scene and neutral skyboxes.



Figure 10. **Comparison with two state-of-the-art commercial image-to-3D systems.** WorldGen generates scenes that are significantly more detailed than single-shot solutions.

We first compare against *image-to-3D generators*, including several commercially available systems that serve as strong baselines for single-image 3D generation. Although these methods produce impressive results for individual objects and small-scale scenes, they are primarily designed for single-shot reconstruction rather than compositional, large-scale, navigable environments. Their generated geometry and texture resolution remain insufficient for direct use in game engines. As shown in Fig. 10, these single-shot models produce outputs that lack the geometry and texture details required to support immersive, explorable 3D experiences.

We also compare with *Marble* [76], a recent system for 3D world generation from World Labs. Because the two methods differ in both inputs and scene representation, we use a qualitative setup that aligns their starting points as closely as possible. WorldGen takes a global layout and reference image as input and outputs explicit textured meshes, whereas Marble expands a scene outward from a single specified viewpoint using Gaussian splats.

For this comparison, we render a central view of our “medieval village” scene and use it as input to *Marble*. *Marble* achieves high visual fidelity near the conditioned viewpoint, particularly in appearance. As the camera moves farther from the initial view, however, the fidelity of the generated environment decreases noticeably (Fig. 11). In contrast, WorldGen is designed for broader traversable extent

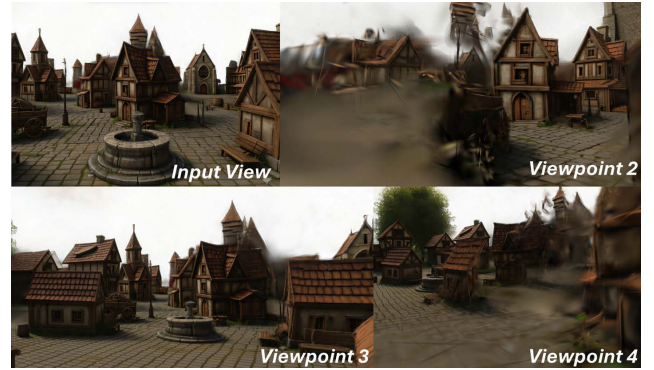


Figure 11. **Marble examples.** Example Marble generations viewed from different viewpoints.

and explicit geometry: our generated scenes span roughly $50 \leftrightarrow 50$ m, remain geometrically and stylistically consistent across the traversable area, and support free navigation and interaction.

Limitations. Our reliance on a single reference view currently restricts generation to bounded, single-story environments, excluding unbounded or multi-layered scenes. Additionally, the lack of asset instancing (i.e., reuse) may impact rendering efficiency in dense areas. Future work should address these gaps, for example via incremental generation [14] and material and geometry reuse. Further discussion is provided in the supplementary material.

5. Conclusions

We presented WorldGen, an end-to-end framework generating traversable, game-ready 3D worlds from text by unifying procedural planning with navmesh-guided synthesis. We believe that WorldGen demonstrates the potential of combining procedural reasoning with generative priors to democratize interactive 3D content creation.

References

- [1] *Ctrl-Room: Controllable Text-to-3D Room Meshes Generation with Layout Constraints*, 2025. 2
- [2] Sherwin Bahmani, Tianchang Shen, Jiawei Ren, Jiahui Huang, Yifeng Jiang, Haithem Turki, Andrea Tagliasacchi, David B. Lindell, Zan Gojic, Sanja Fidler, Huan Ling, Jun Gao, and Xuanchi Ren. Lyra: Generative 3D scene reconstruction via video diffusion model self-distillation. *arXiv*, 2509.19296, 2025. 3
- [3] Miguel A´ngel Bautista, Pengsheng Guo, Samira Abnar, Walter Talbott, Alexander Toshev, Zhuoyuan Chen, Laurent Dinh, Shuangfei Zhai, Hanlin Goh, Daniel Ulbricht, Afshin Dehghan, and Josh M. Susskind. GAUDI: A neural architect for immersive 3D scene generation. *arXiv.cs*, abs/2207.13751, 2022. 3
- [4] Raphael Bensadoun, Yanir Kleiman, Idan Azuri, Omri Harosh, Andrea Vedaldi, Natalia Neverova, and Oran Gafni. Meta 3D Texture Gen: Fast and consistent texture generation for 3D objects. *arXiv preprint*, 2024. 7, 3
- [5] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975. 1
- [6] Alexey Bokhovkin, Quan Meng, Shubham Tulsiani, and Angela Dai. SceneFactor: factored latent 3D diffusion for controllable 3d scene generation. *arXiv*, 2412.01801, 2024. 3
- [7] Shengqu Cai, Eric Ryan Chan, Songyou Peng, Mohamad Shahbazi, Anton Obukhov, Luc Van Gool, and Gordon Wetzstein. DiffDreamer: Towards consistent unsupervised single-view scene extrapolation with conditional diffusion models. In *Proc. ICCV*, 2023. 2
- [8] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J. Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *Proc. CVPR*, 2022. 2
- [9] Minghao Chen, Jianyuan Wang, Roman Shapovalov, Tom Monnier, Hyunyoung Jung, Dilin Wang, Rakesh Ranjan, Iro Laina, and Andrea Vedaldi. AutoPartGen: Autogressive 3D part generation and discovery. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2025. 2, 5, 6
- [10] Rui Chen, Jianfeng Zhang, Yixun Liang, Guan Luo, Weiyu Li, Jiarui Liu, Xiu Li, Xiaoxiao Long, Jiashi Feng, and Ping Tan. Dora: Sampling and benchmarking for 3d shape variational auto-encoders. In *Proc. CVPR*, pages 16251–16261, 2025. 4
- [11] Zilong Chen, Yikai Wang, Feng Wang, Zhengyi Wang, Fuchun Sun, and Huaping Liu. V3d: Video diffusion models are effective 3d generators. 2025. 2
- [12] Jaeyoung Chung, Suyoung Lee, Hyeonjin Nam, Jaerin Lee, and Kyoung Mu Lee. LucidDreamer: Domain-free generation of 3d gaussian splatting scenes. In *arXiv*, 2023. 2
- [13] Tianyuan Dai, Josiah Wong, Yunfan Jiang, Chen Wang, Cem Gokmen, Ruohan Zhang, Jiajun Wu, and Li Fei-Fei. Automated creation of digital cousins for robust policy learning. *arXiv*, 2410.07408, 2024. 3
- [14] Paul Engstler, Aleksandar Shtedritski, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. SynCity: Training-free generation of 3D cities. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2025. 3, 8, 6, 7
- [15] Paul Engstler, Andrea Vedaldi, Iro Laina, and Christian Rupprecht. Invisible stitch: Generating smooth 3D scenes with depth inpainting. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2025. 2
- [16] Rafail Fridman, Amit Abecasis, Yoni Kasten, and Tali Dekel. Scenescape: Text-driven consistent scene generation. *CoRR*, abs/2302.01133, 2023. 2
- [17] Henry Fuchs, Zvi M Kedem, and Bruce F Naylor. On visible surface generation by a priori tree structures. In *Proc. SIGGRAPH*, pages 124–133, 1980. 1
- [18] Daoyi Gao, Da’vid Rozenberszki, Stefan Leutenegger, and Angela Dai. DiffCAD: weakly-supervised probabilistic CAD model retrieval and alignment from an RGB image. *arXiv*, 2311.18610, 2024. 3
- [19] Ruiyuan Gao, Kai Chen, Zhihao Li, Lanqing Hong, Zhen-guo Li, and Qiang Xu. MagicDrive3D: controllable 3D generation for any-view rendering in street scenes. *arXiv*, 2405.14475, 2024. 2
- [20] Ruiqi Gao, Aleksander Holynski, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul Srinivasan, Jonathan T. Barron, and Ben Poole. CAT3D: create anything in 3d with multi-view diffusion models. *arXiv*, 2405.10314, 2024. 2
- [21] Can Gu´meli, Angela Dai, and Matthias Nießner. ROCA: Robust CAD model retrieval and alignment from a single image. In *Proc. CVPR*, 2022. 3
- [22] Jiazhe Guo, Yikang Ding, Xiwu Chen, Shuo Chen, Bohan Li, Yingshuang Zou, Xiaoyang Lyu, Feiyang Tan, Xiaojuan Qi, Zhiheng Li, and Hao Zhao. DiST-4D: disentangled spatiotemporal diffusion with metric depth for 4D driving scene generation. *arXiv*, 2503.15208, 2025. 2
- [23] Lukas Hořlín, Ang Cao, Andrew Owens, Justin Johnson, and Matthias Nießner. Text2Room: Extracting textured 3D meshes from 2D text-to-image models. In *Proc. ICCV*, 2023. 2
- [24] Xiaolu Hou, Mingcheng Li, Dingkan Yang, Jiawei Chen, Ziyun Qian, Xiao Zhao, Yue Jiang, Jinjie Wei, Qingyao Xu, and Lihua Zhang. BloomScene: lightweight structured 3D Gaussian splatting for crossmodal scene generation. *arXiv*, 2501.10462, 2025. 2
- [25] Ziniu Hu, Ahmet Iscen, Aashi Jain, Thomas Kipf, Yisong Yue, David A. Ross, Cordelia Schmid, and Alireza Fathi. SceneCraft: An LLM agent for synthesizing 3d scenes as Blender code. In *Proc. ICML*, 2024. 3
- [26] Zehuan Huang, Yuan-Chen Guo, Xingqiao An, Yunhan Yang, Yangguang Li, Zi-Xin Zou, Ding Liang, Xihui Liu, Yan-Pei Cao, and Lu Sheng. MIDI: Multi-instance diffusion for single image to 3D scene generation. In *Proc. CVPR*, 2025. 3
- [27] Team Hunyuan3D, Shuhui Yang, Mingxin Yang, Yifei Feng, Xin Huang, Sheng Zhang, Zebin He, Di Luo, Haolin Liu, Yunfei Zhao, Qingxiang Lin, Zeqiang Lai, Xianghui Yang, Huiwen Shi, Zibo Zhao, Bowen Zhang, Hongyu Yan, Lifu

- Wang, Sicong Liu, Jihong Zhang, Meng Chen, Liang Dong, Yiwen Jia, Yulin Cai, Jiaao Yu, Yixuan Tang, Dongyuan Guo, Junlin Yu, Hao Zhang, Zheng Ye, Peng He, Runzhou Wu, Shida Wei, Chao Zhang, Yonghao Tan, Yifu Sun, Lin Niu, Shirui Huang, Bojian Zheng, Shu Liu, Shilin Chen, Xiang Yuan, Xiaofeng Yang, Kai Liu, Jianchen Zhu, Peng Chen, Tian Liu, Di Wang, Yuhong Liu, Linus, Jie Jiang, Jingwei Huang, and Chunchao Guo. Hunyuan3D 2.1: From images to high-fidelity 3D assets with production-ready PBR material. *arXiv*, 2506.15442, 2025. 7
- [28] Biliana Kaneva, Josef Sivic, Antonio Torralba, Shai Avidan, and William T. Freeman. Infinite images: Creating and exploring a large photorealistic virtual space. *Proceedings of the IEEE*, 98(8), 2010. 2
- [29] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuhler, and George Drettakis. 3D Gaussian Splatting for real-time radiance field rendering. *Proc. SIGGRAPH*, 42(4), 2023. 2
- [30] Seung Wook Kim, Bradley Brown, Kangxue Yin, Karsten Kreis, Katja Schwarz, Daiqing Li, Robin Rombach, Antonio Torralba, and Sanja Fidler. NeuralField-LDM: Scene generation with hierarchical latent diffusion models. In *Proc. CVPR*, 2023. 3
- [31] Weicheng Kuo, Anelia Angelova, Tsung-Yi Lin, and Angela Dai. Patch2CAD: Patchwise embedding learning for in-the-wild shape retrieval from a single image. In *Proc. ICCV*, 2021. 3
- [32] Han-Hung Lee, Qinghong Han, and Angel X. Chang. NuiScene: exploring efficient generation of unbounded outdoor scenes. In *Proc. ICCV*, 2025. 3
- [33] Jiabao Lei, Jiapeng Tang, and Kui Jia. RGBD2: generative scene synthesis via incremental view inpainting using RGBD diffusion models. In *Proc. CVPR*, 2023. 2
- [34] Bohan Li, Jiazhe Guo, Hongsi Liu, Yingshuang Zou, Yikang Ding, Xiwu Chen, Hu Zhu, Feiyang Tan, Chi Zhang, Tiancai Wang, Shuchang Zhou, Li Zhang, Xiaojuan Qi, Hao Zhao, Mu Yang, Wenjun Zeng, and Xin Jin. UniScene: unified occupancy-centric driving scene generation. In *Proc. CVPR*, 2025.
- [35] Wenrui Li, Fucheng Cai, Yapeng Mi, Zhe Yang, Wangmeng Zuo, Xingtao Wang, and Xiaopeng Fan. SceneDreamer360: text-driven 3D-consistent scene generation with panoramic Gaussian splatting. *arXiv*, 2408.13711, 2024. 2
- [36] Weiyu Li, Jiarui Liu, Rui Chen, Yixun Liang, Xuelin Chen, Ping Tan, and Xiaoxiao Long. CraftsMan: high-fidelity mesh generation with 3d native generation and interactive geometry refiner. *arXiv*, 2405.14979, 2024. 2
- [37] Xinyang Li, Zhangyu Lai, Linning Xu, Yansong Qu, Liujuan Cao, Shengchuan Zhang, Bo Dai, and Rongrong Ji. Director3D: Real-world camera trajectory and 3D scene generation from text. *arXiv*, 2406.17601, 2024. 2
- [38] Yangguang Li, Zi-Xin Zou, Zexiang Liu, Dehu Wang, Yuan Liang, Zhipeng Yu, Xingchao Liu, Yuan-Chen Guo, Ding Liang, Wanli Ouyang, and Yan-Pei Cao. TripoSG: high-fidelity 3D shape synthesis using large-scale rectified flow models. *arXiv*, 2502.06608, 2025. 2
- [39] Zuoyue Li, Tianxing Fan, Zhenqiang Li, Zhaopeng Cui, Yoichi Sato, Marc Pollefeys, and Martin R Oswald. CompNVS: Novel view synthesis with scene completion. In *Proc. ECCV*, 2022. 2
- [40] Hanwen Liang, Junli Cao, Vidit Goel, Guocheng Qian, Sergei Korolev, Demetri Terzopoulos, Konstantinos N. Plataniotis, Sergey Tulyakov, and Jian Ren. Wonderland: Navigating 3D scenes from a single image. In *Proc. CVPR*, 2025. 3
- [41] Chenguo Lin and Yadong MU. InstructScene: instruction-driven 3d indoor scene synthesis with semantic graph prior. In *Proc. ICLR*, 2024. 2
- [42] Lu Ling, Chen-Hsuan Lin, Tsung-Yi Lin, Yifan Ding, Yu Zeng, Yichen Sheng, Yunhao Ge, Ming-Yu Liu, Aniket Bera, and Zhaoshuo Li. Scenethesis: A language and vision agentic framework for 3D scene generation. *arXiv*, 2505.02836, 2025. 2
- [43] A Liu, R Tucker, V Jampani, A Makadia, and N Snavely. . . . Infinite nature: Perpetual view generation of natural scenes from a single image. In *Proc. ICCV*, 2021. 2, 3
- [44] Fangfu Liu, Wenqiang Sun, Hanyang Wang, Yikai Wang, Haowen Sun, Junliang Ye, Jun Zhang, and Yueqi Duan. ReconX: reconstruct any scene from sparse views with video diffusion model. *arXiv*, 2408.16767, 2024. 2
- [45] Minghua Liu, Mikaela Angelina Uy, Donglai Xiang, Hao Su, Sanja Fidler, Nicholas Sharp, and Jun Gao. Partfield: Learning 3d feature fields for part segmentation and beyond. In *Proc. CVPR*, pages 9704–9715, 2025. 6
- [46] Tianqi Liu, Zihao Huang, Zhaoxi Chen, Guangcong Wang, Shoukang Hu, Liao Shen, Huiqiang Sun, Zhiguo Cao, Wei Li, and Ziwei Liu. Free4d: Tuning-free 4d scene generation with spatial-temporal consistency. *arXiv*, 2503.20785, 2025. 2
- [47] W. Lorensen and H. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM Computer Graphics*, 21(24), 1987. 2
- [48] Mahdi Farrokhi Maleki and Richard Zhao. Procedural content generation in games: a survey with insights on emerging llm integration. In *Proc. Conf. on Artificial Intelligence and Interactive Digital Entertainment*, 2024. 3, 1
- [49] Jiageng Mao, Boyi Li, Boris Ivanovic, Yuxiao Chen, Yan Wang, Yurong You, Chaowei Xiao, Danfei Xu, Marco Pavone, and Yue Wang. DreamDrive: generative 4D scene modeling from street view images. *arXiv*, 2501.00601, 2025. 2
- [50] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, Natalia Neverova, Andrea Vedaldi, Oran Gafni, and Filippos Kokkinos. IM-3D: Iterative multiview diffusion and reconstruction for high-quality 3D generation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2024. 2
- [51] Quan Meng, Lei Li, Matthias Nießner, and Angela Dai. LT3SD: latent trees for 3D scene diffusion. *arXiv*, 2409.08215, 2024. 3
- [52] Yanxu Meng, Haoning Wu, Ya Zhang, and Weidi Xie. SceneGen: single-image 3D scene generation in one feedforward pass. *arXiv*, 2508.15769, 2025. 2, 3
- [53] Mikko Mononen and contributors. Recast navigation. <https://github.com/recastnavigation/recastnavigation>, 2016–2026. State-of-the-art navmesh generation and navigation for games. 4

- [54] Chaojun Ni, Xiaofeng Wang, Zheng Zhu, Weijie Wang, Haoyun Li, Guosheng Zhao, Jie Li, Wenkang Qin, Guan Huang, and Wenjun Mei. WonderTurbo: generating interactive 3D world in 0.72 seconds. *arXiv*, 2504.02261, 2025. 2
- [55] Chaojun Ni, Guosheng Zhao, Xiaofeng Wang, Zheng Zhu, Wenkang Qin, Guan Huang, Chen Liu, Yuyin Chen, Yida Wang, Xueyang Zhang, Yifei Zhan, Kun Zhan, Peng Jia, Xianpeng Lang, Xingang Wang, and Wenjun Mei. ReconDreamer: crafting world models for driving scene reconstruction via online restoration. In *Proc. CVPR*, 2025.
- [56] Hao Ouyang, Kathryn Heal, Stephen Lombardi, and Tiancheng Sun. Text2Immersion: Generative immersive scene with 3D gaussians. *arXiv.cs*, abs/2312.09242, 2023. 2
- [57] Geon Yeong Park, Roman Shapovalov, Rakesh Ranjan, Jong Chul Ye, Andrea Vedaldi, and Thu Nguyen-Phuoc. 3d-regen: A unified 3d geometry regeneration framework. *arXiv preprint*, 2026. 2, 6, 7, 3
- [58] Jack Parker-Holder, Philip Ball, Jake Bruce, Vibhavari Dasagi, Kristian Holsheimer, Christos Kaplanis, Alexandre Moufarek, Guy Scully, Jeremy Shar, Jimmy Shi, Stephen Spencer, Jessica Yung, Michael Dennis, Sultan Kenjeyev, Shangbang Long, Vlad Mnihi, Harris Chan, Maxime Gazeau, Bonnie Li, Fabio Pardo, Luyu Wang, Lei Zhang, Frederic Besse, Tim Harley, Anna Mitenkova, Jane Wang, Jeff Clune, Demis Hassabis, Raia Hadsell, Adrian Bolton, Satinder Singh, and Tim Rocktäschel. Genie 2: A large-scale foundation world model, 2024. 3
- [59] Karl Pearson. The problem of the random walk. *Nature*, 72 (1867):342–342, 1905. 1
- [60] Ken Perlin. An image synthesizer. *Proc. SIGGRAPH*, 19(3): 287–296, 1985. 1
- [61] Xiaotian Qiao, Gerhard P. Hancke, and Rynson W.H. Lau. Learning object context for novel-view scene layout generation. In *Proc. CVPR*, 2022. 2
- [62] Alexander Raistrick, Lahav Lipson, Zeyu Ma, Lingjie Mei, Mingzhe Wang, Yiming Zuo, Karhan Kayan, Hongyu Wen, Beining Han, Yihan Wang, Alejandro Newell, Hei Law, Ankit Goyal, Kaiyu Yang, and Jia Deng. Infinite photorealistic worlds using procedural generation. In *Proc. CVPR*, 2023. 3, 1
- [63] Alexander Raistrick, Lingjie Mei, Karhan Kayan, David Yan, Yiming Zuo, Beining Han, Hongyu Wen, Meenal Parakh, Stamatis Alexandropoulos, Lahav Lipson, Zeyu Ma, and Jia Deng. Infinigen indoors: Photorealistic indoor scenes using procedural generation. In *Proc. CVPR*, 2024. 3
- [64] Xuanchi Ren, Tianchang Shen, Jiahui Huang, Huan Ling, Yifan Lu, Merlin Nimier-David, Thomas Müller, Alexander Keller, Sanja Fidler, and Jun Gao. GEN3C: 3D-informed world-consistent video generation with precise camera control. *arXiv*, 2503.03751, 2025. 3
- [65] Chris Rockwell, David F. Fouhey, and Justin Johnson. PixelSynth: Generating a 3D-consistent experience from a single image. In *Proc. ICCV*, 2021. 2
- [66] Katja Schwarz, Norman Mueller, and Peter Kotschieder. Generative Gaussian Splatting: Generating 3D scenes with video diffusion priors. *arXiv*, 2503.13272, 2025. 2
- [67] Zifan Shi, Yujun Shen, Jiapeng Zhu, Dit-Yan Yeung, and Qifeng Chen. 3D-aware indoor scene synthesis with depth priors. In *Proc. ECCV*, 2022.
- [68] Jaidev Shriram, Alex Trevithick, Lingjie Liu, and Ravi Ramamoorthi. RealmDreamer: text-driven 3d scene generation with inpainting and depth diffusion. In *Proc. 3DV*, 2025. 2
- [69] Greg Snook. Simplified 3d movement and pathfinding using navigation meshes. *Game programming gems*, 1(1):288–304, 2000. 3
- [70] Gabriela Ben Melech Stan, Diana Wofk, Scottie Fox, Alex Redden, Will Saxton, Jean Yu, Estelle Aflalo, Shao-Yen Tseng, Fabio Nonato, Matthias Muller, and Vasudev Lal. LDM3D: Latent diffusion model for 3D. *arXiv.cs*, 2305.10853, 2023. 2
- [71] Wenqiang Sun, Shuo Chen, Fangfu Liu, Zilong Chen, Yueqi Duan, Jun Zhang, and Yikai Wang. DimensionX: create any 3D and 4D scenes from a single image with controllable video diffusion. *arXiv*, 2411.04928, 2024. 2
- [72] Stanislaw Szymanowicz, Jason Y. Zhang, Pratul Sriniwasan, Ruiqi Gao, Arthur Brussee, Aleksander Holynski, Ricardo Martin-Brualla, Jonathan T. Barron, and Philipp Henzler. Bolt3D: Generating 3D scenes in seconds. *arXiv*, 2503.14445, 2025. 3
- [73] Shitao Tang, Fuyang Zhang, Jiacheng Chen, Peng Wang, and Yasutaka Furukawa. MVDiffusion: Enabling holistic multi-view image generation with correspondence-aware diffusion. *arXiv.cs*, abs/2307.01097, 2023. 2
- [74] Chaoyang Wang, Ashkan Mirzaei, Vidit Goel, Willi Menapace, Aliaksandr Siarohin, Avalon Vinella, Michael Vasilkovsky, Ivan Skorokhodov, Vladislav Shakhrai, Sergey Korolev, Sergey Tulyakov, and Peter Wonka. 4real-video-v2: Fused view-time attention and feedforward reconstruction for 4d scene generation. *arXiv*, 2506.18839, 2025. 2
- [75] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. In *Proc. CVPR*, 2020. 2
- [76] World Labs. Marble: Create 3d worlds from a simple prompt. <https://marble.worldlabs.ai>, 2025. Accessed: 2026-03-26. 2, 8
- [77] Zhennan Wu, Yang Li, Han Yan, Taizhang Shang, Weixuan Sun, Senbo Wang, Ruikai Cui, Weizhe Liu, Hiroyuki Sato, Hongdong Li, and Pan Ji. BlockFusion: Expandable 3D scene generation using latent tri-plane extrapolation. *arXiv.cs*, 2024. 3
- [78] Jianfeng Xiang, Jiaolong Yang, Binbin Huang, and Xin Tong. 3D-aware image generation using 2D diffusion models. *arXiv.cs*, abs/2303.17905, 2023. 2
- [79] Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. In *Proc. CVPR*, pages 21469–21480, 2025. 7
- [80] Shuai Yang, Jing Tan, Mengchen Zhang, Tong Wu, Yixuan Li, Gordon Wetzstein, Ziwei Liu, and Dahua Lin. LayerPano3D: layered 3D panorama for hyper-immersive scene generation. *arXiv*, 2408.13252, 2025. 2
- [81] Yuanbo Yang, Jiahao Shao, Xinyang Li, Yujun Shen, Andreas Geiger, and Yiyi Liao. Prometheus: 3D-aware latent

- diffusion models for feed-forward text-to-3d scene generation. *arXiv*, 2412.21117, 2025. 3
- [82] Hong-Xing Yu, Haoyi Duan, Junhwa Hur, Kyle Sargent, Michael Rubinstein, William T. Freeman, Forrester Cole, Deqing Sun, Noah Snaveley, Jiajun Wu, and Charles Herrmann. Wonderjourney: Going from anywhere to everywhere. *arXiv.cs*, abs/2312.03884, 2023. 2
- [83] Hong-Xing Yu, Haoyi Duan, Charles Herrmann, William T. Freeman, and Jiajun Wu. WonderWorld: interactive 3D scene generation from a single image. *arXiv*, 2406.09394, 2025. 2
- [84] Guangyao Zhai, Evin Pinar Örnek, Shun-Cheng Wu, Yan Di, Federico Tombari, Nassir Navab, and Benjamin Busam. CommonScenes: generating commonsense 3D indoor scenes with scene graphs. In *Proc. NeurIPS*, 2023. 2
- [85] Shangjin Zhai, Zhichao Ye, Jialin Liu, Weijian Xie, Jiaqi Hu, Zhen Peng, Hua Xue, Danpeng Chen, Xiaomeng Wang, Lei Yang, Nan Wang, Haomin Liu, and Guofeng Zhang. StarGen: a spatiotemporal autoregression framework with video diffusion model for scalable and controllable scene generation. *arXiv*, 2501.05763, 2025. 2
- [86] Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3DShape2VecSet: A 3D shape representation for neural fields and generative diffusion models. In *ACM Transactions on Graphics*, 2023. 2, 3, 4
- [87] Jingbo Zhang, Xiaoyu Li, Ziyu Wan, Can Wang, and Jing Liao. Text2NeRF: Text-driven 3D scene generation with neural radiance fields. *arXiv.cs*, abs/2305.11588, 2023. 2
- [88] Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. CLAY: A controllable large-scale generative model for creating high-quality 3D assets. In *Proc. SIGGRAPH*, 2024. 2
- [89] Songchun Zhang, Yibo Zhang, Quan Zheng, Rui Ma, Wei Hua, Hujun Bao, Weiwei Xu, and Changqing Zou. 3D-SceneDreamer: text-driven 3D-consistent scene generation. In *Proc. CVPR*, 2024. 2
- [90] Guosheng Zhao, Chaojun Ni, Xiaofeng Wang, Zheng Zhu, Xueyang Zhang, Yida Wang, Guan Huang, Xinze Chen, Boyuan Wang, Youyi Zhang, Wenjun Mei, and Xingang Wang. DriveDreamer4D: world models are effective data machines for 4d driving scene representation. In *Proc. CVPR*, 2025.
- [91] Guosheng Zhao, Xiaofeng Wang, Chaojun Ni, Zheng Zhu, Wenkang Qin, Guan Huang, and Xingang Wang. ReconDreamer++: harmonizing generative and reconstructive models for driving scene representation. *arXiv*, 2503.18438, 2025. 2
- [92] Zibo Zhao, Zeqiang Lai, Qingxiang Lin, Yunfei Zhao, Haolin Liu, Shuhui Yang, Yifei Feng, Mingxin Yang, Sheng Zhang, Xianghui Yang, Huiwen Shi, Sicong Liu, Junta Wu, Yihang Lian, Fan Yang, Ruining Tang, Zebin He, Xinzhou Wang, Jian Liu, Xuhui Zuo, Zhuo Chen, Biwen Lei, Hao-han Weng, Jing Xu, Yiling Zhu, Xinhai Liu, Lixin Xu, Changrong Hu, Tianyu Huang, Lifu Wang, Jihong Zhang, Meng Chen, Liang Dong, Yiwen Jia, Yulin Cai, Jiaao Yu, Yixuan Tang, Hao Zhang, Zheng Ye, Peng He, Runzhou Wu, Chao Zhang, Yonghao Tan, Jie Xiao, Yangyu Tao, Jianchen Zhu, Jinbao Xue, Kai Liu, Chongqing Zhao, Xinming Wu, Zhichao Hu, Lei Qin, Jianbing Peng, Zhan Li, Minghui Chen, Xipeng Zhang, Lin Niu, Paige Wang, Yingkai Wang, Haozhao Kuang, Zhongyi Fan, Xu Zheng, Weihao Zhuang, YingPing He, Tian Liu, Yong Yang, Di Wang, Yuhong Liu, Jie Jiang, Jingwei Huang, and Chunchao Guo. Hunyuan3D 2.0: Scaling diffusion models for high resolution textured 3d assets generation. *arXiv*, 2501.12202, 2025. 2
- [93] Mengqi Zhou, Yuxi Wang, Jun Hou, Shougao Zhang, Yiwei Li, Chuanchen Luo, Junran Peng, and Zhaoxiang Zhang. SceneX: procedural controllable large-scale scene generation. *arXiv*, 2403.15698, 2024. 3
- [94] Shijie Zhou, Zhiwen Fan, Dejia Xu, Haoran Chang, Pradyumna Chari, Tejas K Bhargadwaj, Suyu You, Zhangyang Wang, and Achuta Kadambi. DreamScene360: unconstrained Text-to-3D scene generation with panoramic Gaussian splatting. In *Proc. ECCV*, 2024. 2
- [95] Xiaoyu Zhou, Xingjian Ran, Yajiao Xiong, Jinlin He, Zhiwei Lin, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. GALA3D: Towards text-to-3D complex scene generation via layout-guided generative gaussian splatting. *arXiv.cs*, abs/2402.07207, 2024. 2
- [96] Xin Zhou, Dingkan Liang, Sifan Tu, Xiwu Chen, Yikang Ding, Dingyuan Zhang, Feiyang Tan, Hengshuang Zhao, and Xiang Bai. HERMES: a unified self-driving world model for simultaneous 3D scene understanding and generation. *arXiv*, 2501.14729, 2025. 2