Gen-VR: End-to-end Pipeline for Generating 3D Virtual Reality Scenes from User Text Prompts

Richard Fang
University of California Santa Barbara, United States
University of Osaka XR Group, Graduate School of Engineering Science
FrontierLab Program Supervisor: Professor Daisuke Iwai
Email: richardfang@ucsb.edu

***Abstract*—The rising interest in virtual reality (VR) and the broader vision of the metaverse has sparked a growing demand for immersive, interactive digital environments. However, creating high-quality VR content remains difficult, requiring significant technical expertise, time, and resources—barriers that limit participation to skilled developers and artists. Recent advancements in generative AI technologies have introduced new opportunities to democratize this process, enabling intuitive AI tools that automate aspects of content creation. This research presents *Gen-VR*, an early-stage prototype of an end-to-end pipeline that converts natural language prompts into fully instantiated 3D VR scenes. Our system uses multi-stage large language model prompting, 3D asset search and retrieval, automated loading and instantiation, and novel positioning and scaling algorithms to achieve generation of realistic, context-aware virtual environments that maintain spatial coherence and require no manual design intervention. This approach lowers the barrier to VR scene creation, enabling users with no prior expertise to generate immersive environments using only natural language.**

I. Introduction

The growing popularity of virtual reality (VR) technologies and the broader concept of the metaverse have increased demand for immersive, interactive virtual environments in domains such as gaming, education, and simulation [28], [29]. However, creating high-quality VR scenes is labor-intensive, requiring expertise in 3D modeling, programming, and content integration using tools such as Blender, Unity, and 3ds Max [22], [24], [23]. This high barrier to entry limits participation to professional developers and artists, excluding casual users and educators seeking rapid VR content creation.

Generative AI offers a promising avenue to address these barriers. Large language models (LLMs) such as GPT-4 [18] demonstrate strong capabilities in natural language understanding, structured output generation, and reasoning. Parallel advances in generative visual and 3D content synthesis, including Stable Diffusion [4] and text-to-3D pipelines like DreamFusion [6], have shown the ability to create realistic assets from textual descriptions.

Despite these developments, current tools primarily focus on isolated outputs (e.g., single objects or static images) and lack the integration required to generate coherent, interactive VR environments. Existing VR scene creation methods either rely on manual workflows or template-based systems with limited flexibility [25], [26].

To address these limitations, we introduce *Gen-VR*, an end-to-end pipeline that converts natural language prompts into fully realized VR scenes in Unity. Our system uses multi-stage LLM prompting, automated asset retrieval from APIs, generative fallback models, and novel positioning and scaling algorithms to achieve seamless, high-quality scene generation. This approach democratizes VR content creation, enabling non-technical users to construct immersive environments through intuitive language input.

II. Related Work

A. Generative Models for Digital Content

Recent advances in **diffusion models** [1], [2], [3], [4], [5] have revolutionized image generation by iteratively refining noisy inputs to match learned data distributions. Stable Diffusion [4] and similar systems combine diffusion backbones with transformer-based language encoders, achieving state-of-the-art text-to-image synthesis. Building on these methods, text-to-3D approaches such as DreamFusion [6], Magic3D [7], Fantasia3D [8], extend generative modeling into 3D, enabling neural mesh and radiance field creation from text prompts.

In parallel, **generative adversarial networks (GANs)** [9] have played a pivotal role in image synthesis and asset generation. StyleGAN variants [10], [11] and extensions like StyleLight [12] demonstrate high-quality outputs with fine-grained control, informing methods for texture and scene component generation.

B. Language Models and Scene Automation

LLMs enable semantic parsing of unstructured input and generation of structured data for downstream tasks. Models such as GPT-3 [15], Gopher [16], and LaMDA [17] exhibit emergent reasoning capabilities [18], [19], which have been applied to tasks like automated code synthesis and structured content generation. Prompt engineering techniques [20], [21] enhance reliability by guiding LLM outputs toward consistent, schema-compliant formats, making them effective for orchestrating multi-stage pipelines.

C. VR Scene Generation and Procedural Content

Traditional VR scene development requires extensive manual modeling, asset sourcing, and placement [22], [24], [23].

Early language-driven scene generation systems, such as SceneSeer [25] and database-driven synthesis [26], explored translating text descriptions into object layouts but were limited to static indoor scenes and a small set of items. Recent VR generation approaches [28], [30], [27] have improved the automation in certain aspects of environment construction but still have a limited asset pool and strict user prompt constraints, therefore lacking the necessary diversity or customization users are looking for. Our work builds upon these by integrating LLM-driven semantic parsing accommodate a significantly wider range of prompts and a hybrid asset retrieval system that enables diverse search across millions of assets as well as custom content generation with usage of the newest text-to-3D generation models.

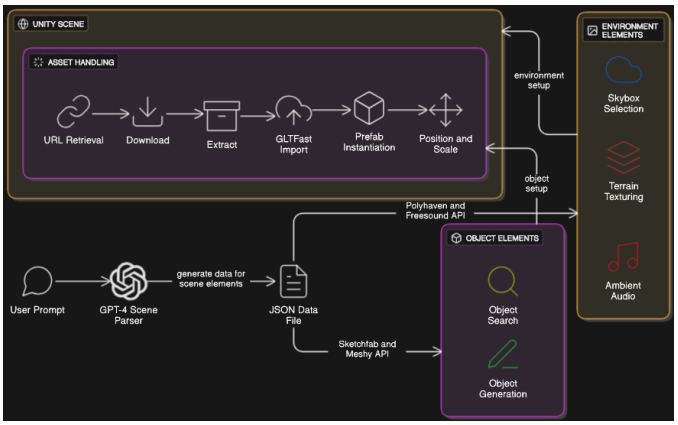


Fig. 1: Pipeline overview: user prompts are parsed by an LLM, assets retrieved or generated, refined, and instantiated in Unity.

III. System Design

The architecture of our system (Fig. 1) consists of several integrated stages that connect LLM-driven semantic parsing with automated asset retrieval and Unity instantiation.

A. Prompt Parsing

User text prompts are processed by large language models to extract structured scene descriptions in JSON format. This structured output includes object elements (3D assets) and environmental context (skybox, terrain textures, and audio). Prompt engineering techniques [20] ensure consistency in output schema and improve the accuracy of object extraction.

B. Asset Retrieval

Each scene element triggers automated asset retrieval:

- **Sketchfab API:** Used for 3D object retrieval, supporting OAuth authentication and direct asset downloads for millions of 3D assets.
- **PolyHaven API:** Provides HDR skyboxes and PBR terrain textures suitable for Unity.
- **Freesound API:** Retrieves ambient soundscapes and effects aligned with the context of the scene from database of more than 500,000 audio files.

These resources leverage large public repositories of freely licensed assets, reducing reliance on manual 3D modeling workflows and optimizing the system performance.

C. Generative Fallbacks

For elements not found in existing libraries, Meshy AI is used to generate 3D assets from text prompts. This ensures coverage for rare or highly specific objects, complementing search-based methods and offering extreme flexibility.

D. Refinement and Selection

Large language models are used to re-rank candidate assets based on semantic relevance and scene coherence, informed by prompt context and learned associations [19], [18]. The best candidate for each object is then selected from the search results and generated assets.

E. Unity Integration

Assets are downloaded, processed, and imported into Unity using GLTFast, which efficiently handles glTF and GLB formats. Unity scripts automate instantiation, component assignment, and VR compatibility. Procedural techniques similar to those in VRoamer [28] are adapted for initial runtime spatial placement.

F. Position and Scale Adjustment

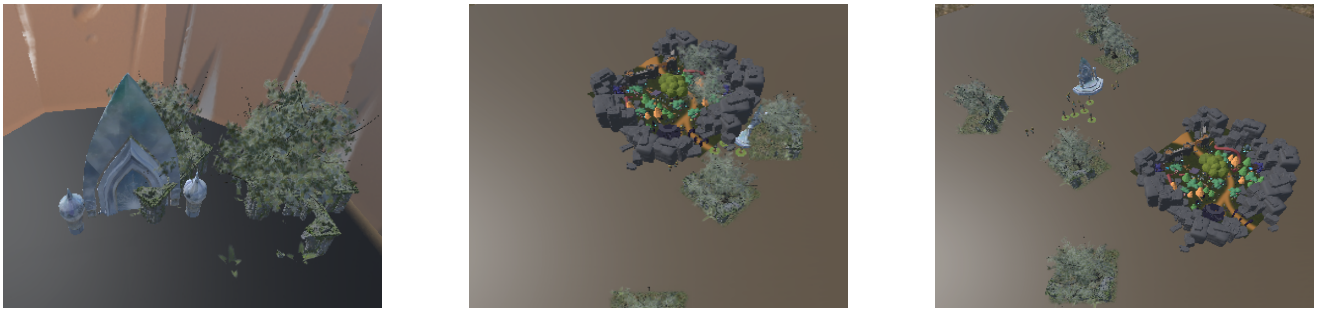
After initial placement, we adjust the position and scale of each object using a custom algorithm that incorporates:

- **Dynamic Normalization:** We dynamically calculate bounding boxes for each object in the scene and normalize all objects to unit size. Then, each object is scaled relative to the anchor terrain following dimension priors from real-world data and large language model prediction.
- **Bayesian Modeling:** We calculate Bayesian probabilistic priors for each object based on prompt context and large language model interpretation. Then we procedurally update positioning of each object, inspired by earlier spatial reasoning approaches in SceneSeer [25]. We iteratively update object position probabilities with each placed object and test multiple runs to prevent outlier placements.
- **LLM Integration:** OpenAI models assist in relational placement (e.g., positioning chairs near tables) using semantic reasoning and also run sanity checks on the adjusted scales and positionings throughout the process.

IV. Results and Evaluation

We evaluated *Gen-VR* on diverse prompts (Fig. 3). The system produced scenes featuring:

- **Prompt representation**, accurately reflecting objects, spatial arrangements, and environmental context described by users.
- **Coherent visual theming**, combining HDR skyboxes and environment-aligned terrain.



(a) Naive positioning (b) Rule-based positioning (c) Proposed positioning

Fig. 2: Comparison of object positioning techniques in VR scenes: (a) naive placement, (b) rule-based, and (c) our probabilistic positioning with LLM scaling.

- **Sensory immersion**, incorporating context-aware ambient soundscapes, atmospheric effects, and lighting variations.
- **Automatic spatial realism**, improving on static layouts from prior works [25], [28].
- **Efficient scene assembly**, reducing setup time by over 80% compared to manual Unity workflows [24].

V. Discussion and Limitations

While our study demonstrates the feasibility and potential of an LLM-driven pipeline for automated VR scene generation, several limitations should be acknowledged.

One major limitation is the system's difficulty in handling complex, multi-room layouts and spatially intricate environments. While our probabilistic placement and scaling algorithms perform well in free space contexts, environments with multiple enclosed spaces and complex positional interactions between different objects often require more nuanced hierarchical reasoning and finer spatial partitioning. This is an area where learning model based spatial reasoning could significantly improve object placement and should be explored.

A further limitation lies in the restricted interactivity and animation within generated scenes. While our system can produce visually coherent static VR environments, it currently lacks dynamic object behaviors or physics-based interactions, both of which are critical for deeper immersion. Similar issues have been observed in prior VR generation systems [28], [27], where static content reduced user engagement. This limitation stems partially from the reliance on Unity's basic instantiation workflows without integration of behavior scripting or physics simulations, which we plan to address by incorporating physics engines and procedural animation libraries.

We also acknowledge methodological limitations in how we evaluated the system. Our evaluation relied on literature review and direct comparison to previous models. While this combination provided a useful overview of the system's potential, it still contains a high level of subjectivity and may not have captured all limitations of LLM-driven automation in VR contexts. Expanding future evaluations to include larger-scale user studies and empirical usability testing will provide more comprehensive insights into the strengths and weaknesses of the proposed framework.

Moreover, our qualitative analysis highlighted imbalances in visual quality between different scene components, particularly between high-resolution 2D skyboxes and lower-quality 3D assets generated from text prompts. This disparity can reduce immersion by making objects appear mismatched with their environments. Addressing this will require continued advancements in text-to-3D model generation as well as improved LLM-guided asset filtering to ensure consistent stylistic alignment across all scene elements.

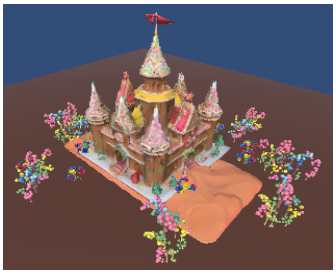
Despite these limitations, our work represents an important step toward a unified, LLM-based framework for automated VR scene generation. The system demonstrates how natural language can be leveraged to control complex scene assembly tasks, reducing the technical burden on end users and democratizing VR content creation. Future research will build upon this framework by incorporating interactive elements, expanding asset retrieval to include animated objects and behaviors, and leveraging ongoing improvements in LLM reasoning and multimodal generation. Ultimately, we envision a finely-tuned learning model to solve many of the existing issues with the VR scene generation framework.

VI. Conclusion

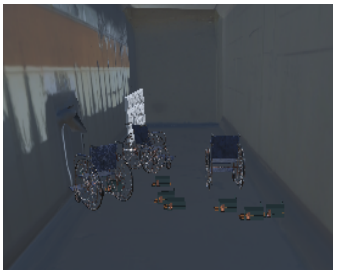
Gen-VR demonstrates the capability of generative AI models for producing VR environments and validates the potential for text-driven pipelines to democratize VR scene creation. By combining language models, asset APIs, and automated Unity integration, our framework allows greater generation diversity and user customization than the predecessors. In addition, we iterate upon previous position and scaling methods presenting novel methods to improve spatial realism in generated 3d scenes. Our research hopes to continue leveraging ongoing advancements in generative AI technology to help pave the path for providing a low-barrier tool for generating immersive VR environments applicable to gaming, education, simulation, and storytelling.

References

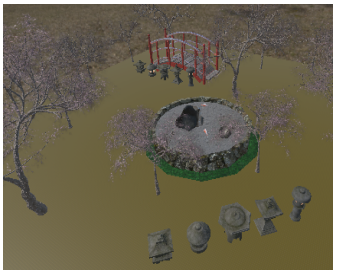
[1] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," NeurIPS, 2020.



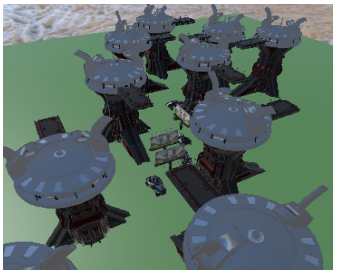
(a) Candy land with chocolatey ground, gumdrop trees, and lollipop flowers. Giant gingerbread castle.



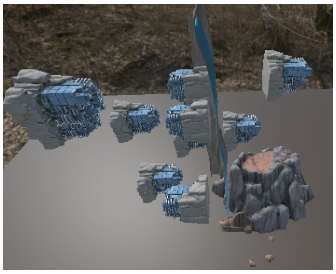
(b) Abandoned asylum hallway with flickering lights, overturned wheelchairs, and peeling walls.



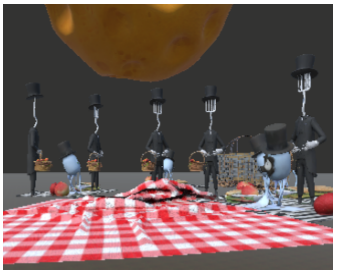
(c) Traditional Japanese Garden



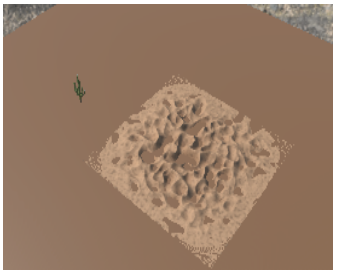
(d) Space Colony



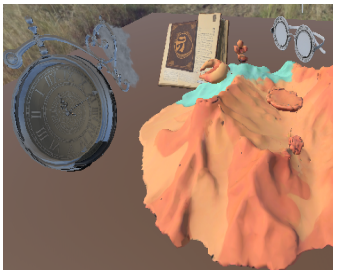
(e) Gravity-defying city where buildings grow sideways from cliffs and rivers flow upward.



(f) Cratered moon of cheese hovers above sentient forks and spoons having a picnic in formal attire.



(g) Empty desert and 1 cactus



(h) Perception

Fig. 3: Comparison of eight generated VR scenes using our system. All images are shown at the same size for visual consistency.

[2] J. Sohl-Dickstein et al., "Deep unsupervised learning using nonequilibrium thermodynamics," ICML, 2015.

[3] Y. Song et al., "Score-based generative modeling through stochastic differential equations," arXiv:2011.13456, 2020.

[4] R. Rombach et al., "High-resolution image synthesis with latent diffusion models," CVPR, 2022.

[5] C. Saharia et al., "Photorealistic text-to-image diffusion models with deep language understanding," NeurIPS, 2022.

[6] B. Poole et al., "DreamFusion: Text-to-3D using 2D diffusion," arXiv:2209.14988, 2022.

[7] C.-H. Lin et al., "Magic3D: High-resolution text-to-3D content creation," CVPR, 2023.

[8] R. Chen et al., "Fantasia3D: Disentangling geometry and appearance for high-quality text-to-3D," arXiv:2303.13873, 2023.

[9] I. Goodfellow et al., "Generative adversarial networks," Commun. ACM, vol. 63, no. 11, 2020.

[10] T. Karras et al., "A style-based generator architecture for GANs," CVPR, 2019.

[11] T. Karras et al., "Analyzing and improving the image quality of StyleGAN," CVPR, 2020.

[12] G. Wang et al., "StyleLight: HDR panorama generation for lighting estimation," ECCV, 2022.

[13] E. Cambria and B. White, "Jumping NLP curves: A review of natural language processing research," IEEE CIM, 2014.

[14] R. Rosenfeld, "Two decades of statistical language modeling," Proc. IEEE, 2000.

[15] T. Brown et al., "Language models are few-shot learners," NeurIPS, 2020.

[16] J. Rae et al., "Scaling language models: Methods and insights from training Gopher," arXiv:2112.11446, 2021.

[17] R. Thoppilan et al., "LaMDA: Language models for dialog applications," arXiv:2201.08246, 2022.

[18] J. Wei et al., "Emergent abilities of large language models," arXiv:2206.07682, 2022.

[19] H. Liu et al., "Evaluating the logical reasoning ability of ChatGPT and GPT-4," arXiv:2304.03439, 2023.

[20] J. White et al., "A prompt pattern catalog to enhance prompt engineering with ChatGPT," arXiv:2302.11382, 2023.

[21] Y. Bang et al., "A multitask, multilingual, multimodal evaluation of ChatGPT," arXiv:2302.04023, 2023.

[22] Blender Foundation, "Blender Software," https://www.blender.org, 2023.

[23] Autodesk, "3ds Max Software," https://www.autodesk.com/products/3ds-max/, 2023.

[24] Unity Technologies, "Unity Software," https://unity.com, 2023.

[25] A. Chang et al., "SceneSeer: 3D scene design with natural language," arXiv:1703.00050, 2017.

[26] R. Ma et al., "Language-driven synthesis of 3D scenes from scene databases," ACM TOG, 2018.

[27] H. Yi et al., "MIME: Human-aware 3D scene generation," CVPR, 2023.

[28] L.-P. Cheng et al., "VRoamer: Generating on-the-fly VR experiences," IEEE VR, 2019.

[29] J.-H. Cheng et al., "Impossible staircase: Vertically real walking in an infinite virtual tower," IEEE VR, 2021.

[30] M. Sra et al., "Procedurally generated virtual reality from 3D reconstructed physical space," VRST, 2016.