[TINY PAPER] MODULAR TRAINING-FREE CONSTRUCTION OF EXECUTABLE 3D WORLDS FROM NARRATIVE TEXT *

Sanchit Singh
Department of Computer Science
San Diego State University
ssingh1949@sdsu.edu

ABSTRACT

Recent work on world generation has largely emphasized foundation-scale generative models trained on large multimodal datasets, often requiring substantial computational resources. In this work, we explore an alternative perspective: treating narrative world construction as a structured compilation problem rather than an end-to-end learned generation problem. We present a modular, training-free framework that uses multimodal large language models only for semantic abstraction, while delegating topology construction, spatial layout, traversability, and environment assembly to deterministic algorithms. The resulting pipeline converts narrative text into story-driven, navigable 3D worlds through lightweight API calls and executable compilation in the Godot engine. Across 20 prompts, the system produces collision-free, overlap-free environments with 100% door reachability on commodity hardware. Our results suggest that world model research can advance not only through larger learned generators, but also through decomposition, controllability, and systems-level structure.

1 INTRODUCTION

Recent advances in world model research have followed a clear generative trajectory, progressing from high-fidelity static image synthesis to temporally coherent video generation, and more recently toward real-time and interactive environments. Latent diffusion models first demonstrated that complex visual distributions could be efficiently modeled in compressed latent spaces (Rombach et al., 2022), establishing diffusion as a scalable paradigm for image-based world generation. This foundation was subsequently extended to the temporal domain, enabling video diffusion models capable of modeling spatiotemporal dynamics and long-horizon visual consistency (Blattmann et al., 2023). Building on these developments, recent systems have begun to incorporate agent conditioning and interaction, positioning diffusion models as general-purpose simulators that respond dynamically to user actions (Bruce et al., 2024). Together, these works reflect a broader shift toward world models that aim to support continuous, interactive, and embodied experience.

Despite their impressive capabilities, this progression has been accompanied by rapidly increasing computational demands. Recent studies have shown that diffusion-based architectures exhibit clear power-law scaling behavior with respect to compute budget, with improvements in generation quality requiring joint increases in model size, data, and total training FLOPs (Liang et al., 2024). In practice, diffusion-based world models for video and interaction incur substantial training and inference costs, particularly as resolution, temporal horizon, and agent conditioning increase (Blattmann et al., 2023; Bruce et al., 2024). As a result, many such systems remain accessible primarily to well-resourced research labs, limiting their practicality for individual researchers, educators, and practitioners.

In this work, we propose an alternative approach to world model construction that prioritizes explicit structure, determinism, and accessibility under strict compute constraints. Rather than learning world dynamics end-to-end, our method constructs story-driven, navigable 3D worlds through a

*Code: https://github.com/sanchitsingh001/NarrativeWorlds

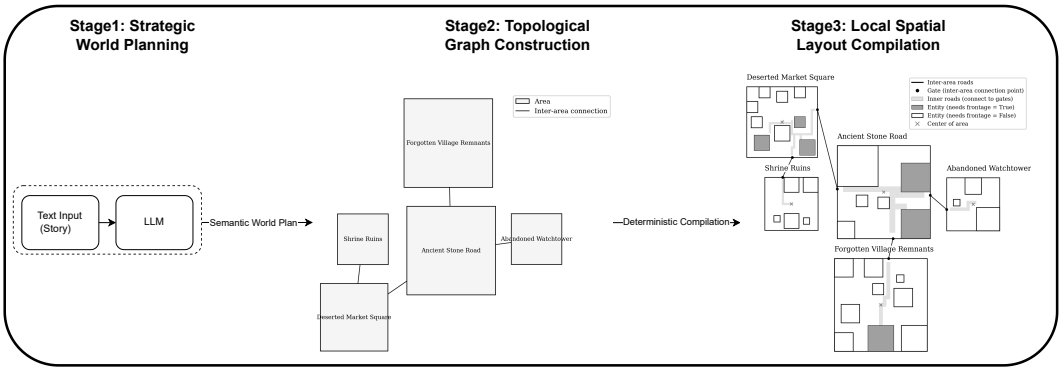


Figure 1: Overview of the training-free world construction pipeline. A natural-language narrative is parsed by a large language model into a semantic world plan that specifies regions and their associated entities (Stage 1). This plan is used to construct an abstract topological world graph defining relative placement and traversable connections between regions (Stage 2). The graph is then deterministically compiled into collision-free, tile-based spatial layouts, with entities grounded within each area and explicit traversal gates and routed road networks (Stage 3).

modular, training-free pipeline that combines pretrained multimodal language models accessed via lightweight API calls with algorithmic world assembly. This perspective complements diffusion-based world models by opening a distinct design space centered on structured abstraction and compositional reasoning, rather than large-scale training and specialized hardware.

2 METHODOLOGY

Our approach constructs a 3D world through five sequential stages, transforming a natural-language narrative into an executable environment instantiated in (Godot Engine Project, 2024). A large language model (GPT-4o (OpenAI, 2024)) is used to extract areas, entities, and relational constraints from text, while all spatial layout, connectivity, and world assembly are handled algorithmically and deterministically.

2.1 STAGE 1: STRATEGIC WORLD PLANNING

The process begins with a natural-language story as input, which is transformed by a large language model into a structured world plan consisting of semantic areas and required entities. Each area is annotated with a localized narrative description and symbolic scale information, providing high-level constraints on the intended spatial extent and content of the region.

Crucially, the language model is not used to predict geometric coordinates. Instead, spatial grounding is deferred to later stages, ensuring that semantic planning remains flexible while all geometric structure is derived deterministically. This separation avoids unreliable geometric inference from language models and enables reproducible world construction.

2.2 STAGE 2: TOPOLOGICAL GRAPH CONSTRUCTION

Given the semantic areas identified in Stage 2.1, we construct a topological world graph that defines relative placement and traversable connectivity between regions. Areas are arranged using symbolic spatial relationships, while explicit inter-area connections specify which regions are directly traversable.

This representation captures global structure without committing to exact geometry. All symbolic placements and connections are grounded into concrete spatial layouts during subsequent deterministic compilation, ensuring both coherent world organization and reproducible connectivity.

Figure 2: Context-conditioned asset grounding and world assembly. Given the resolved tile-based world layout, context-conditioned 3D assets are generated for each entity by conditioning text-to-3D prompts on the entity's semantic role, local area narrative, and global style constraints (Stage 4). Because text-to-3D models produce meshes with arbitrary orientation, canonical front, back, left, and right views are rendered and a vision-language model is used to identify the semantic front-facing direction when required. The resolved tile-based layout and oriented assets are then deterministically assembled into an executable environment instantiated in the Godot engine, producing a coherent and traversable 3D world (Stage 5).

2.3 Stage 3: Local Spatial Layout Compilation

For each area in the world graph (Stage 2.2), we deterministically ground entities, traversal interfaces, and road networks onto a discrete two-dimensional tile grid. Each area is assigned a fixed grid resolution based on its symbolic scale and compiled independently to preserve modularity.

Entities and gates are placed under collision and boundary constraints, and a connected road network is generated to ensure full traversability between regions and landmarks. All geometric decisions are resolved algorithmically, yielding a collision-free and reproducible spatial layout that fully specifies world geometry.

2.4 Stage 4: Context-Conditioned Asset Specification

Given the resolved tile-based world layout from Stage 2.3, we generate context-aware visual specifications for each entity. For every entity, an LLM constructs a text prompt conditioned on the local area narrative (Stage 2.1), the entity's semantic role, and global stylistic constraints. These prompts are passed to a text-to-3D asset generator (Hunyuan3D 2.5 Team (2025)), which produces mesh assets in `.glb` format independently of their final placement, enabling asset reuse across worlds. The asset generation model is treated as a black-box renderer and does not influence world structure, layout, or traversal logic.

Orientation and Frontage Alignment. Because text-to-3D models produce meshes with arbitrary orientation, we resolve asset orientation explicitly during world assembly. Inspired by image-grid based visual reasoning by Kim et al. (2024), we render canonical front, back, left, and right views of each generated asset and arrange them into an image grid. A vision-language model[1] (Dubey et al., 2024) identifies the semantic front-facing view, which is then used to orient the asset. For entities marked with `needs_frontage` (Stage 2.3), the asset's forward direction is aligned with the normal of the nearest road edge or frontage spur tile; other entities are assigned a default or randomly sampled orientation consistent with area-level variation.

[1] We use `meta-llama/Llama-3.2-11B-Vision-Instruct` for vision-language inference

2.5 Stage 5: World Assembly and Environment Instantiation

In the final stage, the resolved tile-based world layout and generated 3D assets are assembled into an executable environment. The global tilemap is instantiated using fixed textures for terrain and road surfaces, and each asset is placed at its assigned tile coordinates with the resolved orientation.

We implement this assembly step in the Godot game engine, chosen for its strong scriptability and headless execution support, which allows fully automated world construction without interactive rendering. For reproducibility, we provide the exact LLM/VLM prompt templates and the deterministic compilation details in Appendix A.1 and Appendix A.2, respectively.

3 Analysis & Discussion

Because the primary contribution of this work is a modular systems pipeline for executable world construction, our evaluation focuses on invariant properties that directly reflect whether the compiled worlds are valid and traversable. We therefore measure collision avoidance, overlap avoidance, reachability, and generation efficiency. We do not claim that these metrics capture narrative fidelity, perceptual realism, or downstream embodied-AI utility, which remain important directions for future evaluation.

Table 1 reports structural validity, functional traversability, and systems-level metrics computed from the compiled world representation. Representative qualitative examples corresponding to these statistics are shown in Appendix A.3. Structural validity is assessed via area overlap and entity collision indicators, while functional traversability is evaluated using road-network diagnostics and, critically, door reachability, which measures whether all inter-area traversal interfaces are reachable from a designated anchor. Across 20 narrative prompts, the pipeline produces collision-free, overlap-free worlds with 100% door reachability, indicating that the resulting environments are consistently executable and navigable; terminal connectivity is reported as a diagnostic and does not affect inter-area traversability when door reachability is satisfied.

Metric	**Value**
Collision-free worlds (%)	100.0
Area overlap-free worlds (%)	100.0
Mean terminal connectivity (diagnostic)	0.786
Door reachability rate (%)	**100.0**
Mean generation time (s)	51.2 ± 9.2
GPU required	No (API-based inference)

Table 1: Structural and functional metrics over 20 prompts. Door reachability (fraction of gates reachable from an anchor via roads) is the primary success criterion.

We leave ablations against monolithic and alternative multi-stage pipelines to future work.

Modeling assumptions and design scope. The current implementation adopts several simplifying assumptions to enable deterministic compilation, including flat terrain within each area and the absence of large natural features such as rivers, lakes, coastlines, or elevation changes (small man-made water features are treated as standard entities). World structure is intentionally decomposed into modular components—including area layout, connectivity, entity placement, and asset grounding—rather than modeled within a single monolithic representation. These assumptions allow spatial layout and traversability to be resolved reproducibly within a discrete grid representation, while preserving extensibility to richer terrain variation or additional decorative assets through future modules.

Limitations and future directions. While the framework enables deterministic construction of coherent and traversable worlds, it does not learn environment dynamics or visual priors from data. World structure is specified through explicit rules rather than optimized to match real-world distributions or perceptual realism. An important direction for future work is to integrate this deterministic world compilation backbone with learned generative components, following hybrid approaches such as Wang et al., 2025, which combine algorithmic structure with diffusion-based image generation.

REFERENCES

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023. URL https://arxiv.org/abs/2311.15127.

Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. *arXiv preprint arXiv:2402.15391*, 2024. URL https://arxiv.org/abs/2402.15391.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Godot Engine Project. Godot engine, 2024. URL https://godotengine.org. Version 4.3.

Wonkyun Kim, Changin Choi, Wonseok Lee, and Wonjong Rhee. An image grid can be worth a video: Zero-shot video question answering using a vlm. *IEEE Access*, 12:193057–193075, 2024. doi: 10.1109/ACCESS.2024.3517625.

Zhengyang Liang, Hao He, Ceyuan Yang, and Bo Dai. Scaling laws for diffusion transformers, 2024. URL https://arxiv.org/abs/2410.08184.

OpenAI. Gpt-4o system card, 2024. URL https://arxiv.org/abs/2410.21276.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn 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. URL https://arxiv.org/abs/2112.10752.

Tencent Hunyuan3D Team. Hunyuan3d 2.5: Towards high-fidelity 3d assets generation with ultimate details, 2025. URL https://arxiv.org/abs/2506.16504.

Dilin Wang et al. Worldgen: From text to traversable and interactive 3d worlds. *arXiv preprint arXiv:2511.16825*, 2025. URL https://arxiv.org/abs/2511.16825.

A APPENDIX

A.1 PROMPT TEMPLATES (LLM AND VLM)

Stage 1: Area decomposition and local narratives (LLM). We prompt an LLM to decompose the input narrative into 3–7 outdoor areas and to produce architecture-focused descriptions intended for downstream 3D asset generation. We enforce: (i) outdoor-only areas, (ii) no time-sliced duplicates (e.g., day/night variants), and (iii) simplified terrain assumptions.

```
You are a creative world planner for a procedural map
generator.
Step 1:  Plan the areas.  Based on the story, identify 3-7
distinct areas.  - Treat areas as abstract locations, not
specific moments in time.  - Do NOT create separate areas
that are only distinguished by time-of-day, weather, or
other transient state.  - Describe such changes as possible
states of the SAME area.
CRITICAL: ALL AREAS ARE OUTDOOR SPACES ONLY - Every area
is an EXTERIOR location on the game map.  - Do NOT create
indoor areas (cathedral_interior, shop_interior, throne_room,
etc.).  - Buildings should be described from the OUTSIDE
only.
Simplifying assumption:  - Assume flat terrain; do not
include large natural features such as rivers, lakes,
cliffs, or elevation changes.
```

```
For each area: - Provide a unique ID (snake_case).
- Assign a scale. - Write a narrative focusing on
ARCHITECTURAL VIBE and BUILDING STYLES: materials,
construction methods, roof/facade styles, structural
elements, and distinctive exterior details. Keep it
grounded in what players see when walking through the
outdoor area.
```

Stage 2: Inter-area topology planning (LLM). We prompt an LLM to produce a connected world graph over areas, using only symbolic relative placement and discrete distance buckets. The deterministic compiler grounds these symbolic relations into tile-grid layouts.

```
You are a world topology planner for a procedural map
generator.
Return ONLY JSON that matches the provided JSON schema
(strict). Rules: - DO NOT output coordinates, positions,
or meters. - Choose one center_area_id. - Create
placements[] for ALL areas except the center: area_id,
relative_to (area_id or "center"), dir (N/NE/E/SE/S/SW/W/NW),
dist_bucket (near/medium/far). - Create connections[] that
forms a CONNECTED graph. - Aim for a tree (N-1 edges)
or tree + 1 loop. - Use connection kinds: trunk_road,
road, footpath. - For each connection, specify distance
(near/medium/far). - DO NOT specify boundary edges or gate
coordinates; the compiler grounds them. Goal: produce a
connected, traversable world topology.
```

Stage 4: Per-entity 3D asset prompt generation (LLM). Given an entity label, tags, and (when available) target footprint constraints from compilation, we prompt an LLM to output a short, material-focused description suitable for text-to-3D generators.

```
You generate visual, material-focused 3D asset descriptions
for text-to-3D generators.
Hard rules: - Describe ONLY the object (no people, no
story, no actions, no camera language). - 2--4 sentences
max. - Mention: primary materials, key shapes/forms,
surface condition, and distinctive details. - If
needs_frontage_any is true, include a clear front with an
entrance/door orientation detail. - If placement_dimensions
are provided, include approximate footprint and height. -
Keep it grounded in the provided context and tags.
Return ONLY valid JSON: "group": "...", "prompt": "..."
```

Frontage selection for directional assets (VLM). To decide which yaw view corresponds to the main facade, we query a VLM with a 2×2 grid of the same object rendered from four yaw directions and request the panel containing the entrance/main facade.

```
You are given a 2x2 grid image of the same 3D building
from four yaw directions. Define panels by position: A =
top-left, B = top-right, C = bottom-left, D = bottom-right.
Task: pick which panel shows the building's FRONT
(entrance/main facade). If the object is non-directional
or you cannot tell, answer 'NONE'.
Reply with ONLY one of: A, B, C, D, NONE.
```

A.2 Implementation / Compilation Details

A central design hypothesis of this work is that narrative world construction should not be delegated to a single monolithic language model output. While LLMs are effective at extracting semantic structure from text, they are not reliable mechanisms for enforcing geometric validity, connectivity constraints, collision avoidance, or executable assembly. We therefore decompose the problem into

semantic planning, topological organization, deterministic spatial compilation, and asset grounding. This separation yields a controllable and reproducible pipeline for generating executable 3D worlds from narrative text under strict compute constraints.

A.2.1 Strategic World Planning Details

The strategic planning stage produces a structured representation, `world_plan.json`, which encodes the semantic decomposition of the input narrative into distinct areas. Each area entry contains a localized narrative description, a symbolic scale label, and a list of entities required to instantiate the region.

Symbolic Scale Categories. Each area is assigned a discrete scale label selected from a fixed set: `tiny`, `small`, `medium`, `large`, and `huge`. These symbolic scales are deterministically mapped to predefined two-dimensional tile grid resolutions, which constrain downstream spatial layout generation while avoiding continuous size estimation.

Entity Specification. Entities are extracted as named semantic objects without spatial coordinates. This representation specifies *what* must exist within an area, but not *where*, ensuring that all geometric decisions are deferred to later deterministic compilation stages.

A.2.2 Topological Graph Construction Details

The topological world graph (`world_graph`) encodes both the relative placement of areas and explicit traversable connections between them. One area is designated as a global anchor, while all other areas are positioned relative to this anchor or to previously placed areas.

Symbolic Relative Placement. Relative placement is specified symbolically using a reference area, a compass direction, and a discrete distance category. These symbolic descriptors are deterministically mapped to tile-based geometric offsets using the scale mapping defined in Appendix A, enabling coarse spatial arrangement without metric inference from language.

Inter-Area Connectivity and Gates. Connectivity between areas is specified through named inter-area connections. Each connection defines a traversal type (e.g., trunk road, road, footpath) and the boundary edges of the source and destination areas on which the connection terminates. These boundaries define abstract *gates*, which serve as symbolic traversal interfaces and are grounded to exact tile locations during local layout compilation.

Overlap Resolution. After initial projection of symbolic placements, area bounding boxes are tested for overlap. If conflicts are detected, inter-area distances are deterministically increased until all layouts become disjoint. This rule-based resolution preserves global traversability while maintaining the narrative ordering encoded in the world graph.

A.2.3 Local Spatial Layout Compilation Details

Each semantic area is compiled into a local spatial layout defined over a discrete two-dimensional tile grid. The grid resolution is determined by the symbolic scale assigned during strategic planning, and a central anchor tile is used as the local coordinate origin.

Entity Placement via Relational Constraints. Entities are placed using relational placement constraints derived from semantic planning. Each constraint specifies a reference target, symbolic direction, discrete distance category, and a categorical footprint size. Entities are placed sequentially by projecting relative offsets from the reference target; if a placement violates grid bounds or overlaps an existing footprint, the offset distance is incrementally increased until a valid placement is found.

Gate Grounding. Traversal interfaces between areas are represented as symbolic gates and are grounded to deterministic tile locations along the boundary edges of the area grid. Gates serve as entry points for all intra-area road routing.

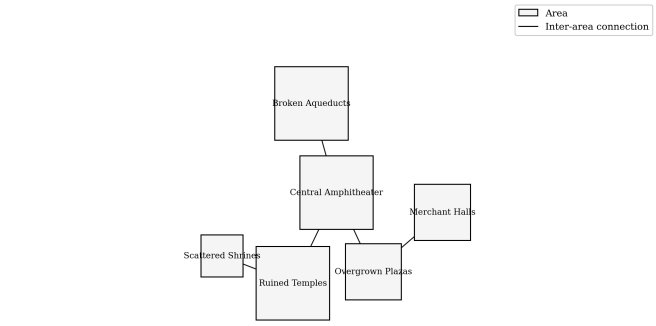
Road Network Construction. An intra-area road network is constructed directly on the tile grid to ensure connectivity between gates and entities. Routing is performed in multiple phases, including arterial connections to the area anchor, perimeter connections between adjacent gates, and secondary connections to individual entities.

Direction-Aware Pathfinding. Road segments are routed using an A*-based shortest-path search over the tile grid. Each search state is augmented with an incoming direction to model turn costs and directional preferences. The routing cost function incorporates penalties for sharp turns, proximity to entity footprints, and area boundaries, while encouraging reuse of existing road segments.

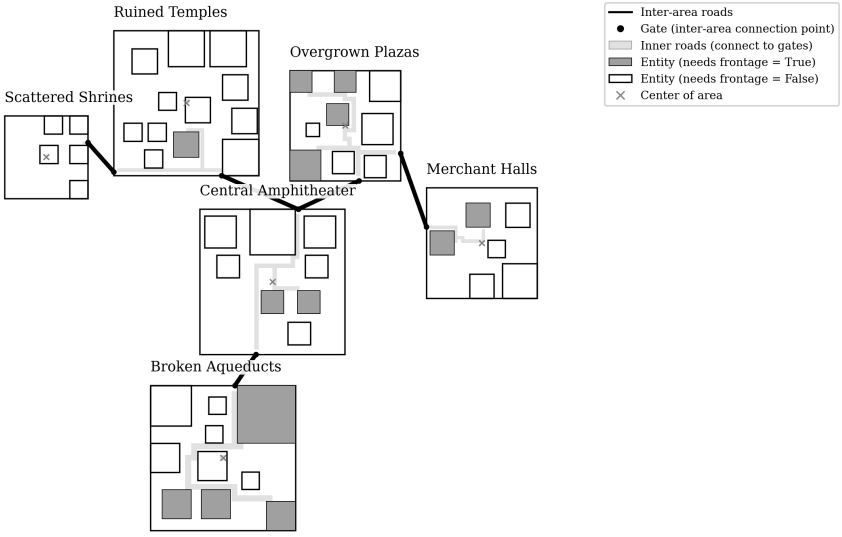
Road Materialization and Post-Processing. Once a path is computed, all tiles along the route are marked as road tiles. Road width is enforced by lateral expansion based on road type, followed by local post-processing steps including diagonal gap filling and corner smoothing to reduce discretization artifacts.

A.3 Qualitative Results

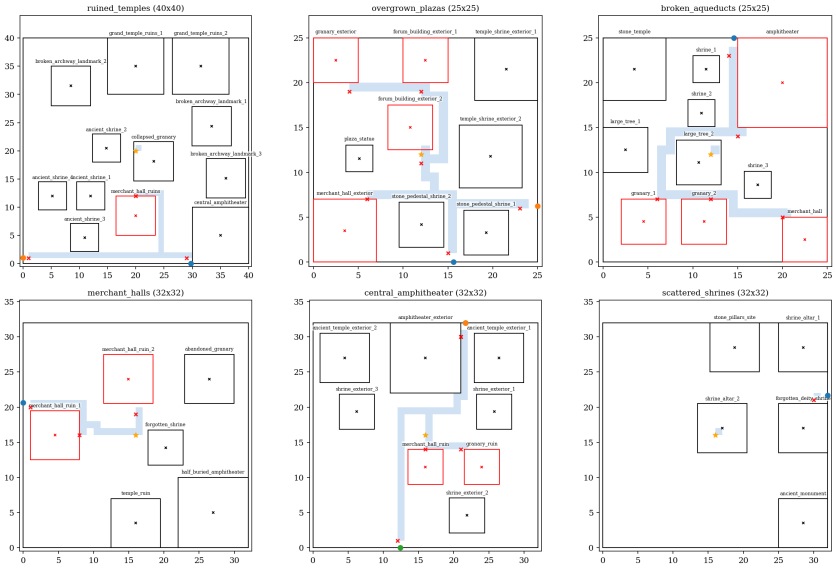
We report qualitative outputs for 5 representative prompts (out of 20), selected to cover diverse built environments. For each prompt, we include: (i) per-area local compilation (Areas), (ii) the inter-area topological graph, and (iii) the stitched global layout with inter-area roads.



(a) Inter-area graph

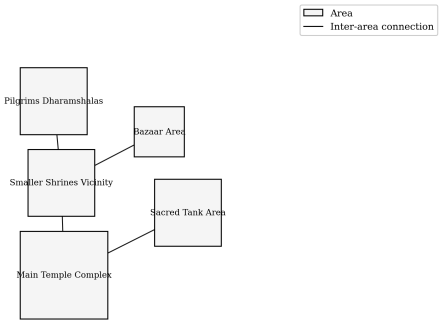


(b) Global stitched layout

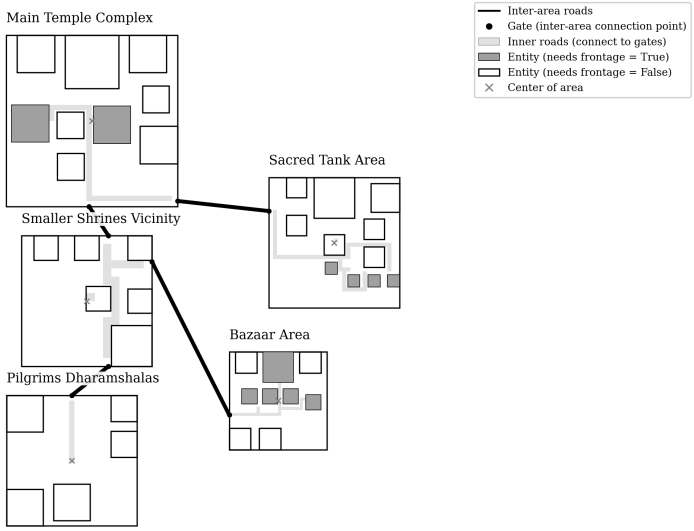


(c) Per-area local compilation

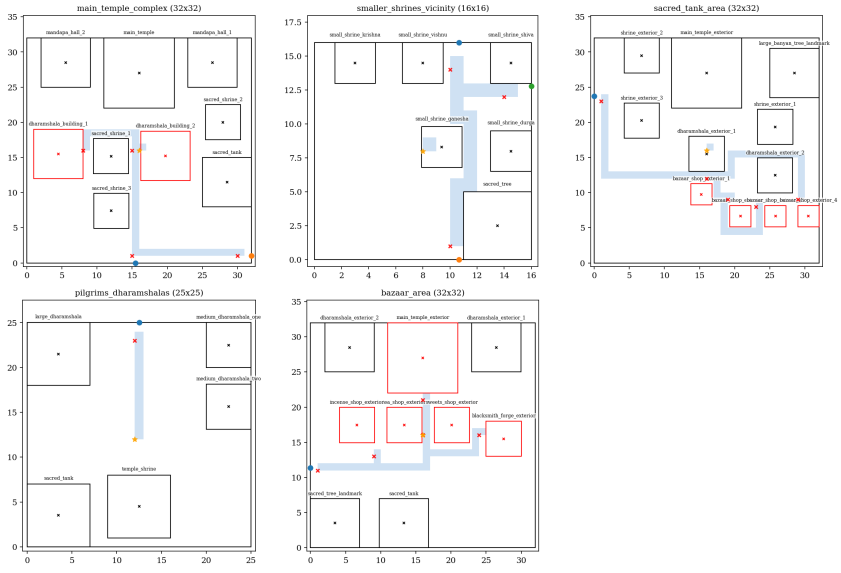
Figure 3: **Prompt 1 (world_00).** *An ancient civilization's ruins sprawl across a windswept plateau. Crumbling temples with weathered stone columns stand beside overgrown plazas. Broken aqueducts trace paths between collapsed merchant halls and abandoned granaries. A central amphitheater, half-buried in sand, hints at past gatherings. Scattered shrines mark forgotten deities, their altars still visible among the debris.*



(a) Inter-area graph

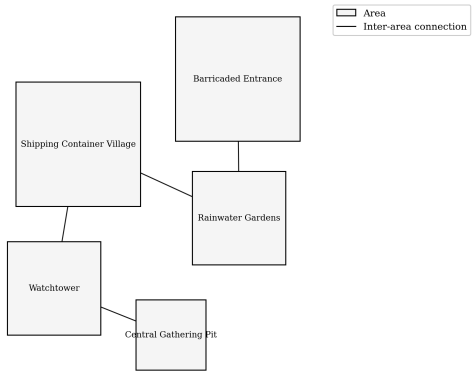


(b) Global stitched layout

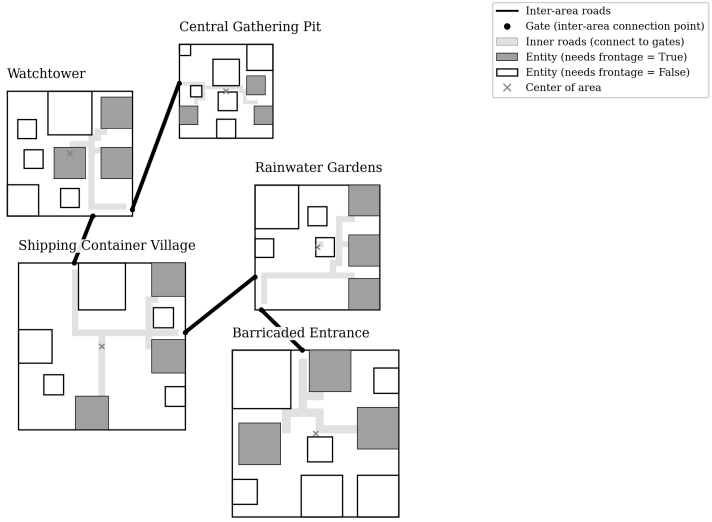


(c) Per-area local compilation

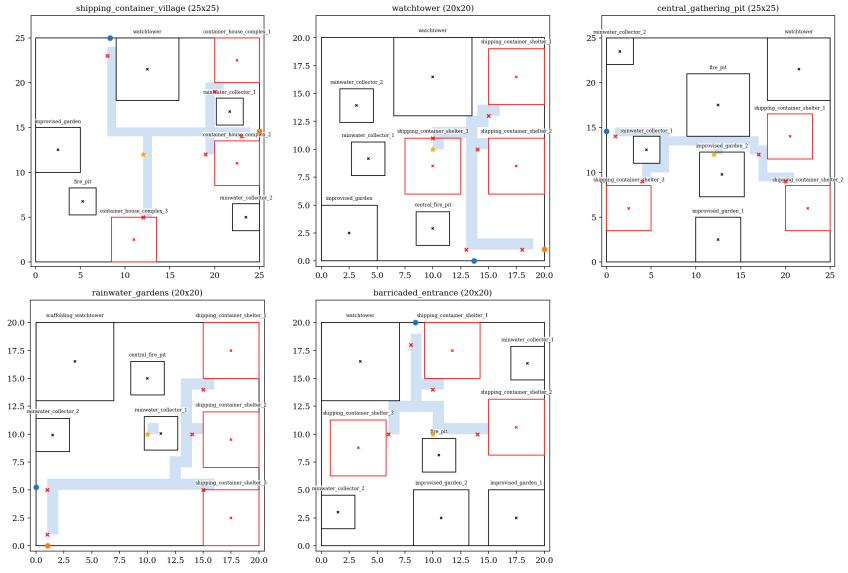
Figure 4: **Prompt (world_03).** *A sacred Indian temple complex rises from the landscape. The main temple features intricate stone carvings, gopuram towers, and mandapa halls. Smaller shrines dedicated to various deities surround the main structure. A sacred tank for ritual bathing lies to the east. Dharamshalas provide shelter for pilgrims, and a bazaar sells offerings and religious items.*



(a) Inter-area graph

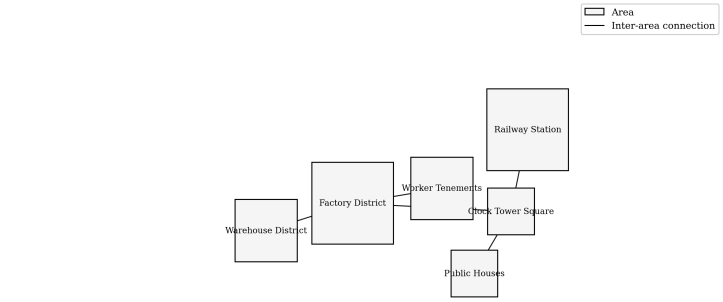


(b) Global stitched layout

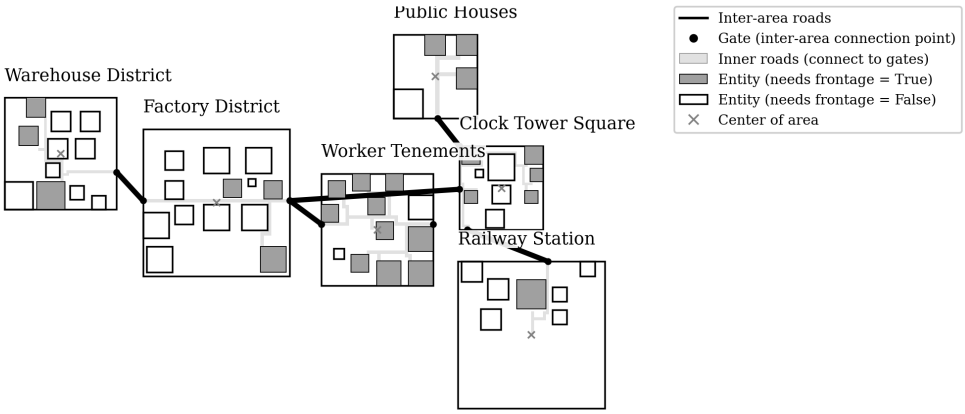


(c) Per-area local compilation

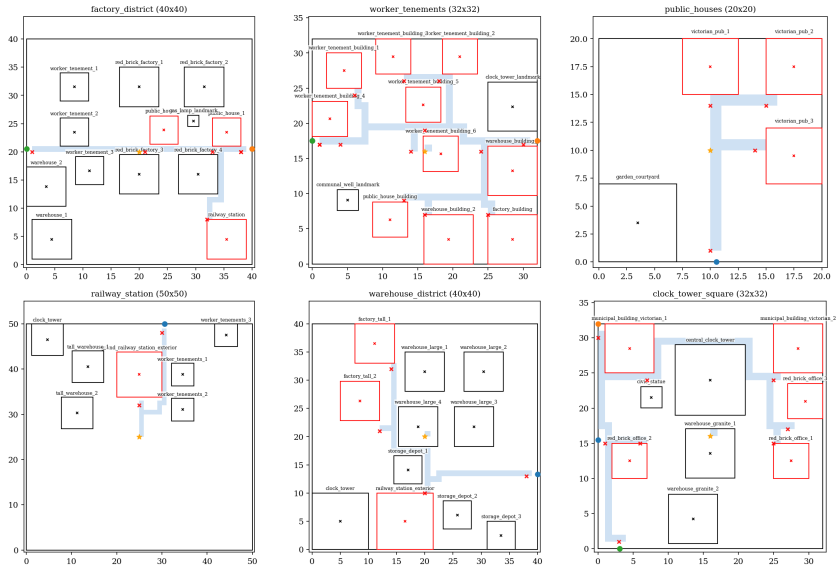
Figure 5: **Prompt (world_07).** *A survivor outpost built from the ruins of the old world. Repurposed shipping containers serve as shelters, reinforced with scrap metal. A watchtower made from scaffolding overlooks the perimeter. A central fire pit serves as the gathering place. Rainwater collectors and improvised gardens sustain the community. Barricades of wrecked vehicles protect the entrance.*



(a) Inter-area graph

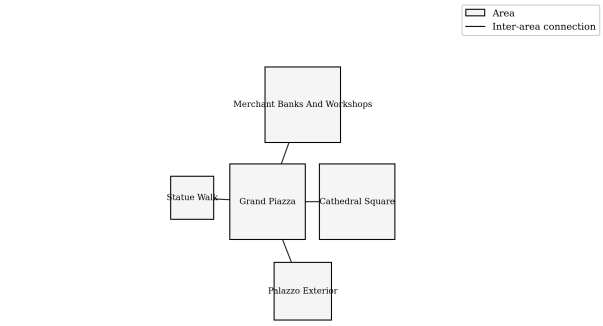


(b) Global stitched layout

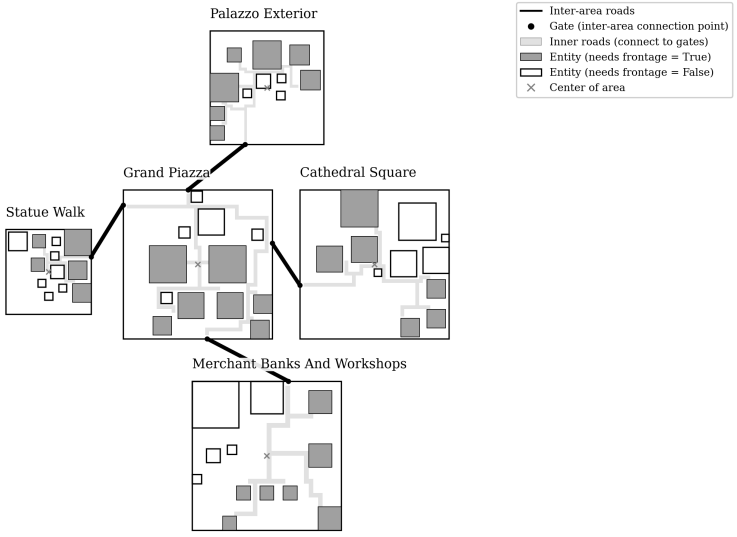


(c) Per-area local compilation

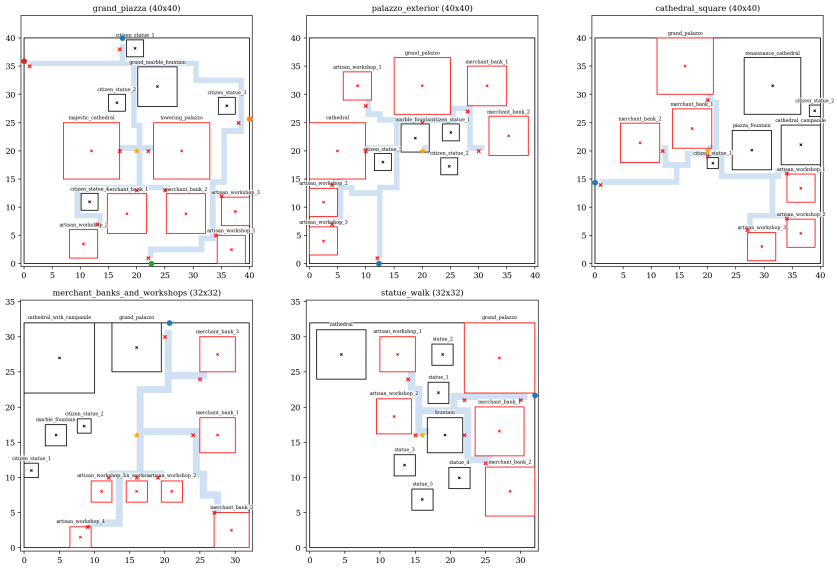
Figure 6: **Prompt (world_10).** *A Victorian industrial district shrouded in perpetual smog. Red-brick factories with tall chimneys dominate the skyline. Cobblestone streets are lined with worker tenements and public houses. A railway station connects to distant cities. Warehouses store goods from the empire. Gas lamps illuminate the fog, and a clock tower keeps time for shift changes.*



(a) Inter-area graph



(b) Global stitched layout



(c) Per-area local compilation

Figure 7: **Prompt (world_17).** *A Renaissance Italian piazza surrounded by elegant architecture. A grand palazzo with arched colonnades faces a marble fountain. A cathedral with a bronze-domed campanile dominates one side. Merchant banks and artisan workshops line the remaining edges. Statues of famous citizens adorn the square. Cafes and tavernas welcome visitors under painted awnings.*