

# SIMWORLD: An Open-ended Realistic Simulator for Autonomous Agents in Physical and Social Worlds

Jiawei Ren<sup>1\*</sup> Yan Zhuang<sup>2\*</sup> Xiaokang Ye<sup>1\*</sup> Lingjun Mao<sup>1</sup> Xuhong He<sup>3</sup> Jianzhi Shen<sup>4</sup>  
 Mrinaal Dogra<sup>1</sup> Yiming Liang<sup>5</sup> Ruixuan Zhang<sup>4</sup> Tianai Yue<sup>4</sup> Yiqing Yang<sup>6</sup> Eric Liu<sup>7</sup> Ryan Wu<sup>4</sup>  
 Kevin Benavente<sup>1</sup> Rajiv Mandya Nagaraju<sup>7</sup> Muhammad Faayez<sup>4</sup> Xiyan Zhang<sup>4</sup>  
 Dhruv Vivek Sharma<sup>1</sup> Xianrui Zhong<sup>3</sup> Ziqiao Ma<sup>8</sup> Tianmin Shu<sup>4†</sup> Zhiting Hu<sup>1†</sup> Lianhui Qin<sup>1†</sup>  
<sup>1</sup>UCSD <sup>2</sup>UVA <sup>3</sup>UIUC <sup>4</sup>JHU <sup>5</sup>Purdue <sup>6</sup>PolyU <sup>7</sup>USC <sup>8</sup>UMich


 <https://simworld.org>



Figure 1: **An Overview of the SIMWORLD Simulator**, featuring three key designs: (1) realistic, open-ended world simulation, (2) rich interface for LLM/VLM agents, and (3) diverse physical and social reasoning scenarios.

\* Equal contribution; † Equal advising

While LLM/VLM-powered AI agents have advanced rapidly in math, coding, and computer use, their applications in complex physical and social environments remain challenging. Building agents that can survive and thrive in the real world (e.g., by autonomously earning income or running a business) requires massive-scale interaction, reasoning, training, and evaluation across diverse embodied scenarios. However, existing world simulators for such development fall short: they often rely on limited hand-crafted environments, simulate simplified game-like physics and social rules, and lack native support for LLM/VLM agents. We introduce SIMWORLD, a new simulator built on Unreal Engine 5, designed for developing and evaluating LLM/VLM agents in rich, real-world-like settings. SIMWORLD offers three core capabilities: (1) *realistic, open-ended world simulation*, including accurate physical and social dynamics and language-driven procedural environment generation; (2) *rich interface for LLM/VLM agents*, with multi-modal world inputs/feedback and open-vocabulary action outputs at varying levels of abstraction; and (3) *diverse extensible physical and social reasoning scenarios* that are easily customizable by users. We demonstrate SIMWORLD by deploying frontier LLM agents (e.g., GPT-4o, Gemini-2.5-Flash, Claude-3.5, and DeepSeek-Prover-V2) on long-horizon multi-agent delivery tasks involving strategic cooperation and competition. The results reveal distinct reasoning patterns and limitations across models. We open-source SIMWORLD and hope it becomes a foundational platform for advancing real-world agent intelligence across disciplines: <https://simworld.org>.

## Table of Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>The SIMWORLD Simulator</b>	<b>4</b>
2.1	Unreal Engine Backend . . . . .	4
2.1.1	Diverse Scenes . . . . .	5
2.1.2	Rich Assets and Physics Realism . . . . .	6
2.2	Environment . . . . .	8
2.2.1	Procedural City Generation . . . . .	8
2.2.2	LLM-based Scene Editing . . . . .	9
2.2.3	Waypoint System . . . . .	9
2.2.4	Traffic System . . . . .	10
2.2.5	Gym-like Interface for Agent-Environment Interaction . . . . .	11
2.3	Agent . . . . .	12
2.3.1	Agent Framework . . . . .	12
2.3.2	Observation Space . . . . .	12
2.3.3	Action Space . . . . .	12
2.3.4	Action Planner . . . . .	13
2.4	UnrealCV+ Communication Module . . . . .	15
<b>3</b>	<b>Case Study: Delivery Task</b>	<b>15</b>
3.1	Task Formulation . . . . .	15
3.2	Main Results . . . . .	16
3.3	Ablation Study . . . . .	17
<b>4</b>	<b>Related Works</b>	<b>20</b>

---

## 1. Introduction

Large language and vision models (e.g., LLMs and VLMs) have emerged as powerful foundations for building intelligent agents, demonstrating remarkable reasoning capabilities, particularly in structured domains such as mathematics, coding, and computer use (e.g., web browsing). However, these *mathematical* and *digital* settings are relatively clean, with well-defined rules and clear feedback. In contrast, the embodied *physical* and *social* worlds, where real-world agents and robots are ultimately expected to operate, are inherently complex, noisy, dynamic, and unpredictable. In such environments, agents must interact with rich and evolving contexts, from navigating urban spaces and interacting with humans, to pursuing long-term goals such as earning a living, building a career, or running an organization (Brohan et al., 2023; Driess et al., 2023; Wang et al., 2023).

To advance embodied agent development, recent efforts have explored simulation environments that offer different interactive experiences for training and evaluation (Table 1). However, game-like platforms such as Minecraft (Fan et al., 2022; White et al., 2025; Wang et al., 2023; Long et al., 2024; Liu et al., 2024; Li et al., 2025b) and Pokémon (Ha, 2025; Anthropic, 2025) provide accessible setups for embodied interaction but lack realistic physical dynamics and social structures, limiting real-world generalization. Domain-specific simulators such as CARLA (Dosovitskiy et al., 2017) and AI2-THOR (Kolve et al., 2017) target areas like autonomous driving and household robotics but are limited to narrow task scopes or static environments. Social sandboxes such as Virtual Village (Park et al., 2023) and Project Sid (AL et al., 2024) simulate interpersonal interactions in scripted, small-scale communities, but lack the open-endedness and scalability required for modeling richer social complexity. Moreover, many of these environments do not support natural language interfaces for goal setting, planning, and control, limiting their compatibility with modern LLM-based agents.

To meet these growing demands, we present SIMWORLD, a platform designed to support the development and evaluation of autonomous agents in complex, dynamic, and interactive environments. SIMWORLD is grounded in three core design principles (Figure 1):

**1) Realistic, Open-Ended World Simulation.** SIMWORLD advances simulation by integrating two key aspects: *realistic* physical and social dynamics, and *open-ended, language-steerable* world generation. On the realism side, SIMWORLD produces complex, dynamic environments grounded in physical laws (e.g., gravity, momentum) and enriched with dynamic elements such as lighting, weather, and pedestrian flow in city-scale 3D scenes. It also embeds socially grounded behaviors, such as obeying traffic signals and maintaining personal space, directly into agent logic to support realistic interactions. On the open-ended side, SIMWORLD offers a broad range of scenes (e.g., city, countryside, wilderness, islands) and supports infinite environment expansion through procedural generation, including diverse road networks, building layouts, and urban configurations. Moreover, users or AI agents can modify scenes on-the-fly via natural language prompts (e.g., “add a tree next to the hospital”). Powered by SIMWORLD’s LLM-based editing and asset-generation modules, this capability enables adaptive, interactive world creation.

**2) Rich Interface for LLM/VLM Agents.** SIMWORLD provides a *Gym-like interface* that enables LLM/VLM agents to interact with simulated worlds using *open-ended natural language actions*. Agents can perceive rich multimodal observations (e.g., visual scenes, abstract layouts, and action feedback) and respond with high-level language commands. For example, an agent may reason and generate an abstract action, “sit on the nearest chair,” which SIMWORLD automatically decomposes into a sequence of low-level actions (e.g., navigating through waypoints, sitting down). After executing the actions, the simulator provides updated observations and feedback, allowing the agent to refine its strategy and continue reasoning. This closed-loop interaction supports open-ended, language-driven behaviors and empowers agents to perform long-horizon reasoning at a proper abstraction level.

**3) Diverse Physical and Social Reasoning Scenarios.** Building on the above physically and socially grounded environments and the agentic interface, SIMWORLD naturally supports systematic evaluation and training of agent reasoning in diverse realistic, long-horizon settings. Beyond short, task-oriented behaviors, agents can pursue extended objectives such as earning money, developing a career trajectory, or running a multi-agent business, where strategic decisions compound over time and social dynamics influence outcomes. To illustrate how these capabilities integrate in practice, we showcase a *Delivery Task*, a case study demonstrating how physical and social reasoning jointly lead to multi-agent collaboration and competition in SIMWORLD. The task models an urban delivery economy in which agents bid, invest, and share orders while navigating dynamic environments.

Table 1: **Comparison of SIMWORLD and Existing Simulators** across key dimensions: **Open-ended World** (procedural scene/asset generation, language-controllable editing), **Physical/Social Realism** (fidelity to real-world mechanics), **Action Space** (action abstraction level, open-vocabulary action space), **Agent Type** (types of controllable agents: Humanoid (Hum.), Robot, Drone or Vehicle (Veh.)), and **Physics Engine** (underlying simulation engine). **H** means high-level actions (e.g., “deliver”, “navigate to”), and **L** means low-level actions (e.g., “forward by 1 step”).

Simulator	Open-ended World		Physical/Social Realism	Action Space		Agent Type	Physics Engine
	Procedural	Lang.-Ctrl		Abstr.	Open-Vocab		
Minedojo (Fan et al., 2022)	✓	✗	+	L	✗	Hum.	Minecraft
Mindcraft (White et al., 2025)	✓	✗	+	H	✗	Hum.	Minecraft
MetaUrban (Wu et al., 2025)	✓	✗	++	L	✗	Veh.	PyBullet
EmbodiedCity (Gao et al., 2024)	✗	✗	+++	L	✗	Drone/Veh.	Unreal Engine
CARLA (Dosovitskiy et al., 2017)	✗	✗	+++	L	✗	Veh.	UE & Unity
GRUtopia (Wang et al., 2024a)	✗	✗	++	L	✗	Hum./Robot	Isaac Sim
OmniGibson (Li et al., 2024)	✗	✗	++	H/L	✗	Robot	Omniverse
AI2-THOR (Kolve et al., 2017)	✓	✗	++	L	✗	Robot	Unity
Habitat 3.0 (Puig et al., 2023)	✗	✗	++	L	✗	Hum./Robot	Bullet
Genesis (Authors, 2024)	✓	✗	+++	L	✗	Robot	Taichi
VirtualCommunity (Zhou et al., 2025)	✓	✗	++	L	✗	Hum./Robot	Genesis
UnrealZoo (Zhong et al., 2025)	✗	✗	+++	L	✗	Hum./Robot/Veh.	Unreal Engine
<b>SIMWORLD</b>	✓	✓	+++	H/L	✓	Hum./Robot/Veh.	Unreal Engine

With different personas, budgets, and tools (e.g., vehicles), agents develop diverse strategies shaped by their goals and changing conditions (e.g., fluctuating prices). The task highlights complex decision-making and long-horizon planning, where cooperation, competition, and emergent social behaviors arise naturally.

We deploy frontier LLMs as agents such as GPT-4o, Claude-3.5-Sonnet, Gemini-2.5-Flash, and others on the *Delivery Task*. We observe that Claude-3.5-Sonnet and DeepSeek-V3 earn the highest profits, but often behave erratically, such as overbidding on low-value orders or spending all their money on scooters they never use. In contrast, Gemini-2.5-Flash and DeepSeek-Prover-V2 follow more conservative, stable strategies, trading peak performance for consistency. Personality traits also shape agent behavior: conscientious agents focus on task completion, while open agents explore but frequently lose money. These findings expose both the strengths and limitations of LLM-based agents, while revealing rich, often unexpected behaviors that emerge from their interaction with complex environments.

We open-source SIMWORLD with the aim of establishing a foundational infrastructure for real-world agent research across disciplines. By supporting advanced LLM/VLM-based agents and enabling large-scale, realistic agent-environment and agent-agent interactions, SIMWORLD expands the capabilities of modern *agent-based simulation (ABS)*. This allows researchers in robotics, business, public health, social science, education, and beyond to study complex systems and emergent behaviors in rich, dynamic, and controllable environments. More details of the SIMWORLD project are available at <https://simworld.org>.

## 2. The SIMWORLD Simulator

Realistic, open-ended, and natively LLM/VLM-compatible simulators are crucial for advancing agent development in complex physical and social scenarios. SIMWORLD takes a step toward this goal through a three-tier architecture as illustrated in Figure 2. It separates the *Unreal Engine Backend* (§2.1) from two added Python layers: the *Environment* layer providing infinite environment generation and standard Gym-like environment-agent interface (§2.2), and the *Agent* layer supporting diverse input observations, open-ended output actions, and different reasoning/planning components (§2.3). In addition, the *UnrealCV+* communication module enables seamless interaction between the Unreal Engine backend and the Environment layer (§2.4).

### 2.1. Unreal Engine Backend

The Unreal Engine backend forms the foundation of SIMWORLD, providing high-fidelity rendering and physics simulation. It consists of three tightly coupled modules: (1) **Scenes** (§2.1.1) supporting both procedurally generated and curated maps; (2) **Asset Library** (§2.1.2) ensuring diverse and physically grounded content; and



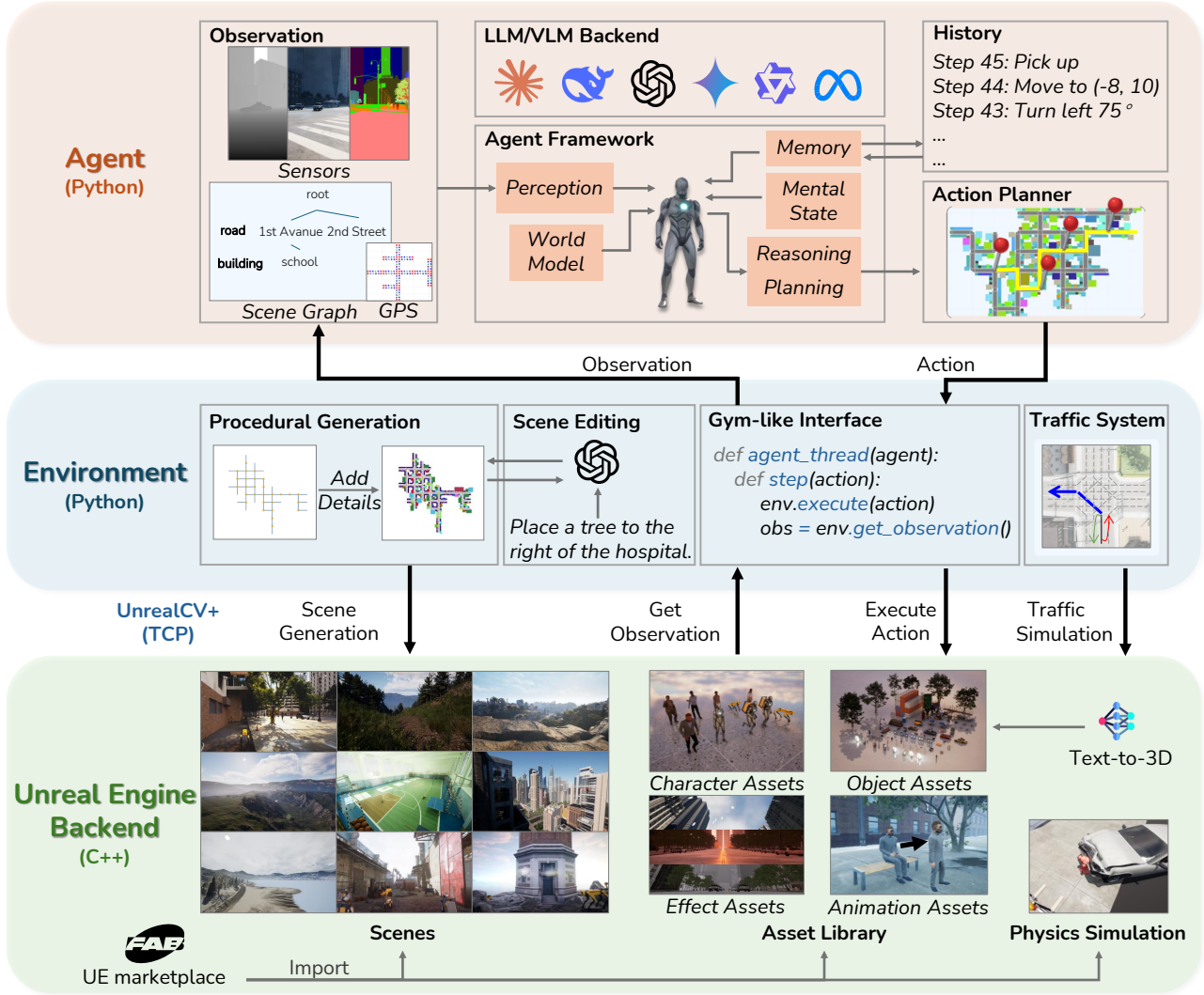


Figure 2: **Architecture of SIMWORLD.** SIMWORLD adopts a hierarchical, closed-loop architecture that decouples agent reasoning from high-performance rendering while maintaining coherent information flow across modules. At its core, the *Unreal Engine Backend* provides high-fidelity scenes, assets, and physics, serving as the foundation for realistic simulation. Built upon it, the *Environment* layer functions as an intermediary that abstracts the underlying rendering and physics into structured representations. It enables procedural city generation, traffic simulation, and exposes a Gym-like interface for agent interaction through *UnrealCV+*. The *Agent* layer operates on this interface, integrating LLM/VLM agents that interpret observations from the *Environment*, perform reasoning, and issue actions that are subsequently executed through the *Environment*’s connection to the *Unreal Engine Backend*, thereby forming a closed perception–planning–action loop.

(3) **Physics Simulation** (§2.1.2) governing realistic physical behaviors.

### 2.1.1. Diverse Scenes

SIMWORLD supports two scene-building modes: handcrafted scenes and procedurally generated scenes.

**Handcrafted Scenes.** Thanks to SIMWORLD ’s foundation in Unreal Engine, users can easily import a large collection of high-quality environments directly from the Unreal Engine Marketplace<sup>1</sup> or create custom scenes by

<sup>1</sup><https://www.fab.com/>



Figure 3: **Example Scenes in SIMWORLD.**

hand. In our current implementation, we curate over 100 handcrafted scenes<sup>2</sup> spanning a wide variety of visual and structural styles, from ancient towns and natural landscapes to futuristic cities and imaginative fantasy worlds. Each scene provides distinct visual cues, spatial layouts, and interaction affordances, enabling thorough evaluation of embodied agents across diverse settings. Figure 3 illustrates several examples.

**Procedurally Generated Scenes.** Complementing these handcrafted assets, SIMWORLD features a procedural generation module for automatically constructing diverse urban environments. Users can specify high-level parameters (e.g., city size, road density, layout style), and the system generates large numbers of city variants efficiently and consistently. This supports scalable experimentation under controlled, customizable conditions. Additional details are provided in Section 2.2.1.

By combining high-fidelity handcrafted scenes with flexible procedural generation, SIMWORLD offers a broad and extensible set of environments suitable for both controlled experiments and open-ended agent research.

### 2.1.2. Rich Assets and Physics Realism

SIMWORLD provides a comprehensive asset library to support realistic, physics-driven simulations across diverse environments. The system integrates static assets (e.g., buildings) and dynamic assets (e.g., pedestrians), and further incorporates environmental factors (e.g., lighting, weather) to create immersive virtual worlds. It also supports a wide range of animations and interactions, enabling agents to perform diverse actions faithfully within the environments.

**Object Assets.** The object asset library forms the structural backbone of SIMWORLD environments, where each scene can be viewed as a composition of multiple object assets. These assets include detailed material definitions and collision meshes, enabling a wide range of physically accurate interactions such as reflection, occlusion, and contact dynamics. Overall, they can be broadly grouped into three categories: (i) *Building assets*: Primary

<sup>2</sup>The SIMWORLD release includes all scenes as executable builds, available at <https://github.com/SimWorld-AI/SimWorld>.

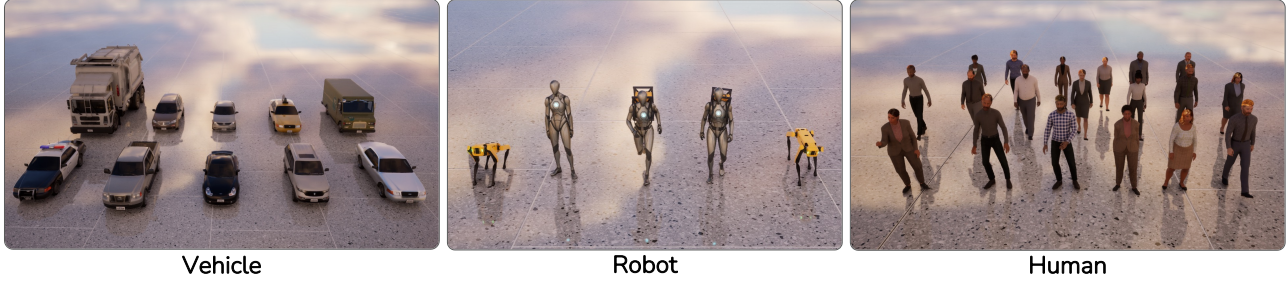


Figure 4: **Embodied Agents.** SIMWORLD supports three types of agent embodiments: vehicle, robot, and human.

structural elements of urban scenes, covering a wide range of architectural types (e.g., residential, commercial, industrial) and supporting both indoor and outdoor environment construction. (ii) *Vegetation assets*: Natural elements such as trees, grass, and shrubs, modeled with realistic material appearance and optional seasonal variations. (iii) *Urban prop assets*: Fine-grained objects such as benches, mailboxes, lampposts, and traffic signs, enabling diverse agent interactions such as sitting, opening, or manipulating objects.

**Text-to-3D Asset Generation.** To further expand the range of available objects, SIMWORLD introduces an *Asset Generation Pipeline* based on recent Text-to-3D models (Hunyuan3D, 2025). This system allows users to describe assets in natural language, automatically generating 3D objects with consistent scale, texture, and physical properties. The generated assets can be seamlessly integrated into the simulator, inheriting various properties (e.g., materials, lighting, collision configurations) compatible with UE’s physics engine.

**Characters and Embodiments.** Character assets in SIMWORLD represent embodied entities capable of acting, navigating, and interacting within the virtual environment. The system supports three primary types of agent embodiments: human, vehicle, and robot (Figure 4). Human embodiments capture diverse human appearances and employ fully rigged skeletal structures that enable realistic animations produced through coordinated bone articulation, such as running or carrying objects. Vehicle embodiments reproduce a range of real-world transportation modes (e.g., buses, cars) and implement accurate physical driving dynamics, such as acceleration, steering, braking, and traction. Robotic embodiments model specific categories of robots (e.g., quadruped systems) with realistic actuation, joint control, and sensing modules, making them suitable for evaluating robot locomotion and stability across different environments and tasks. All these embodiments operate within a unified physics framework and share common attributes (e.g., mass, inertia, contact forces), which ensures consistent handling of physical properties and interactions across all entities.

**Weather and Lighting.** SIMWORLD supports a wide range of lighting and weather conditions. The lighting system models multiple light types (e.g., directional, ambient, and dynamic sources) with controllable parameters such as intensity, orientation, and color temperature. The weather system supports a variety of conditions (e.g., rain, snow, and fog) that influence visual appearance and drive atmospheric effects, including phenomena like fog-induced light scattering. Together, these components recreate the complexity and dynamism of real-world environments, enabling the study of embodied agents’ perception and adaptation under realistic conditions.

**Physical Dynamics and Animations.** Powered by Unreal Engine, SIMWORLD provides accurate and continuous physical simulation. Unlike popular agent environments such as Minecraft (Fan et al., 2022; Yu et al., 2024), which rely on discrete, block-based mechanics without real gravity or inertia, SIMWORLD models real-world physical dynamics. Agents are subject to physical forces that produce grounded behaviors like sliding down slopes or tripping over steps. These effects produce physically grounded, embodied interactions. By combining Unreal Engine’s physics engine with physically informed animations (e.g., motion blending, inverse kinematics, collision responses), SIMWORLD maintains coherence between motion and environmental forces, enabling believable and adaptable agent behaviors in complex environments.



---

**Algorithm 1** Procedural City Layout Generation using QuadTree

---

```
1: Input: Configuration parameters config
2: Output: Final QuadTree  $Q_{city}$  representing the city layout
3: Initialize empty QuadTree  $Q_{city}$ 
4: if  $s = \text{road}$  then
5:   Generate road network via growth-based model                                ▷ Procedural street expansion
6:   Insert road geometries into  $Q_s$                                            ▷ Store road segments
7:   Merge  $Q_s$  into  $Q_{city}$                                                   ▷ Integrate road layout
8: if  $s = \text{building}$  then
9:   Sample building candidates (orientation, position)
10:  Reject invalid samples by collision test                                    ▷ Spatial consistency filtering
11:  Greedy fill remaining gaps with valid buildings
12:  Insert buildings into  $Q_s$ 
13:  Merge  $Q_s$  into  $Q_{city}$                                                   ▷ Integrate building layout
14: if  $s = \text{street element}$  then
15:   Sample decorative/environmental elements
16:   Reject overlapping samples by collision test
17:   Insert detail elements into  $Q_s$ 
18:   Merge  $Q_s$  into  $Q_{city}$                                                   ▷ Integrate street-level details
19: return  $Q_{city}$ 
```

---

## 2.2. Environment

SIMWORLD introduces an environment layer on top of the Unreal Engine backend (Figure 2). This layer manages the creation and organization of simulated environments and provides a clean abstraction that enables easy deployment of agents into Unreal Engine–based worlds through AI-native, user-friendly interfaces, without requiring users to handle the complexities of the underlying UE system. Specifically, the environment layer integrates modules for Procedural City Generation (§2.2.1), LLM-based Scene Editing (§2.2.2), Traffic Systems (§2.2.4), and, crucially, a Gym-like Interface (§2.2.5) for agent–environment interaction. It also offers an auxiliary Waypoint System (§2.2.3) that simplifies agent navigation within complex worlds.

### 2.2.1. Procedural City Generation

Previous simulators typically rely on a limited set of hand-crafted scenes (e.g., 15 scenes in CARLA and 211 scenes in Habitat 3.0). SIMWORLD develops a procedural generation system (Figure 5a) capable of producing diverse, unlimited urban environments, including road networks, building layouts, dynamic traffic, and fine-grained elements like street furniture, enabling effectively infinite simulation scenarios. All parameters (e.g., city size, building density, vehicle and pedestrian count) are customizable, allowing users to generate varied and controllable environments with minimal manual effort.

Inspired by (Phiresky, 2024), SIMWORLD’s procedural generation system adopts a modular and extensible architecture. The pipeline proceeds through three sequential stages: road generation, building generation, and street element generation, each progressively enriching the environment with structural and visual complexity. The system constructs a hierarchical scene graph based on a quadtree data structure as illustrated in Algorithm 1.

**Road Generation.** Road generation defines the structural backbone of the city layout. Roads are first initialized and then expanded using a spanning-tree–based algorithm with a priority queue that balances depth and branching during network construction. Additional procedures, including road-end attachment and intersection validation, maintain topological coherence and realism in the generated layout.

**Building Generation.** Following the road generation stage, the pipeline proceeds to building generation, where building assets are procedurally instantiated along road segments. Candidate locations are sampled and validated for spatial feasibility to prevent overlap. A greedy placement strategy then fills residual gaps near intersections and road ends, improving spatial utilization and maintaining visual consistency.



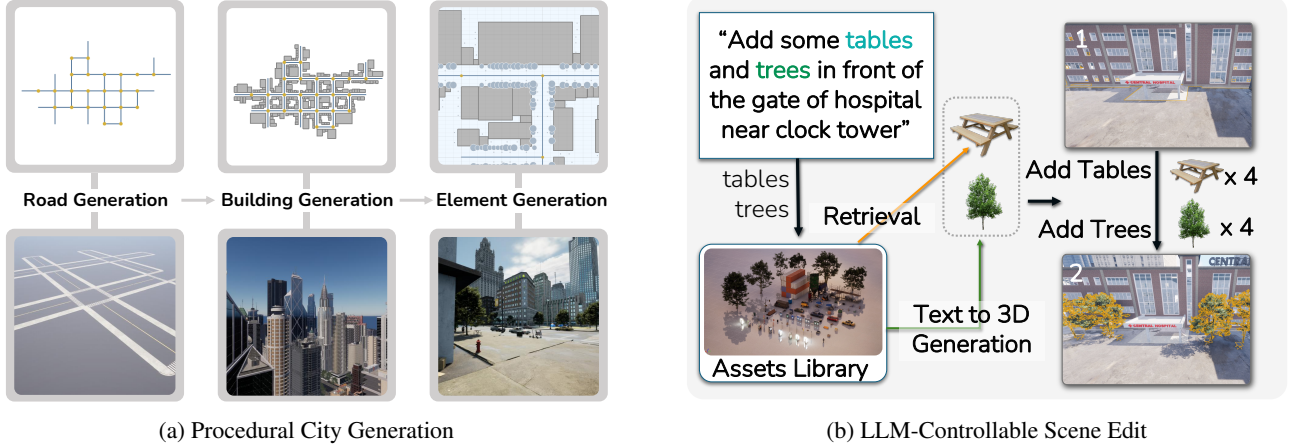


Figure 5: **Overview of Procedural City Generation and LLM-Based Scene Editing.**

**Street Element Generation.** Finally, street element generation adds detailed environmental elements (e.g., trees, road cones, benches, and parked vehicles). Elements are categorized and positioned based on contextual zones, either adjacent to buildings or along sidewalks. While strict collision enforcement is relaxed to maintain performance, placement still respects basic accessibility and spatial coherence constraints.

### 2.2.2. LLM-based Scene Editing

Beyond procedural generation, SIMWORLD supports natural language-based scene editing (Figure 5b), enabling dynamic world construction through open-ended instructions. Users or AI agents can modify scenes on-the-fly with commands such as “add a red sports car next to the hospital near a museum”. SIMWORLD contains a retrieval-augmented LLM-based scene agent that grounds the command by querying the current environment’s scene graph. The agent identifies the intended location using spatial anchors (e.g., “hospital”) and contextual landmarks (“museum”), retrieves a matching asset from a library, and inserts it accordingly. If a suitable asset is unavailable, the agent invokes an off-the-shelf text-to-3D generation model (Hunyuan3D, 2025) to synthesize a new object from the prompt (“red sports car”), converts it into a compatible format, and integrates it into the environment. This approach enables semantically grounded, spatially coherent, and scalable world construction, laying the foundation for interactive and compositional simulation.

### 2.2.3. Waypoint System

SIMWORLD implements a waypoint system that provides a structured representation of navigable space to support agent navigation and path planning (Figure 6). As an auxiliary abstraction layer, the waypoint system simplifies movement by offering a clean, graph-based representation of where agents can go and how they can get there. It forms the spatial backbone for both the traffic system (§2.2.4) and the action planner (§2.3.4), enabling agents to move efficiently through complex environments.

The system includes two complementary waypoint representations, coarse-grained and fine-grained, which together create a unified navigation graph. Coarse waypoints capture high-level connectivity (e.g., roads, intersections), while fine-grained waypoints represent detailed walkable paths. This hierarchical structure enables flexible and robust navigation behaviors, including lane following, turning, detouring, and obstacle avoidance.

**Coarse-grained Waypoints.** The coarse-grained waypoints are generated from the geometric outputs of the procedural city generation module (§2.2.1), including road centerlines and intersection coordinates. These waypoints represent major structural points within the road network and capture the primary connectivity among different routes.

**Fine-grained Waypoints.** The fine-grained waypoints are interpolated along the roads between coarse-grained waypoints. These additional points increase the density of the navigation graph, allowing agents to follow



Figure 6: **Overview of Waypoint System.** Vehicles and pedestrians navigate through the environment by following waypoints.

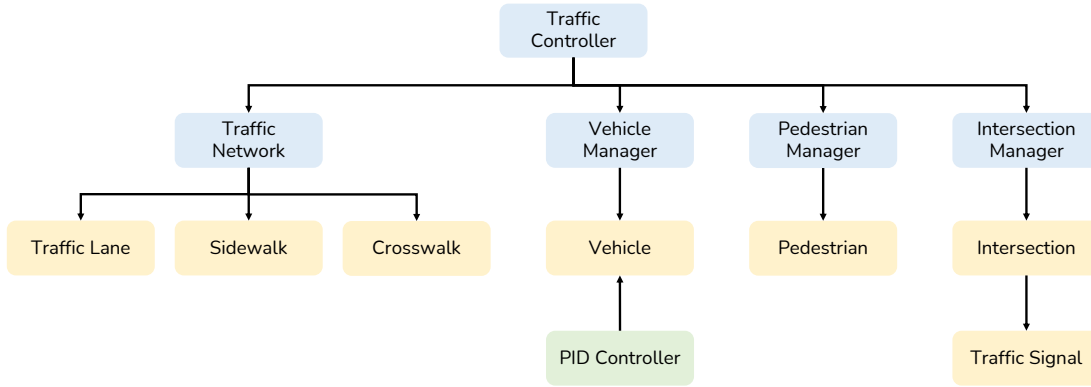


Figure 7: **Architecture of Traffic System in SIMWORLD.**

smoother and more continuous trajectories. Parameters such as interpolation step size and spatial offset magnitude can be customized by users.

#### 2.2.4. Traffic System

The traffic system in SIMWORLD simulates dynamic road usage involving both vehicles and pedestrians. It models realistic traffic flow through modules of vehicle spawning, route assignment, intersection control, and pedestrian movement (Figure 7). By managing interactions among agents and coordinating traffic signals, the system supports complex urban phenomena such as congestion, pedestrian crossings, and traffic light synchronization.

The traffic simulation supports route assignment, intersection control, and pedestrian flow simulation, running on a fixed-timestep update loop for consistent and deterministic updates (Algorithm 2). Vehicle motion is governed by a proportional–integral–derivative (PID) controller, with empirically tuned parameters for realistic acceleration, braking, and turning dynamics (Jain & Babel, 2024). Pedestrian motion follows a lightweight model that adjusts pedestrians’ orientations incrementally toward their goals based on angular differences. To simulate realistic patterns, SIMWORLD uses a stochastic routing policy at intersections, i.e., agents select outgoing routes according to predefined probability distributions. This stochastic behavior introduces natural variability and enhances scene diversity.

The traffic system is built upon the waypoint system (§2.2.3), enabling traffic simulation that generalizes to any procedurally generated city layout. Using the waypoints, the system calculates the detailed traffic areas (e.g., road lanes, sidewalks, and crosswalks) and procedurally instantiates vehicles, pedestrians, and traffic signals accordingly. Three specialized managers coordinate these processes:

**Vehicle Manager.** Vehicle manager initializes vehicles along designated traffic lanes and assigns either predefined or dynamically generated routes through the navigation network.

---

**Algorithm 2** Simulation Loop for Urban Traffic Environments

---

```
1: Initialize: Sample initial states for vehicles  $\mathcal{V}$ , pedestrians  $\mathcal{P}$ , and traffic signals  $\mathcal{S}$ .
2: Set simulation time  $t \leftarrow 0$ .
3: while  $t < T_{max}$  do ▷ Main simulation loop
4:   UPDATEVEHICLES( $\mathcal{V}, \mathcal{P}, \mathcal{S}$ )
5:   UPDATEPEDESTRIANS( $\mathcal{P}, \mathcal{V}, \mathcal{S}$ )
6:   UPDATESIGNALS( $\mathcal{S}, t$ )
7:    $t \leftarrow t + \Delta t$ 
10: function UPDATEVEHICLES( $\mathcal{V}, \mathcal{P}, \mathcal{S}$ )
11:   for all  $v \in \mathcal{V}$  do
12:     Perceive environment ( $\mathcal{V}, \mathcal{P}, \mathcal{S}$ )
13:     Execute driving model (throttle, brake, steering)
14:     Update  $v$ 's state (position, velocity)
16: function UPDATEPEDESTRIANS( $\mathcal{P}, \mathcal{V}, \mathcal{S}$ )
17:   for all  $p \in \mathcal{P}$  do
18:     Perceive ( $\mathcal{V}, \mathcal{S}$ ), execute walking logic
19:     Update  $p$ 's position
21: function UPDATESIGNALS( $\mathcal{S}, t$ )
22:   for all  $s \in \mathcal{S}$  do
23:     Update  $s$  according to timing plan or adaptive policy
```

---

**Pedestrian Manager.** Pedestrian manager spawns pedestrians on sidewalks and governs their motion patterns, including crossing behavior and local avoidance at intersections.

**Intersection Manager.** Intersection manager detects intersections within the traffic network and deploys traffic signals that regulate right-of-way according to configurable timing cycles or adaptive control policies.

Together, these components constitute a unified traffic simulation pipeline, enabling the virtual city to exhibit realistic, adaptive, and scalable mobility dynamics across diverse urban layouts.

### 2.2.5. Gym-like Interface for Agent-Environment Interaction

SIMWORLD provides a standard Gym-like interface, enabling seamless integration with existing reinforcement learning pipelines and agent frameworks. Because this interface follows the widely adopted API conventions of Gym (Farama Foundation, 2023), such as standardized `reset()`, `step()`, and observation–action exchange (Figure 2), it becomes straightforward for users to plug in their RL agents and immediately begin interacting with SIMWORLD’s simulated environments. This design significantly lowers the barrier for conducting large-scale experimentation, benchmarking, and agent–environment interaction studies using modern LLM/VLM or policy-based agents.

To support a broad variety of research goals ranging from open-ended simulations to highly controlled evaluations, SIMWORLD offers two simulation modes inspired by prior work such as CARLA (Dosovitskiy et al., 2017): asynchronous and synchronous execution.

**Asynchronous Mode.** In asynchronous mode, each agent runs in its own thread and advances independently, without waiting for other agents to finish their reasoning or action generation. Agents pull observations from a centralized buffer and submit actions whenever they are ready. The environment processes all received actions at fixed intervals (default: 0.1s), allowing real-time, continuous, and scalable multi-agent interactions. This mode is ideal for large-scale, open-ended, or exploratory simulations where throughput, diversity, and responsiveness are key.

**Synchronous Mode.** In synchronous mode, all agents advance in lockstep: the simulator proceeds to the next step only after every agent has submitted its action. This ensures strict temporal alignment between perception and control, making the mode particularly suitable for experiments requiring reproducibility, coordinated multi-agent

---

behavior, or high-quality data collection (e.g., video generation or RL training with fixed step timing).

## 2.3. Agent

SIMWORLD provides a unified interface for LLM/VLM agents, supporting a flexible **Agent Framework** (§2.3.1), a diverse **Observation Space** (§2.3.2) and an open-ended **Action Space** (§2.3.3). This interface is designed to accommodate both low-level control and high-level reasoning for LLM/VLM agents through an **Action Planner** (§2.3.4) module, serving as the primary entry point for agent–environment interaction.

### 2.3.1. Agent Framework

The agent framework in SIMWORLD defines a unified interface that structures the full agent loop of perception, reasoning, planning, and execution. Across different embodiments—humanoids, vehicles, and robots—the framework provides a common control pipeline.

Each agent first acquires observations from SIMWORLD’s observation space (§2.3.2) via API calls (e.g., `get_camera_observation()`, `get_agent_location()`). These observations (e.g., visual inputs, scene graphs) are then processed by the agent’s reasoning backend, which may incorporate LLMs, VLMs, VLAs, or other decision-making models. Based on these observations, agents can employ any advanced reasoning or planning algorithms (Guo et al., 2025; Hao et al., 2023; AL et al., 2024).

Reasoning outputs may be expressed in natural language (e.g., “sit on the nearest chair”) or in structured formats such as function calls. Both formats are compatible with the action planner (§2.3.4), which interprets them into executable low-level actions.

The framework is also highly extensible. Researchers can plug in advanced reasoning components such as world models (Hu & Shu, 2023; Xing et al., 2025), memory systems (Ho et al., 2025; Wang et al., 2024b), or mental-state modules (e.g., emotions or preferences), enabling broad investigation into long-horizon reasoning, planning, and embodied intelligence.

### 2.3.2. Observation Space

SIMWORLD provides multiple observation modalities for agent perception and reasoning. The observation space is organized into two primary categories: visual observations and structured semantic information (Figure 2).

**Visual Observations.** Agents can access three types of camera-based inputs from a first-person view: (1) color images capturing the raw visual appearance of the environment, (2) depth maps encoding geometric distance from the agent’s viewpoint, and (3) semantic segmentation masks providing pixel-level object category information.

**Structured Semantic Observations.** Beyond pixel-based perception, SIMWORLD exposes high-level spatial and semantic representations, including a semantic scene graph and GPS-like localization information. The scene graph encodes entities, attributes, and relational structures within the environment, offering a symbolic abstraction of the 3D world. The localization interface specifies each agent’s or object’s position and orientation, enabling precise reasoning about spatial relationships.

### 2.3.3. Action Space

SIMWORLD enables open-vocabulary action execution by organizing the action space into two hierarchical layers: high-level semantic actions and low-level primitive actions:

**High-Level Semantic Actions.** To facilitate abstract reasoning and long-horizon decision-making, agents can issue natural language commands. These commands are interpreted and executed by the built-in action planner (§2.3.4), enabling flexible, open-ended behaviors (e.g., “sit on the nearest chair”).

**Low-Level Primitive Actions.** Primitive actions provide fine-grained control over agents. Vehicles support continuous control signals (e.g., “acceleration”, “braking”, and “steering”). Robots allow continuous translation and rotation (e.g., “forward”, “backward”, “lateral\_movement”, and “rotation”). Human



Table 2: **Low-Level Primitive Actions in SIMWORLD.**

Action	Agent Type	Description
<i>Object Interaction Actions</i>		
Pick Up / Drop Off	Humanoid	Grasp or release an object
Carry / Put Down Heavy Object	Humanoid	Transport and place large objects
Sit Down / Stand Up	Humanoid	Transition between seated and standing states
Open Door / Enter / Exit Car	Humanoid	Interact with doors or vehicles
Ride Scooter	Humanoid	Control and ride a scooter
<i>Observation Actions</i>		
Look Up / Down	Humanoid, Dog	Adjust gaze vertically
Focus	Humanoid, Dog	Narrow or widen the field of view
Take Photo	All	Capture current view as an image
<i>Social Actions</i>		
Have Conversation / Discuss	Humanoid	Exchange verbal communication
Point Direction / Wave Hand	Humanoid	Use gestures for social signaling
Argue with Body Language	Humanoid	Express disagreement through gestures
<i>Navigation Actions</i>		
Move Forward / Step Forward-Backward	Humanoid, Dog	Move or step in the current direction for a short duration
Rotate / Steering	Humanoid, Dog, Vehicle	Adjust facing or steering direction
Throttle / Brake	Vehicle	Accelerate or decelerate the vehicle
Stop	All	Halt all current motion

agents can navigate (“move”, “turn”) and perform interactive actions, including human–object (e.g., “pick\_up”, “drop”, “sit”), human–vehicle (e.g., “enter\_car”, “exit\_car”), and human–human (e.g., “wave\_hands”, “discussion”) interactions. A complete list of supported actions is provided in Table 2.

#### 2.3.4. Action Planner

SIMWORLD includes an action planner module that bridges high-level reasoning with low-level execution, allowing researchers to focus on abstract planning without needing to manage embodiment-specific control details. The planner consists of two components: a *parser* and an *executor*. The parser receives high-level plans from the agent, often expressed in natural language or structured function calls, and translates them into sequences of low-level primitive actions. The executor then carries out these actions step by step, conditioned on the current environment state.

To support diverse research objectives, SIMWORLD provides two executor variants: a *rule-based* executor, which operates on abstract city-layout information, and a *visual-based* executor, which directly consumes visual observations from the simulator. The latter enables seamless integration with VLMs or VLAs, supporting end-to-end perception–reasoning–action pipelines.

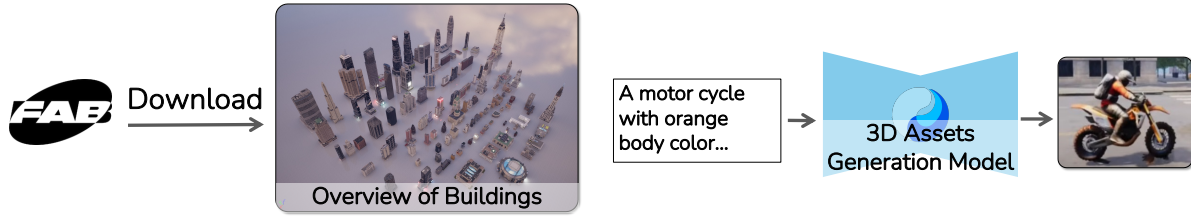
By handling the translation from high-level intent to low-level control, the action planner enables agents to perform long-horizon, semantic planning while SIMWORLD automatically manages movement, navigation, and interaction details.

For example, when the action planner receives a plan such as “go to the nearest chair and sit down”, the parser first decomposes the instruction into an action list: “navigate” and “agent\_sit\_down”. The navigate action is non-atomic and can be further expanded into primitive operations such as “step\_forward” and “rotate” by executor. In the *rule-based* execution mode, the planner computes the shortest path from the agent’s current position to the nearest chair, generating a sequence of navigation primitives such as `navigate(0, 1)`, `navigate(1, 10)`, and `navigate(10, 10)`, where `(10, 10)` denotes the chair’s location. Once the agent reaches the chair, the executor executes “agent\_sit\_down” and terminates when the action list becomes empty. In the *visual-based* mode, the executor directly feeds environmental observations into a VLM (e.g., GPT-4o), which determines the next action step by step (e.g., execute “step\_forward” then “step\_forward” and finally “agent\_sit\_down”) based on visual context.

## 1. Assets Preparation

Download from FAB Marketplace

Generated by assets generation models



## 2. Scene Preparation

Load layouts to Unreal



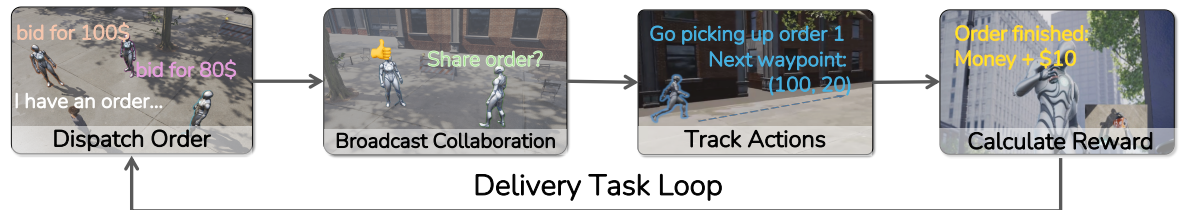
## 3. Spawning Agent

Spawn vehicles, pedestrians and agents step by step



## 4. Simulation

Managing and control task pipeline



## 5. Evaluation

Evaluate agents capabilities

Analysis agent behaviors

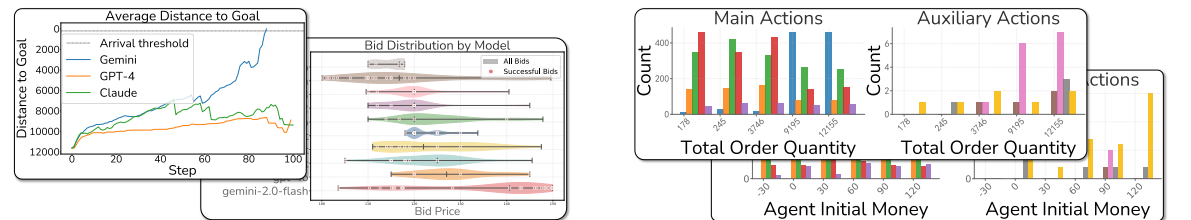


Figure 8: Workflow for Constructing a Task Suite in SIMWORLD.

---

## 2.4. UnrealCV+ Communication Module

Inspired by UnrealCV (Qiu et al., 2017), we develop UnrealCV+ as a communication module that bridges the Unreal Engine backend (§2.1) with the environment layer (§2.2). By establishing a Transmission Control Protocol (TCP) connection, UnrealCV+ enables efficient, reliable, and bidirectional communication between the two sides. Implemented in both Python and C++, the module supports flexible data exchange and fine-grained control of the simulation.

To adapt UnrealCV+ to agent-task scenarios, we introduce a customized command set for scene control, actor manipulation, and data querying. For example, the environment layer can issue commands to the Unreal Engine backend such as “spawn actors at locations”, “get position of a pedestrian”, or “execute action of a robot”.

Within the simulation loop, the environment layer governs the logical evolution of the simulation, while the Unreal Engine backend continuously returns updated physical states and visual observations of all agents. All communications are transmitted through UnrealCV+. This decoupled architecture cleanly separates logic computation from rendering, improving flexibility, scalability, and modularity.

## 3. Case Study: Delivery Task

SIMWORLD is built with extensibility and evaluation task creation as core design goals. It provides tools to easily define tasks, agent roles, reward functions, and evaluation metrics, minimizing the engineering effort required to create new experiments. Figure 8 showcases how to define a multi-agent delivery task within the SIMWORLD ecosystem. The process is delineated as follows:

First, users prepare assets for the delivery task, either by importing 3D assets purchased from third-party marketplaces or by generating them using text-to-3D models. Next, SIMWORLD initializes the city environments via its procedural city generation module to build roads, buildings, trees, and city props. Third, vehicles, pedestrians and delivery agents are spawned. Fourth, the simulation runs, allowing delivery agents to act and interact within the configured environment dynamics. Finally, delivery agents are evaluated across multiple dimensions, such as delivery success rate, average completion time, and total profit. The following sections provide detailed definitions and analysis. The eventual architecture of the delivery task is shown in Figure 9.

### 3.1. Task Formulation

The delivery task is designed to evaluate the social reasoning capabilities of foundation models in realistic, open-world urban environments. LLM agents are deployed as delivery agents in a city-scale environment built using SIMWORLD. Their goal is to grow and thrive in this dynamic setting. To enhance realism and complexity, the environment incorporates several dynamic systems: (1) an energy system, where agents must manage stamina and replenish it through consumables; (2) an economic system, where agents earn and spend currency on purchases such as scooters and drinks; (3) an order-sharing mechanism, enabling agents to collaborate by sharing delivery tasks and optimizing group performance. These components create a rich, interactive simulation environment for evaluating agents’ decision-making, adaptability, and social reasoning in complex urban scenarios. The overview of the delivery task is shown in Figure 10.

**Environment.** All experiments are conducted on the same map generated by the procedural city generation module in SIMWORLD. To ensure fair and efficient evaluation, graphical rendering is disabled during simulation. However, the physical simulation and evaluation modules remain active to preserve environment dynamics and task fidelity.

**Action Space.** The task features a two-tiered action space: high-level actions decided by LLM agents and low-level actions executed by the SIMWORLD action planner. At each decision step, the agent may choose: **Bid Order** (offer a price to compete for a new order), **Pick Up Order** (navigate to the pick-up point), **Deliver Order** (complete delivery at the destination), **Share Order** (publish the order for collaboration), **Purchase Scooter** (purchase a scooter and equip it to move faster). The full action space is shown in Table 3.

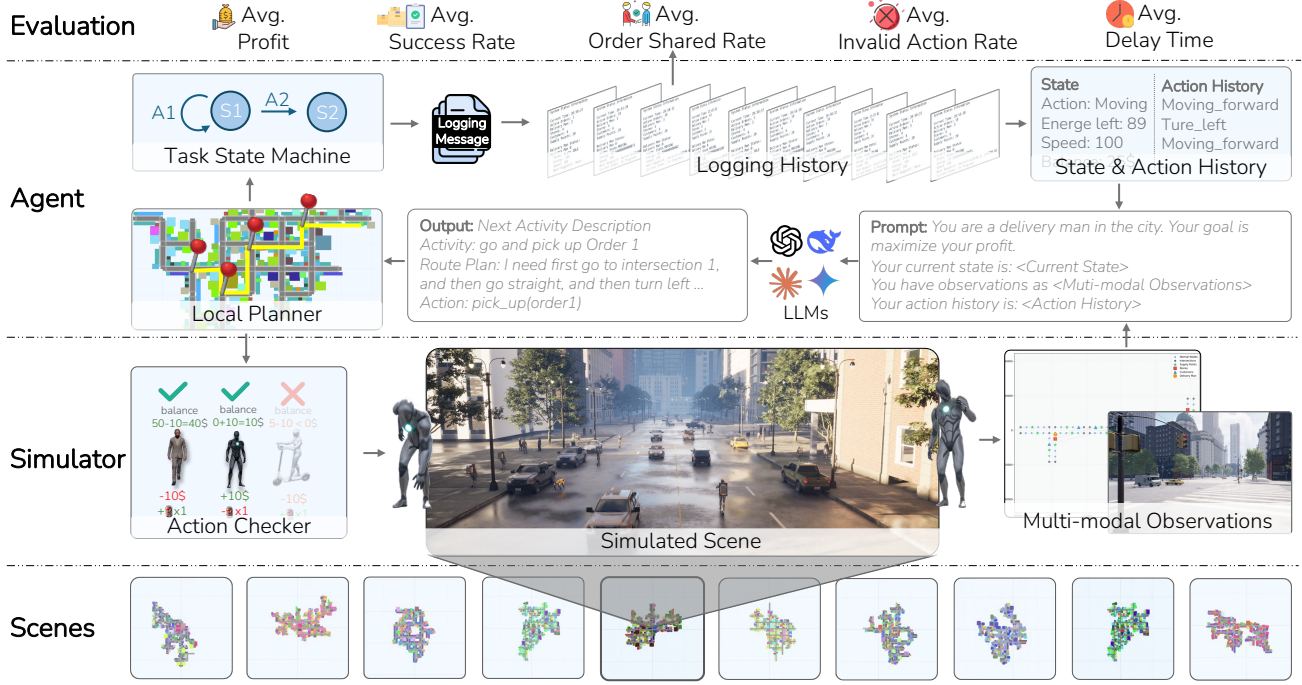


Figure 9: Architecture of the Delivery Task.

**Baseline Agents.** We evaluate multiple foundation models serving as the backbone of delivery agents, including Claude-3.5-Sonnet, DeepSeek-V3, GPT-4o, Gemini-2.5-Flash, and QWQ. The ReAct (Yao et al., 2022) prompting framework is employed to explicitly separate reasoning and action selection.

**Metrics.** Aligned with the agent’s hierarchical decision-making, we design a three-level evaluation framework. Overall performance is measured by **total profit** (the cumulative monetary gain the agent achieves over the simulation period), while operational effectiveness is assessed using **order success rate** (the proportion of orders that the agent successfully completes relative to the total assigned orders), **energy efficiency** (the ratio of energy consumed to revenue generated), **order sharing count** (the number of sharing orders), and **investment count** (the number of strategic investments).

### 3.2. Main Results

For each evaluation, we sample 20 agents controlled by the same language model, each running for 5000 simulation steps. At each step, an agent issues two API requests, averaging around 7000 tokens per request. Based on results in Table 4, our empirical analysis over three simulation rounds reveals distinct operational behaviors across models.

DeepSeek-V3 ( $69.475 \pm 16.772$ ) and Claude-3.5-Sonnet ( $69.068 \pm 20.685$ ) achieved the highest mean profits, with Claude-3.5-Sonnet also leading in mean successful orders ( $2.733 \pm 1.102$ ) and energy efficiency ( $0.5411 \pm 0.1981$ ). Notably, these superior average outcomes were associated with substantial performance variability, as reflected in their respective standard deviations.

Conversely, Gemini-2.5-Flash, while attaining a moderate mean profit of 42.423, exhibited markedly more consistent profit generation with a standard deviation of  $\pm 3.103$ , and also demonstrated stability in successful orders ( $2.100 \pm 0.173$ ). Extreme variability was evident in specific metrics for certain models; for instance, sharing counts for DeepSeek-Prover-V2 ( $7.333 \pm 8.386$ ) and Claude-3.5-Sonnet ( $11.333 \pm 8.386$ ) showed standard deviations exceeding their means, indicating highly unpredictable behavior in this aspect.



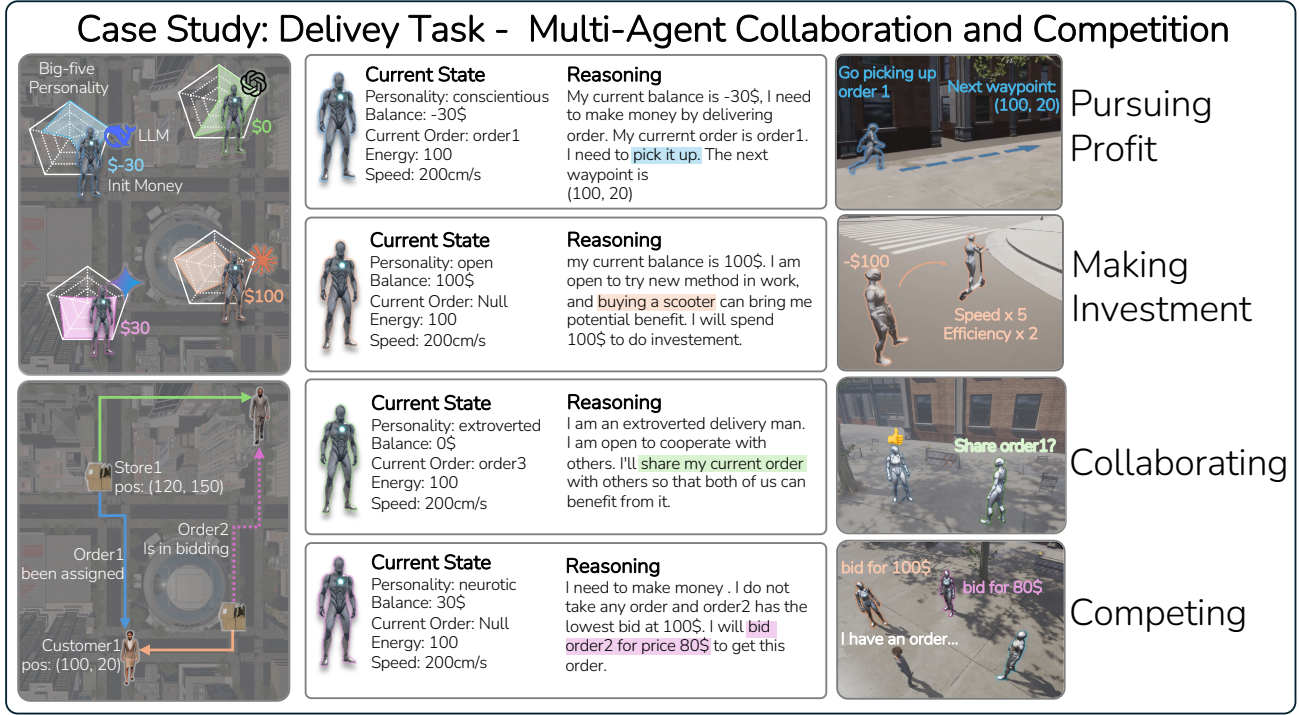


Figure 10: **Delivery Task**. A delivery scenario requiring multi-agent collaboration and competition. Each agent is initialized with distinct personalities and internal states and can act to grow, thrive, and ultimately maximize its earnings.

The GPT-4o-mini model consistently yielded zero values across all metrics ( $0.000 \pm 0.000$ ), suggesting it does not truly understand the goals well enough to make reasonable decisions based on the given instructions and context.

Higher investment strategies, such as those adopted by DeepSeek-V3 ( $8.000 \pm 3.000$ ) and Claude-3.5-Sonnet ( $9.000 \pm 3.464$ ), generally correlated with greater mean profit achievement but also with increased outcome volatility. These findings underscore a prevalent trade-off between optimizing for peak average performance metrics and ensuring consistent, predictable agent behavior, a critical consideration for robust deployment in dynamic environments.

#### Takeaway: Model Performances in Multi-agent Tasks

Top-performing models such as DeepSeek-V3 and Claude-3.5-Sonnet achieve high average profits but show greater variance, whereas Gemini-2.5-Flash demonstrates more consistent yet moderate performance. GPT-4o-mini failed entirely across all metrics. Overall, the results highlight a trade-off between maximizing average performance and ensuring consistent, reliable behavior (Table 4).

### 3.3. Ablation Study

To take a step deeper on multi-agent collaboration and competition, we conduct three ablation experiments. In the **Model Competition** setting, we sample 24 agents controlled by 12 models, with each model managing two agents over 1000 rounds. In this ablation experiment, we study how models make choices within a highly competitive environment in order to maximize their returns; In the **Environment Configuration** setting, we vary two environment configurations: the initial financial budget and the global order volume. For each configuration, we sample several stages from low to high to observe how agents' behavior changes with the environment conditions; In the **Persona** setting, we use the model with the best performance to control the agents with persona

Table 3: **Hierarchical Action Space Design in Delivery Task.** High-level actions are given to language models to make decision, which correspond to strategic decisions, while low-level actions are only exposed to local action planner module to execute concrete movements and interactions.

Action Level	Action Name	Description	Invocation Method
High-Level	Bid Order	Offer a price to a new order on platform to compete with other Model Generations	Model Generation
	Pick Up Order	Navigate to the pick-up point of order	
	Deliver Order	Navigate to the delivery point and complete the order	
	Share Order	Publish the order for multi-model generation cooperation	
	Cancel Share Order	Cancel a shared order that has been published	
	Go to Meet-point	Navigate to the meet point for the shared orde	
	Purchase Scooter	Buy and use a scooter	
	Purchase Drinks	Buy consumables to restore energy	
Low-Level	Adjust Speed	Adjust travel speed	Action Planner
	Move Forward	Basic movement action	
	Stop	Stop moving	
	Rotate	Adjust the facing direction	
	Change Speed	Adjust walking speed	
	Drive Scooter	Control a scooter for movement	

Table 4: **Performance of Model-Controlled Agents.** Metrics are reported as mean (Avg) and standard deviation (Std) over three 5000-step simulations. Bold indicates the best Avg per column.

Model	Profit		Successful Orders		Energy Efficiency		Sharing Count		Investment Count	
	Avg	Std	Avg	Std	Avg	Std	Avg	Std	Avg	Std
DeepSeek-V3	<b>69.48</b>	16.77	2.10	0.47	0.34	0.07	2.33	0.47	8.00	3.00
Claude-3.5-Sonnet	69.07	20.69	<b>2.73</b>	1.10	<b>0.54</b>	0.20	<b>11.33</b>	8.39	<b>9.00</b>	3.46
GPT-4o	43.91	14.16	1.63	0.43	0.30	0.06	0.67	0.47	4.67	0.47
Gemini-2.5-Flash	42.42	3.10	2.10	0.17	0.17	0.04	2.67	1.25	2.00	2.00
Gemini-2.0-Flash	28.72	12.04	1.53	0.58	0.11	0.03	0.67	0.47	0.67	1.00
Qwen3-32B	24.73	7.95	1.37	0.13	0.40	0.17	1.33	0.47	5.33	2.06
DeepSeek-Prover-V2	21.66	7.18	0.67	0.14	0.42	0.03	7.33	8.39	1.00	1.00
QwQ	17.31	4.07	0.87	0.20	0.41	0.20	0.33	0.47	3.33	2.52
GPT-4o-mini	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

description in prompts. We sample 20 agents, assign each persona to two agents, and observe how these personas shape the agents’ behaviors and decision strategies.

**Model Competition.** To intensify inter-agent competition, we constrain each agent to handle at most one order at a time and set the environment’s hunger rate to 0.9, ensuring a high demand for delivery. In each experimental session, 24 agents are jointly controlled by 12 different models, where each model governs two agents. These agents actively bid for orders in a shared environment with the goal of maximizing profit. Each session runs for 1000 simulation steps, and results are averaged across three random seeds.

As shown in Figure 11a, models exhibit distinct bidding behaviors. Notably, Claude-3.7-Sonnet, Gemini-2.5-Flash and Gemini-2.0-Flash demonstrate broad bid price distributions, indicating a flexible bidding strategy. This flexibility increases their chances of winning orders when in competition with other models. In contrast, models such as LLaMA-4-Scout and LLaMA-3.2-11b tend to use narrower bidding ranges, which limits their competitiveness and results in lower win rates.

Figure 11b presents the head-to-head competition outcomes. Deepseek-Prover-V2 and Qwen3-32B achieve the highest win rates against other models. This is primarily because they often bid lower prices, making their offers more likely to be accepted by the platform. Conversely, models like GPT-4o and LLaMA-3.2-11b tend to place higher bids, reducing their success rate despite frequent participation. Models such as QwQ-32B and GPT-4o-mini are less active overall, leading to fewer bids and lower order acquisition rates. This inactivity contributes to their diminished final profit, as shown in Table 4.

### Takeaway: Multi-Agent Competition

Models with flexible bidding strategies, like Claude-3.7-Sonnet and Gemini-2.5-Flash, achieve higher order win rates, while those with narrow or high bids, like LLaMA-3.2-11b and GPT-4o, underperform. Models that bid aggressively, such as Deepseek-Prover-V2 and Qwen3-32B, dominate head-to-head competitions, whereas inactive models like GPT-4o-mini fail to secure bids and profits. QwQ and GPT-4o-mini show minimal bidding activity and weak task participation (Figure 11).

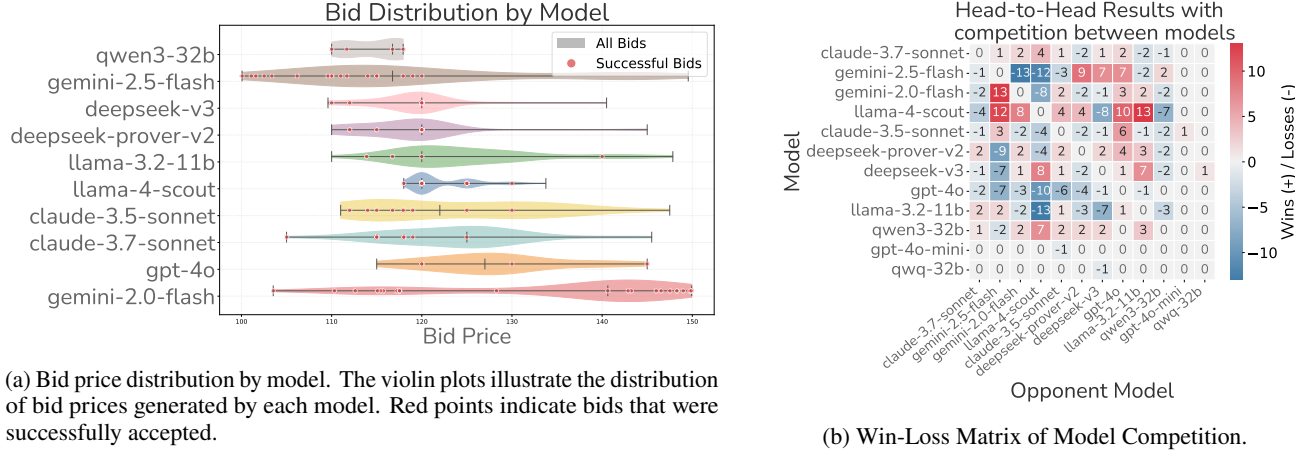


Figure 11: **Bidding Behavior and Evaluation Results.** (a) Lower bid prices may increase the likelihood of being assigned an order, but often come at the cost of reduced profit margins. (b) Higher values in red represent more wins; lower values in blue indicate more frequent losses in pairs.

**Environment Configuration.** We further investigate how different environmental configurations impact agent behavior and overall performance. Specifically, we explore two key factors: the global order availability and the agents’ initial financial endowment. For each factor, we conduct a series of controlled experiments to observe how variations affect agents’ action distributions.

As shown in Figure 12a, when the total number of available orders increases, agents tend to perform fewer pickup and delivery actions and instead choose the do **nothing** action more frequently. This suggests that in resource-rich environments, agents are more inclined to conserve energy and avoid unnecessary effort, opting to wait for optimal opportunities rather than actively pursue deliveries. Conversely, in low-resource settings, agents are more motivated to engage in delivery tasks to secure profits. Additionally, as resource abundance increases, agents demonstrate a higher tendency to initiate and complete shared deliveries, likely as a means to reduce energy costs through collaboration.

Figure 12b illustrates the impact of agents’ initial monetary resources. As initial capital increases, agents are less reliant on aggressive bidding and instead prioritize actions such as order pickup. When funds are limited, competition intensifies, leading to more frequent bidding behavior. Furthermore, with sufficient initial capital, agents are more willing to invest in infrastructure, such as purchasing a scooter, which enhances their long-term delivery efficiency.

### Takeaway: Resource and Decision-Making Strategy

Order resource scarcity increases agent competitiveness and task urgency. Sufficient agent initial money leads to more relaxed, profit-insensitive behavior (Figure 12).

These observations suggest that agents are more competitive and task-driven in resource-constrained environments. In contrast, resource-rich conditions reduce the urgency to complete tasks and generate immediate profits. Importantly, agents are also more likely to engage in actions that involve upfront costs but promise long-term benefits—such as investment and shared delivery—provided they have the financial capacity and enough orders taken to do so.

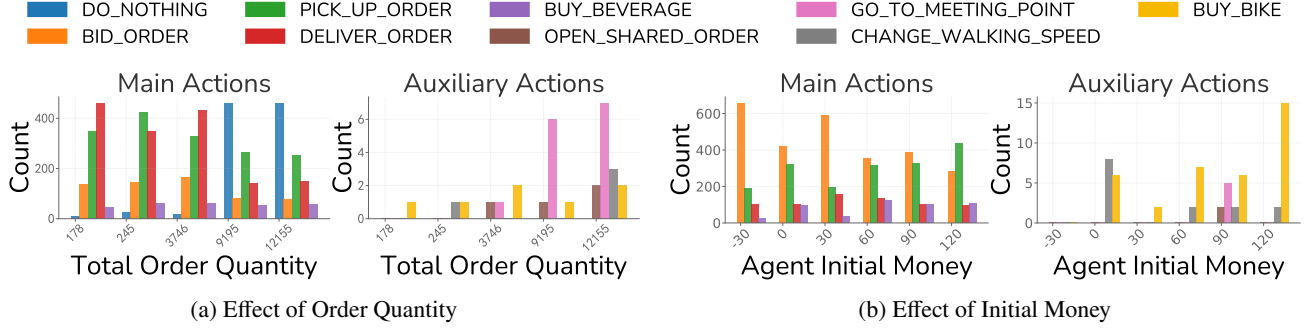


Figure 12: **Action Distribution across Environmental Settings.** (a) shows how global order quantity affects agent behavior; (b) shows the effect of initial money on action selection.

**Influence of Persona.** Personality traits significantly affect the decision-making and performance of delivery agents. As shown in Figure 13, agents with higher Conscientiousness tend to exhibit a lower frequency of bidding actions, a higher frequency of task-completion actions (e.g., picking up orders), and achieve a higher bid win rate. This suggests that conscientious agents prioritize task completion over strategic competition. Agents with higher Agreeableness are less likely to remain inactive (i.e., perform **do nothing** actions) and tend to achieve higher bid win rates. Conversely, agents with lower Agreeableness display higher inactivity and narrower bidding price ranges, limiting their competitiveness. Interestingly, agents with higher Openness exhibit reduced engagement in delivery tasks, possibly because they explore competitive or unconventional bidding strategies that divert attention from task execution.

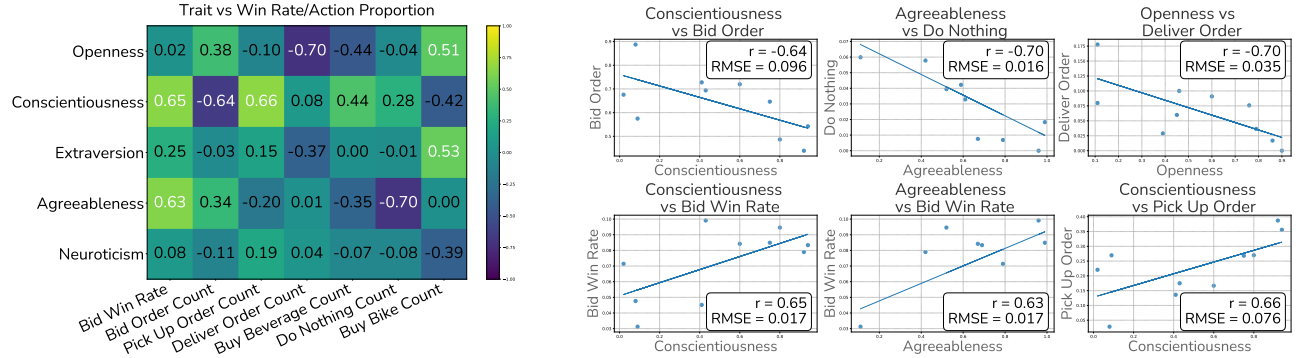


Figure 13: **Persona Influence on Agent Performance and Behavior.** (a) Agents with higher Agreeableness are less likely to remain inactive and tend to achieve higher bid win rates. Conversely, agents with lower agreeableness display higher inactivity and narrower bidding price ranges, limiting their competitiveness. (b) The results demonstrate that agent behaviors are tightly aligned with their corresponding persona attributes, highlighting the effectiveness of trait-driven behavior modeling.

#### Takeaway: Impact of Persona in Multi-agent Interaction

Agent personalities shape strategic tendencies: conscientious agents prioritize task fulfillment, while openness and agreeableness modulate competitiveness and inactivity (Figure 13).

## 4. Related Works

Simulations have played a crucial role in constructing environments for training and evaluating autonomous agents. Text-based simulators often emphasize social scenarios, such as human interaction (Yang et al., 2024),



daily activities (Park et al., 2023), and relational polarization (Piao et al., 2025). Popular embodied simulators support a broader range of applications, particularly in embodied AI research and 2D/3D scene synthesis (Li et al., 2025a). However, most embodied simulators remain constrained to either indoor household environments (e.g., AI2-THOR (Kolve et al., 2017), Habitat (Puig et al., 2023), iGibson (Li et al., 2021)) or outdoor driving scenarios (e.g., CARLA (Dosovitskiy et al., 2017), MetaDrive (Li et al., 2022)) or natural scenes (e.g., AirSim (Shah et al., 2017)). Most of these simulators (Dosovitskiy et al., 2017; Puig et al., 2023; Li et al., 2021; Shah et al., 2017; Wang et al., 2024a; Gao et al., 2024) rely on a limited number of manually crafted scenes, which hinders scalability and diversity. Some platforms, such as MetaUrban (Wu et al., 2025), MetaDrive (Li et al., 2022), AI2-THOR (Kolve et al., 2017) and Genesis (Authors, 2024), introduce rule-based procedural generation to alleviate this issue. Nonetheless, existing embodied simulators typically lack support for dynamic multi-agent interactions in complex diverse environments.

Recent advancements have introduced large-scale, language-driven social simulators capable of modeling complex societal behaviors. OASIS (Yang et al., 2024) simulates up to one million LLM-powered agents interacting on social media platforms, capturing phenomena such as information diffusion, echo chambers, and polarization. Casevo (Jiang et al., 2024) integrates chain-of-thought reasoning, retrieval-augmented generation, and customizable memory mechanisms to simulate intricate social phenomena and decision-making processes. MineLand (Yu et al., 2024) offers a multi-agent Minecraft environment where agents, driven by physiological needs and limited multimodal perception, engage in collective behaviors, fostering ecological and detailed simulations. Project sid (AL et al., 2024) further advances this landscape by deploying a large number of AI agents within a Minecraft environment to explore the emergence of AI civilizations. VirtualCommunity (Zhou et al., 2025) leverages Genesis simulator to conduct community influence task in outdoor, multi-agent scenes. These simulation platforms demonstrate agents’ capabilities to form complex social structures, economies, and governance systems, providing insights into large-scale societal simulations and agentic organizational intelligence.

None of the existing simulators are explicitly designed to support dynamic, multi-agent interactions in large-scale outdoor and other diverse environments with both realistic rendering and physical simulation. SIMWORLD addresses this limitation by providing a scalable, procedurally generated, and LLM/VLM-compatible platform that enables multi-agent collaboration and competition, language-grounded interactions, and comprehensive benchmarking for embodied intelligence.

Another emerging direction in world simulation involves end-to-end neural world models, which generate interactive video predictions conditioned on environment states, agent actions, and high-level controls (Xiang et al., 2024; DeepMind, 2025; Xiang et al., 2025). Recent systems can simulate short video rollouts or 3D-consistent scenes using learned dynamics, offering a flexible alternative to traditional engine-based simulation. On the other hand, SIMWORLD, built on the Unreal Engine, provides high-fidelity, physically grounded, and deterministically controllable environments capable of supporting thousands to even millions of interacting agents at scale. Moreover, because SIMWORLD supports diverse, high-quality procedural and handcrafted scenes, it can serve as a powerful generator of large-scale training data, offering a rich source of supervised trajectories, multi-agent interactions, and physically realistic rollouts that can be used to train and improve neural world models.

## References

- AL, A., Ahn, A., Becker, N., Carroll, S., Christie, N., Cortes, M., Demirci, A., Du, M., Li, F., Luo, S., Wang, P. Y., Willows, M., Yang, F., and Yang, G. R. Project sid: Many-agent simulations toward ai civilization, 2024. URL <https://arxiv.org/abs/2411.00114>.
- Anthropic. Claude’s extended thinking, 2025. URL <https://www.anthropic.com/research/visible-extended-thinking>. Accessed: 2025-05-14.
- Authors, G. Genesis: A generative and universal physics engine for robotics and beyond, December 2024. URL <https://github.com/Genesis-Embodied-AI/Genesis>.
- Brohan, A., Chebotar, Y., Finn, C., Hausman, K., Herzog, A., Ho, D., Ibarz, J., Irpan, A., Jang, E., Julian, R., et al. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on robot learning*, pp. 287–318. PMLR, 2023.

- 
- DeepMind. Genie 3: A new frontier for world models. <https://deepmind.google/blog/genie-3-a-new-frontier-for-world-models/>, 2025. Accessed: 2025-11-27.
- Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pp. 1–16, 2017.
- Driess, D., Xia, F., Sajjadi, M. S., Lynch, C., Chowdhery, A., Wahid, A., Tompson, J., Vuong, Q., Yu, T., Huang, W., et al. Palm-e: An embodied multimodal language model. 2023.
- Fan, L., Wang, G., Jiang, Y., Mandlekar, A., Yang, Y., Zhu, H., Tang, A., Huang, D.-A., Zhu, Y., and Anandkumar, A. Minedojo: Building open-ended embodied agents with internet-scale knowledge. *Advances in Neural Information Processing Systems*, 35:18343–18362, 2022.
- Farama Foundation. Gymnasium. <https://gymnasium.farama.org/>, 2023. Accessed: 2025-11-27.
- Gao, C., Zhao, B., Zhang, W., Mao, J., Zhang, J., Zheng, Z., Man, F., Fang, J., Zhou, Z., Cui, J., et al. Embodiedcity: A benchmark platform for embodied agent in real-world city environment. *arXiv preprint arXiv:2410.09604*, 2024.
- Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.
- Ha, A. Google’s gemini has beaten pokémon blue (with a little help). *TechCrunch*, 2025. URL <https://techcrunch.com/2025/05/03/googles-gemini-has-beaten-pokemon-blue-with-a-little-help/>. Accessed: 2025-05-14.
- Hao, S., Gu, Y., Ma, H., Hong, J. J., Wang, Z., Wang, D. Z., and Hu, Z. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*, 2023.
- Ho, M., Si, C., Feng, Z., Yu, F., Yang, Y., Liu, Z., Hu, Z., and Qin, L. Arcmemo: Abstract reasoning composition with lifelong llm memory. *arXiv preprint arXiv:2509.04439*, 2025.
- Hu, Z. and Shu, T. Language models, agent models, and world models: The law for machine reasoning and planning. *arXiv preprint arXiv:2312.05230*, 2023.
- Hunyuan3D, T. Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3d assets generation, 2025.
- Jain, H. and Babel, P. A comprehensive survey of pid and pure pursuit control algorithms for autonomous vehicle navigation. *arXiv preprint arXiv:2409.09848*, 2024.
- Jiang, Z., Shi, Y., Li, M., Xiao, H., Qin, Y., Wei, Q., Wang, Y., and Zhang, Y. Casevo: A cognitive agents and social evolution simulator. *arXiv preprint arXiv:2412.19498*, 2024.
- Kolve, E., Mottaghi, R., Han, W., VanderBilt, E., Weihs, L., Herrasti, A., Deitke, M., Ehsani, K., Gordon, D., Zhu, Y., et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.
- Li, C., Xia, F., Martín-Martín, R., Lingelbach, M., Srivastava, S., Shen, B., Vainio, K., Gokmen, C., Dharan, G., Jain, T., et al. igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. *arXiv preprint arXiv:2108.03272*, 2021.
- Li, C., Zhang, R., Wong, J., Gokmen, C., Srivastava, S., Martín-Martín, R., Wang, C., Levine, G., Ai, W., Martinez, B., et al. Behavior-1k: A human-centered, embodied ai benchmark with 1,000 everyday activities and realistic simulation. *arXiv preprint arXiv:2403.09227*, 2024.
- Li, Q., Peng, Z., Feng, L., Zhang, Q., Xue, Z., and Zhou, B. Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning. *IEEE transactions on pattern analysis and machine intelligence*, 45(3): 3461–3475, 2022.
- Li, X., Song, R., Xie, Q., Wu, Y., Zeng, N., and Ai, Y. Simworld: A unified benchmark for simulator-conditioned scene generation via world model. *arXiv preprint arXiv:2503.13952*, 2025a.

- 
- Li, Z., Xie, Y., Shao, R., Chen, G., Jiang, D., and Nie, L. Optimus-2: Multimodal minecraft agent with goal-observation-action conditioned policy. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pp. 9039–9049, June 2025b.
- Liu, S. et al. Odyssey: Empowering minecraft agents with open-world skills. *arXiv preprint arXiv:2407.15325*, 2024.
- Long, Q., Li, Z., Gong, R., Wu, Y. N., Terzopoulos, D., and Gao, X. Teamcraft: A benchmark for multi-modal multi-agent systems in minecraft. *arXiv preprint arXiv:2412.05255*, 2024.
- Park, J. S., O’Brien, J., Cai, C. J., Morris, M. R., Liang, P., and Bernstein, M. S. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.
- Phiresky. procedural-cities. <https://github.com/phiresky/procedural-cities>, 2024.
- Piao, J., Yan, Y., Zhang, J., Li, N., Yan, J., Lan, X., Lu, Z., Zheng, Z., Wang, J. Y., Zhou, D., et al. Agentsociety: Large-scale simulation of llm-driven generative agents advances understanding of human behaviors and society. *arXiv preprint arXiv:2502.08691*, 2025.
- Puig, X., Undersander, E., Szot, A., Cote, M. D., Yang, T.-Y., Partsey, R., Desai, R., Clegg, A. W., Hlavac, M., Min, S. Y., et al. Habitat 3.0: A co-habitat for humans, avatars and robots. *arXiv preprint arXiv:2310.13724*, 2023.
- Qiu, W., Zhong, F., Zhang, Y., Qiao, S., Xiao, Z., Kim, T. S., and Wang, Y. Unrealcv: Virtual worlds for computer vision. In *Proceedings of the 25th ACM International Conference on Multimedia, MM ’17*, pp. 1221–1224, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450349062. doi: 10.1145/3123266.3129396. URL <https://doi.org/10.1145/3123266.3129396>.
- Shah, S., Dey, D., Lovett, C., and Kapoor, A. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and service robotics: Results of the 11th international conference*, pp. 621–635. Springer, 2017.
- Wang, G., Xie, Y., Jiang, Y., Mandlekar, A., Xiao, C., Zhu, Y., Fan, L., and Anandkumar, A. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.
- Wang, H., Chen, J., Huang, W., Ben, Q., Wang, T., Mi, B., Huang, T., Zhao, S., Chen, Y., Yang, S., et al. Grutopia: Dream general robots in a city at scale. *arXiv preprint arXiv:2407.10943*, 2024a.
- Wang, Y., Gao, Y., Chen, X., Jiang, H., Li, S., Yang, J., Yin, Q., Li, Z., Li, X., Yin, B., et al. Memoryllm: Towards self-updatable large language models. *arXiv preprint arXiv:2402.04624*, 2024b.
- White, I., Nottingham, K., Maniar, A., Robinson, M., Lillemark, H., Maheshwari, M., Qin, L., and Ammanabrolu, P. Collaborating action by action: A multi-agent llm framework for embodied reasoning. *arXiv preprint arXiv:2504.17950*, 2025.
- Wu, W., He, H., He, J., Wang, Y., Duan, C., Liu, Z., Li, Q., and Zhou, B. Metaurban: An embodied ai simulation platform for urban micromobility. *International Conference on Learning Representation*, 2025.
- Xiang, J., Liu, G., Gu, Y., Gao, Q., Ning, Y., Zha, Y., Feng, Z., Tao, T., Hao, S., Shi, Y., et al. Pandora: Towards general world model with natural language actions and video states. *arXiv preprint arXiv:2406.09455*, 2024.
- Xiang, J., Gu, Y., Liu, Z., Feng, Z., Gao, Q., Hu, Y., Huang, B., Liu, G., Yang, Y., Zhou, K., et al. PAN: A world model for general, interactable, and long-horizon world simulation. *arXiv preprint arXiv:2511.09057*, 2025.
- Xing, E., Deng, M., Hou, J., and Hu, Z. Critiques of world models. *arXiv preprint arXiv:2507.05169*, 2025.
- Yang, Z., Zhang, Z., Zheng, Z., Jiang, Y., Gan, Z., Wang, Z., Ling, Z., Chen, J., Ma, M., Dong, B., et al. Oasis: Open agents social interaction simulations on one million agents. *arXiv preprint arXiv:2411.11581*, 2024.

- 
- Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K. R., and Cao, Y. React: Synergizing reasoning and acting in language models. In The eleventh international conference on learning representations, 2022.
- Yu, X., Fu, J., Deng, R., and Han, W. Mineland: Simulating large-scale multi-agent interactions with limited multimodal senses and physical needs. arXiv preprint arXiv:2403.19267, 2024.
- Zhong, F., Wu, K., Wang, C., Chen, H., Ci, H., Li, Z., and Wang, Y. Unrealzoo: Enriching photo-realistic virtual worlds for embodied ai. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 5769–5779, 2025.
- Zhou, Q., Zhang, H., Lin, X., Zhang, Z., Chen, Y., Liu, W., Zhang, Z., Chen, S., Fang, L., Lyu, Q., et al. Virtual community: An open world for humans, robots, and society. arXiv preprint arXiv:2508.14893, 2025.