

VLNVERSE: A BENCHMARK FOR VISION-LANGUAGE NAVIGATION WITH VERSATILE, EMBODIED, REALISTIC SIMULATION AND EVALUATION

Sihao Lin^{1,2*} Zerui Li^{1,2*} Xunyi Zhao^{1,2*} Gengze Zhou¹
 Liuyi Wang³ Rong Wei⁴ Rui Tang⁴ Juncheng Li⁵ Hanqing Wang⁶
 Jiangmiao Pang⁶ Anton van den Hengel^{1,2} Jiajun Liu^{2,7} Qi Wu^{1,2†}

¹Adelaide University; ²Responsible AI Research Centre, Australian Institute for Machine Learning;
³Tongji University; ⁴ManyCore; ⁵Zhejiang University; ⁶Shanghai AI Lab; ⁷CSIRO Data61
 {sihao.lin, zerui.li, xunyi.zhao, gengze.zhou, qi.wu01}@adelaide.edu.au

<https://sihaoevery.github.io/vlnverse/>

ABSTRACT

Despite remarkable progress in Vision-Language Navigation (VLN), existing benchmarks remain confined to fixed, small-scale datasets with naive physical simulation. These shortcomings limit the insight that the benchmarks provide into sim-to-real generalization, and create a significant research gap. Furthermore, task fragmentation prevents unified/shared progress in the area, while limited data scales fail to meet the demands of modern LLM-based pretraining. To overcome these limitations, we introduce VLNVerse: a new large-scale, extensible benchmark designed for *Versatile*, *Embodied*, *Realistic Simulation*, and *Evaluation*. VLNVerse redefines VLN as a scalable, full-stack embodied AI problem. Its *Versatile* nature unifies previously fragmented tasks into a single framework and provides an extensible toolkit for researchers. Its *Embodied* design moves beyond intangible and teleporting “ghost” agents that support full-kinematics in a *Realistic Simulation* powered by a robust physics engine. We leverage the scale and diversity of VLNVerse to conduct a comprehensive *Evaluation* of existing methods, from classic models to MLLM-based agents. We also propose a novel unified multi-task model capable of addressing all tasks within the benchmark. VLNVerse aims to narrow the gap between simulated navigation and real-world generalization, providing the community with a vital tool to boost research towards scalable, general-purpose embodied locomotion agents.

1 INTRODUCTION

Vision-Language Navigation (VLN) (Anderson et al., 2018) is a fundamental task in Embodied AI where an agent must comprehend natural language instructions to navigate through a 3D space and arrive at a target. VLN is essential for developing versatile Embodied AI, as it seamlessly combines multi-modal data comprehension with action prediction in dynamic settings, making it an excellent platform to evaluate the entire perception-to-action pipeline (Nguyen et al., 2021; Duan et al., 2022; Szot et al., 2021; Hu et al., 2023; Chu et al., 2023). Conventionally, VLN models are often trained and evaluated in simulated environments (Misra et al., 2018; Ku et al., 2020; Jain et al., 2019; Nguyen et al., 2019; Irshad et al., 2021; Liu et al., 2023; Qi et al., 2020; Thomason et al., 2020; Song et al., 2025; Nguyen & Daumé III, 2019; Krantz et al., 2020; Shridhar et al., 2020; Padmakumar et al., 2022; Gao et al., 2022; Zhao et al., 2023). However, the advancement of VLN is fundamentally constrained by the limitations of existing simulation-based platforms. While current simulators have laid the groundwork, they exhibit significant shortcomings that impede progress toward generalizable and deployable agents. For instance, widely-used environments like Matterport3D (MP3D) (Chang et al., 2017) are built on static and discrete graphs, effectively degenerating agents to “ghost-like” entities devoid of physical properties. Subsequent platforms like Habitat (Savva et al., 2019) and Gibson (Xia et al., 2018) have introduced physics, yet they often rely

*Equal Contribution, † Corresponding Author.

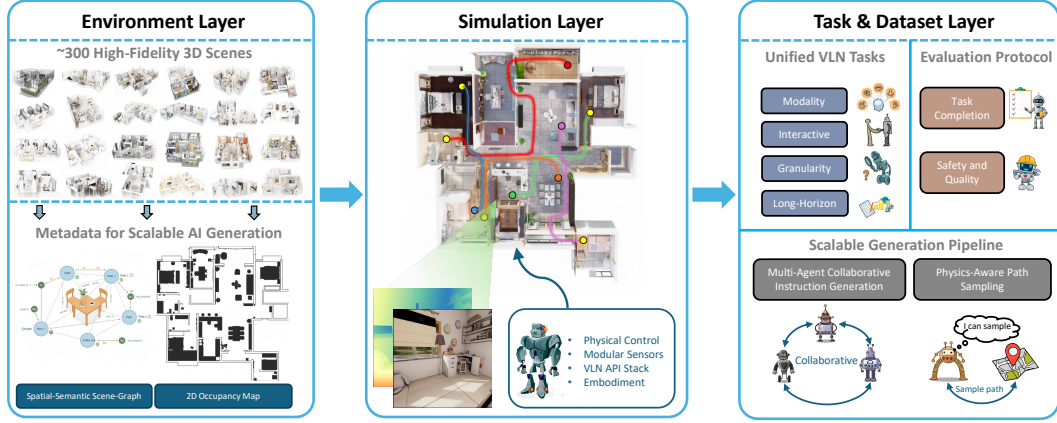


Figure 1: VLNVerse is built on a decoupled three-layer architecture that separates the responsibilities of the Agent (Simulator Layer) Sec. 3.1, World (Environments Layer) Sec. 3.2, and Benchmark (Task & Dataset Layer) Sec. 3.3.

on simplified engines with limited support for complex robot kinematics, continuous control, and flexible user customization. This lack of a high-fidelity simulation represents a critical bottleneck for the field.

This fragmentation at the simulator level directly propagates to task design and algorithmic development. The VLN landscape is populated with disparate benchmarks (Anderson et al., 2018; Ku et al., 2020; Jain et al., 2019; Qi et al., 2020; Thomason et al., 2020; Song et al., 2025; Nguyen & Daumé III, 2019), each optimized for the specific capabilities of its underlying simulator. Naively unifying these disparate tasks on existing platforms is often impractical, as the simulators themselves are not designed for the general physics support required for true versatility. Consequently, existing algorithms are often over-specialized. That is, a model developed for one dataset (*e.g.*, R2R (Anderson et al., 2018)) frequently requires non-trivial modifications to be applied to another, obstructing direct comparison and the knowledge transfer. This paradigm of task-specific solutions fundamentally contradicts the goal of general-purpose embodied agents.

The root of this fragmentation lies in the absence of truly interactive and physics-aware 3D environments. Current benchmarks are predominantly built upon static 3D scans or environments with only hard-coded and simplistic physical interaction (Savva et al., 2019; Xia et al., 2018). They fail to support dynamic interactions between an embodied agent and its surroundings, where an embodied agent must obey the laws of physics and account for its own physical form (kinematics and dynamics) (Ma et al., 2022; Zhou et al., 2024c; Puig et al., 2024; Li et al., 2024b). This deficiency creates a significant sim-to-real gap, as models trained in these abstract environments fail when confronted with the complexities of real-world robotics. Consequently, the field lacks a scalable and unified benchmark designed from the ground up to propel research towards physics-aware navigation and bridge the gap between simulation and deployment (Qiao et al., 2024; Shi et al., 2025a; Li et al., 2025b; Wang et al., 2024c).

These limitations are further compounded by issues of data scale and outdated training paradigms. Existing VLN datasets are often small-scale, inconsistent with the data-hungry trend of modern Multimodal Large Language Models (MLLMs) (Wang et al., 2023b). Furthermore, their static nature binds benchmarks to discrete environments and fixed modalities (*e.g.*, RGBD), making it difficult to extend them to new sensory information or adapt to continuous action spaces. As a result, many advances are confined to offline training regimes, which further hinders physical interaction and obstructs sim-to-real deployment (Hong et al., 2021; Chen et al., 2021; 2022).

To address this fundamental limitation, we propose VLNVerse, a new framework for VLN, specifically designed for **Versatile, Embodied, Realistic Simulation and Evaluation**. VLNVerse is a scalable **full-stack development** framework for VLN, covering embodiment, task fragment, data pipeline, and sim-to-real simulation. Compared to previous benchmark, we redefine VLN from the following aspects:

High-Fidelity Physics-Aware Simulation. We move beyond “ghost-like” agents by building VLNVerse on NVIDIA Isaac Sim (NVIDIA, 2021), leveraging its robust physics engine and photorealistic rendering. Unlike previous simulators that treat navigation as discrete teleportation (Anderson et al., 2018), we provide a comprehensive embodied robotics framework with unified APIs specifically tailored for VLN research. Specifically, we seamlessly integrate Isaac Sim’s physics capabilities into VLN-specific workflows, eliminating the need for researchers to handle low-level robotics details. This foundation enables true embodiment, supporting robot kinematics, continuous control, collision dynamics, and physical object interactions. This design directly targets the sim-to-real gap by forcing agents to obey physical constraints.

Unified Multi-Task Framework. The versatility of our simulator allows us to unify the fragmented landscape of VLN tasks. VLNVerse introduces a unified task taxonomy (Zhou et al., 2024a) that covers a comprehensive spectrum of navigation challenges: (1) *fine-grained navigation*, (2) *coarse-grained (goal-oriented) navigation*, (3) *visual-target search*, (4) *long-horizon, multi-stage tasks*, and (5) *interactive dialogue-based navigation*. This comprehensive taxonomy spans multiple dimensions: granularity, modality, temporal scope, and interaction type. Critically, VLNVerse unifies previously disparate benchmarks under a single framework, enabling the development and evaluation of general-purpose agents. The modular design also ensures effortless extensibility, where researchers can define and integrate new tasks by simply composing existing primitives.

Scalable and Flexible Data Pipeline. To support modern data-hungry models and diverse training paradigms, we provide 263 large-scale, interactive 3D home environment assets. More importantly, we release a scalable data generation toolkit that shifts VLN from static, fixed-size datasets to a dynamic, on-demand generation paradigm. Such a pipeline enable generate training instances at arbitrary scales, customize environmental parameters, and define novel task configurations on demand. This pipeline flexibly supports both offline pre-training and online, interactive fine-tuning.

Novel Unified Multi-Task Model. To promote a shift away from over-specialized models, we propose a unified multi-task navigation model capable of handling all VLNVerse tasks simultaneously. Built on a State-Adaptive Mixture-of-Experts architecture (Zhou et al., 2024a), our model learns task-specific representations while enabling cross-task knowledge transfer. This approach serves as a baseline and represents a step towards general-purpose agent development.

Systematic Evaluation and Baselines. As a new benchmark, we conduct a systematic evaluation of existing methods spanning the evolution of VLN, from classic Seq2Seq (Sutskever et al., 2014) models to cutting-edge MLLM-based agents (*e.g.*, NavGPT (Zhou et al., 2024b), MapGPT (Chen et al., 2024)). We re-implement and test these models in our physics-aware setting, providing crucial insights into their robustness and generalization under embodiment constraints. All implementations, datasets, and evaluation protocols will be made publicly available to ensure reproducibility and facilitate future research.

2 RELATED DATASETS

The evolution of Vision Language Navigation (VLN) benchmarks has been intrinsically tied to, and constrained by, the capabilities of their underlying simulators. Table 1 presents a comprehensive comparison of existing VLN datasets, dissecting their simulator capabilities, environmental diversity, and task taxonomy.

Simulator Capabilities and Limitations. Early, seminal work in VLN was built on platforms like Matterport3D (Chang et al., 2017). While foundational, this platform discretizes environments into navigation graphs. As noted in Tab. 1, this effectively forces agents to “jump” between nodes rather than using low-level physics to navigate a continuous physical space. This approach, built on static 3D scans, also abstracts away physical collision and results in lower-fidelity observations due to rendering artifacts.

Subsequent platforms, most notably Habitat (Savva et al., 2019), introduced physics and continuous control, enabling tasks like VLN-CE (Krantz et al., 2020). However, these environments often employ simplistic physics engines where interactions are limited to basic collision, and the continuous control is not grounded in realistic robot kinematics. Conversely, interaction-centric platforms like AI2-THOR Kolve et al. (2017) (used in ALFRED (Shridhar et al., 2020)) introduce rich physical interaction but are fundamentally designed for object manipulation rather than complex navigation.

Table 1: Comprehensive comparison of VLN Datasets. **Simulator**: ‘Specialized’ implies physics restricted to specific cases; ‘Full’ implies total physical simulation. **New Scenes**: Count of unique environments introduced (0 denotes directly re-use of existing scenes). **Taxonomy**: F (Fine-grained), C (Coarse-grained), I (Interactive), LH (Long-horizon). **Action Space**: ‘Discrete’ (Discrete steps), ‘Graph’ (Node-hopping), ‘Continuous’ (Velocity based control), ‘Hybrid’ (Multi-modal control).

Dataset	Simulator				Taxonomy				Action Space
	Name	Physical	Render Quality	New Scenes #	F	C	I	LH	
LANI/CHAI (2018)	CHALET	Specialized	Low	1	✓	-	-	-	Discrete
R2R (2018)	Matterport3D	-	Low	90	✓	-	-	-	Graph
R4R (2019)	Matterport3D	-	Low	0	✓	-	-	✓	Graph
RxR (2020)	Matterport3D	-	Low	0	✓	-	-	-	Graph
REVERIE (2020)	Matterport3D	-	Low	0	-	✓	-	-	Graph
SOON (2021)	Matterport3D	-	Low	0	-	-	✓	-	Graph
VNLA (2019)	Matterport3D	-	Low	0	-	-	✓	-	Graph
HANNA (2019)	Matterport3D	-	Low	0	-	-	✓	-	Graph
CVDN (2020)	Matterport3D	-	Low	0	-	-	✓	-	Graph
VLN-CE (2020)	Habitat, Matterport3D	Specialized	Medium	0	✓	-	-	-	Discrete
Robo-VLN (2021)	Habitat, Matterport3D	Specialized	Medium	0	✓	-	-	-	Continuous
TouchDown (2019)	Google Street View	-	Low	1	✓	-	-	-	Graph
ALFRED (2020)	AI2-THOR	Specialized	Low	120	✓	-	-	-	Discrete
TEACH (2022)	AI2-THOR	Specialized	Low	0	-	-	✓	-	Discrete
DialFRED (2022)	AI2-THOR	Specialized	Low	0	-	-	✓	-	Discrete
AerialVLN (2023)	AirSim	Specialized	High	25	✓	-	-	-	Discrete
LH-VLN (2025)	Habitat	Specialized	Medium	216	-	-	-	✓	Graph
GSA-R2R (2025)	Habitat, Matterport3D	Specialized	Medium	150	✓	-	-	-	Graph
VLN-PE (2025)	Isaac	Full	High	1	✓	-	-	-	Hybrid
VLNVerse	Isaac	Full	High	263	✓	✓	✓	✓	Hybrid

These environments are often spatially constricted, limiting their utility for training robust navigators that require extensive exploration.

Stagnation in Environmental Diversity. Crucially, our comparative analysis reveals a severe stagnation in environmental diversity. As indicated by the ‘New Scenes #’ column in Tab. 1, the vast majority of subsequent benchmarks (*e.g.*, RxR, CVDN, VLN-CE) contribute zero new environments, instead relying on re-annotations of existing Matterport3D scans. In this context, we explicitly define a “new scene” as an environment with unique geometry and visual appearance, distinct from previously existing datasets. Mere file format conversions do not qualify.

For instance, while VLN-PE (Wang et al., 2025) reports utilizing 101 scenes, it comprises 90 Matterport3D scenes (Chang et al., 2017), 10 from GRScene (Wang et al., 2024a), and 1 custom scan. Although the 90 Matterport3D scenes were converted to Universal Scene Description (USD) (Pixar Animation Studios, 2021) format for Isaac Sim compatibility, they remain topologically identical to the original dataset and thus do not contribute to environmental diversity. This continued reliance on a static set of scenes risks overfitting, where agents memorize specific floor plans rather than learning generalizable navigation policies.

Task Fragmentation. Parallel to simulator limitations and data stagnation, the VLN task landscape itself has become **highly fragmented** (Anderson et al., 2018; Qi et al., 2020; Ku et al., 2020; Thomason et al., 2020; Jain et al., 2019; Zhu et al., 2021; Krantz et al., 2022; Song et al., 2025; Nguyen & Daumé III, 2019; Nguyen et al., 2019; Wang et al., 2025; Zhu et al., 2020; Hong et al., 2025; 2020; Chen et al., 2019). Benchmarks are often over-specialized, isolating single instruction styles: fine-grained navigation (*e.g.*, R2R (Anderson et al., 2018), VLN-CE Krantz et al. (2020), RxR (Ku et al., 2020), FGR2R (Hong et al., 2020)); coarse-grained, goal-oriented navigation (*e.g.*, REVERIE (Qi et al., 2020), SOON (Zhu et al., 2021)); long-horizon tasks requiring multi-stage reasoning (*e.g.*, R4R (Jain et al., 2019; Zhu et al., 2020), IR2R (Krantz et al., 2022), LH-VLN (Song et al., 2025)); and interactive, dialogue-based navigation (*e.g.*, HANNA (Nguyen & Daumé III, 2019), VNLA (Nguyen et al., 2019), CVDN (Thomason et al., 2020)). This separation makes it difficult to develop and evaluate general-purpose locomotion agents.

Advancements with VLNVerse. We note that while VLNVerse is built on NVIDIA’s Isaac Sim, other recent benchmarks leveraging this simulator, such as BEHAVIOR-1K (Li et al., 2024a), Arnold (Gong et al., 2023), and GRUtopia (Wang et al., 2024a), have primarily focused on other embodied tasks like robotic manipulation. In contrast, VLNVerse is the first to harness the high-fidelity

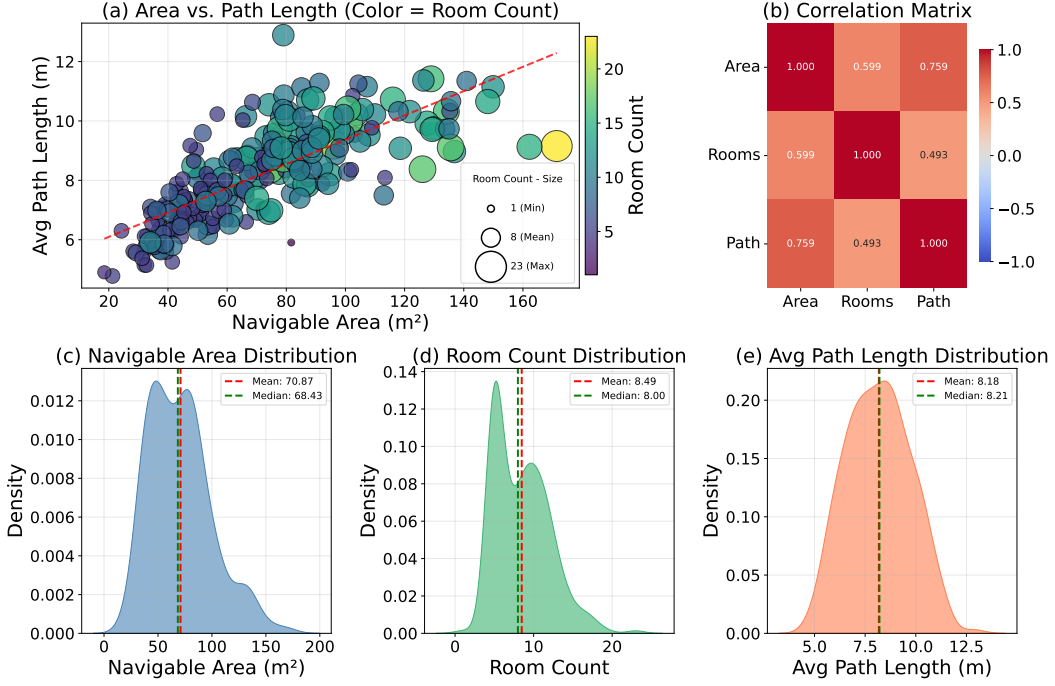


Figure 2: Distribution of navigable area, room and trajectory length.

physics and photorealism of Isaac Sim specifically to address the core challenges of full-stack embodied navigation.

VLNVerse addresses the systemic limitations of previous works by leveraging the Isaac simulator to deliver high-fidelity, fully physical simulation with a ‘Hybrid’ action space. It introduces the largest expansion of diversity with 263 unique scenes and is the first benchmark to unify all taxonomies (Fine-grained, Coarse-grained, Interactive, and Long-horizon), establishing a holistic standard for embodied AI research.

3 THE VLNVVERSE

The VLNVerse is designed as a decoupled three-layer architecture, as illustrated in Fig. 1. The simulator layer serves as the foundation of the framework, encapsulating embodiment, control logic, and perception APIs. The environment layer provides high-fidelity and interactive 3D environment assets, as well as its meta-information. The task & dataset layer builds upon the first two layers, encompassing navigation tasks, data generation pipeline, and evaluation protocol.

3.1 SIMULATION LAYER

Our framework is designed as a full-stack solution, with the Simulator Layer serving as its foundation. This layer is built upon NVIDIA Isaac Sim to leverage its high-fidelity rendering and robust physics engine. However, our primary contribution is not the engine itself, but a novel, high-level abstraction API stack designed specifically for VLN research. This API provides: (1) **seamless integration** of Isaac Sim’s physics capabilities into VLN-specific workflows; (2) **standardized interfaces** that abstract complex robot control while preserving physical realism; and (3) **universal compatibility** with various robot morphologies and sensors.

This design eliminates the need for researchers to handle low-level robotics details, allowing them to focus on high-level navigation and interaction logic. Our Simulator Layer’s design manifests these principles in three key areas: embodiment, control, and perception.

Parameterizable Agent Embodiment. Instead of relying on abstract floating cameras, VLNVerse defines agents as concrete, physics-aware entities. We provide a highly parameterizable agent definition, where researchers can specify the agent’s physical footprint (*e.g.*, as a cylinder with configurable height and diameter) and kinematic properties. This ensures that all navigation is subject to realistic physical constraints, such as collision and momentum, which is critical for sim-to-real transfer.

Physics-Aware Control and Locomotion. Our framework provides a unified controller interface that governs both agent locomotion and camera articulation. This interface executes actions via a physics-aware controller, simulating realistic movement rather than discrete, grid-world teleports like MP3D (Chang et al., 2017). This abstracts the complexity of continuous control, allowing researchers to develop policies without needing to implement the underlying physics simulation.

Configurable and Modular Perception Rig. To support the diverse sensory needs of modern VLN models, agents are equipped with a modular sensor rig. This API allows researchers to easily attach, detach, and configure multiple sensors (*e.g.*, RGB, Depth, LiDAR). Crucially, all sensor parameters are exposed to the user, including sampling frequency, image resolution, and Field of View (FoV). This flexibility enables researchers to simulate different hardware trade-offs and customize the observation data pipeline to their specific research questions.

3.2 ENVIRONMENT LAYER

High-Fidelity Physics-Aware Environment Assets. As the foundation of our data pipeline, this work introduces a set of 263 large-scale, diverse, and interactive 3D environments. Unlike static 3D scans, these environments are all hand-crafted as Universal Scene Description (USD) (Pixar Animation Studios, 2021) assets, where every object is fully interactive and physics-aware. For example, mirrors in our scenes realistically reflect light and other objects, and collisions are not simple binary events. In other words, an agent colliding with an obstacle will be physically deflected based on its mass and velocity. We leverage the capabilities of NVIDIA Isaac Sim to initialize these scenes, assign physical properties (*e.g.*, mass, friction, reflectivity) to all interactable objects, and ensure a physically-grounded simulation environment.

Meta Information and Scene Prior. For each 3D environment, the objects are initialized with their physics and collision properties. Based on this, we generate a 2D occupancy map, which is sufficient for a general-purpose agent and serves as the navigation ground. Concurrently, we build a detailed spatial-semantic scene-graph. Since all objects (*e.g.*, cup, chair) are trackable assets with precise coordinates and semantic labels, this scene-graph captures the spatial state and semantic relationships of all entities (*e.g.*, cup \rightarrow on \rightarrow table). This graph serves as a critical ground-truth prior for subsequent path and instruction generation.

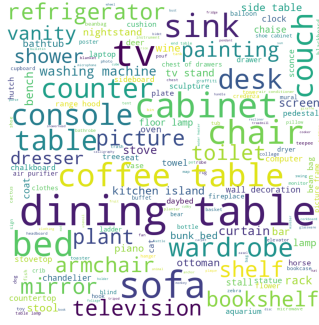
3.3 TASK & DATASET LAYER

3.3.1 TASK DEFINITION

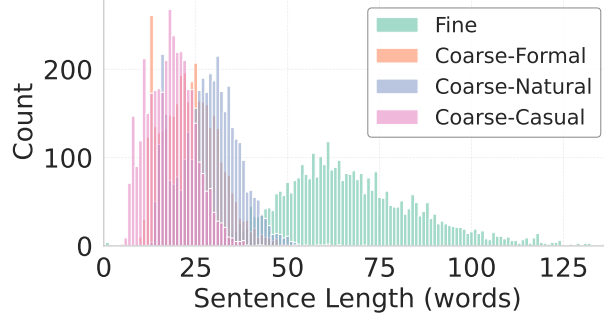
Our pipeline generates data for a unified task framework designed to cover the key axes of VLN research (Anderson et al., 2018; Qi et al., 2020; Zhu et al., 2021; Thomason et al., 2020; Song et al., 2025; Ku et al., 2020):

- **Instruction Granularity:** We provide (1) Fine-grained Navigation (*i.e.*, R2R-style) (Anderson et al., 2018), requiring step-by-step instruction following, and (2) Coarse-grained Navigation (goal-oriented) (Qi et al., 2020; Zhu et al., 2021), where the agent must reach a target described by its properties, testing high-level understanding.
- **Input Modality:** We include (3) Visual-Reference Navigation, which extends the input modality beyond language by providing a visual cue (*e.g.*, a photo of an object (Zhu et al., 2017)) that the agent must locate.
- **Planning Horizon:** We introduce (4) Long-Horizon Navigation, which links multiple instructions into a complex multi-stage task. This challenges long-term planning, serving as a proxy to lifelong learning scenarios.

- **Interactivity:** We support (5) Dialogue-based Navigation, enabling two-way interactive navigation where the agent can actively seek clarification from an Oracle to resolve ambiguity.



(a) Landmarks.



(b) Instruction length distribution.

Figure 3: Statistics of instruction landmarks and length.

3.3.2 SCALABLE GENERATION PIPELINE

To create diverse and large-scale training instances, we designed a two-stage generation pipeline:

Physics-Aware Path Sampling. First, we sample navigation paths. By sampling start and goal points on the 2D occupancy map, we use an A* algorithm to find a collision-free trajectory. Critically, to ensure navigational safety and account for true embodiment, we dilate the occupancy map based on the agent’s physical dimensions (*i.e.*, its height and diameter). This ensures the sampled path is navigable by the physical agent and prevents collisions.

Ground-Truth Instruction Generation. Then, we generate the ground-truth language instruction matched with the sampled path. This is a robust three-stage process:

- **Prior-based Initialization:** We use the spatial-semantic prior (scene graph) to initialize a factually-grounded description. This grounds the instruction in semantic truth (*e.g.*, staring at the bedroom) or spatial relationship (*e.g.*, the mirror on the sink).
- **Collaborative AI Generation:** We employ a multi-agent AI system for refinement. **Describer** (Agent 1) analyzes visual primitives (*e.g.*, video clips) from the path to describe the environment. **Verifier** (Agent 2) cross-checks the Describer’s output against the scene-graph prior to identify discrepancies or hallucinations. **Synthesizer** (Agent 3) receives the verified information, task definition (*e.g.*, fine-grained *vs.* coarse-grained), and linguistic style (*e.g.*, formal *vs.* casual) to compose the high-quality instruction. More details are in Appendix C.
- **Human Verification:** As a final quality-control step, all generated instructions are presented to human volunteers. These annotators score the instructions for clarity, naturalness, and accuracy, ensuring the quality of VLNVerse.

3.4 EVALUATION PROTOCOL AND METRICS

VLNVerse introduces a comprehensive suite of metrics that augments standard benchmarks with measures of physical robustness.

Traditional VLN Metrics. We evaluate agents using established metrics for task completion and efficiency. These include: Success Rate (SR), Oracle Success Rate (OSR), Success weighted by Path Length (SPL), Navigation Error (NE), Trajectory Length (TL), and normalized Dynamic Time Warping (nDTW) (Ilharco et al., 2019). Detailed definitions for each of these metrics can be found in the Appendix B.

Long-Horizon Task Metrics. For long-horizon tasks with sequential goals, we introduce SR_n to indicate the success rate for reaching the n -th goal (*e.g.*, SR_1 for the first goal). Subsequent metrics

Table 2: Data volume of VLNVers. The number of episodes (instruction-trajectory pairs) are listed, with the numbers in brackets indicating the scene amounts.

Taskonomy	Train	Seen val	Unseen val	Test
Fine-grained	3963(177)	423(157)	825(33)	1325(53)
Coarse-grained	11895(177)	1269(162)	2505(33)	3975(53)
Visual reference	11895(177)	1269(162)	2505(33)	3975(53)
Long horizon	11946(177)	1329(177)	2475(33)	3975(53)
Dialogue	11895(177)	1269(162)	2505(33)	3975(53)

like SR_2 are conditional, measuring success in reaching the second goal given that the agent successfully reached the first. This pattern continues for all intermediate goals. Finally, SR_{All} measures the agent’s success in stopping within the threshold distance of the *final* goal, representing overall task completion.

Physics-Aware Interaction Metrics. To capture the crucial aspects of embodied navigation, we introduce a new metric *Collision Rate* (CR). This metric quantifies the frequency of collision events the agent experiences during an episode. It directly reflects the agent’s low-level obstacle avoidance capability. A low CR indicates the agent is navigating safely and smoothly.

3.5 DATA VOLUME & STATISTICS

Dataset Scale. The scale of the VLNVers dataset is detailed in Tab. 2. Our benchmark is built upon a total of 263 unique 3D scenes, which are exclusively split into training (177 scenes), unseen validation (33 scenes), and test (53 scenes) sets. We generate the dataset based on Sec. 3.3.1.

Specifically, for coarse-grained navigation, we generate three distinct styles (formal, natural, and casual) of instruction for each trajectory to enhance linguistic diversity. The visual reference and dialogue tasks use the identical trajectories and instructions as the coarse-grained dataset. As an auxiliary signal, we provide a reference image sampled at the final goal location for visual reference task. For dialogue navigation, we implement an LLM-based oracle that the agent can query during navigation to gain additional information about the environment. Finally, the long-horizon tasks are constructed by randomly sampling and sequencing 2-3 sub-tasks to challenge the agent’s long-term planning and memory. More details are in Appendix C.

Instruction Statistics. We provide a statistical analysis of the instructions in VLNVers. As shown in Fig. 4 and Fig. 3a, the vocabularies for both navigation actions and environmental landmarks are rich, reflecting the diversity of our tasks and scenes. The Fig. 3b details the instruction lengths, where we observe clear patterns: Fine-grained (Fine) instructions are significantly longer, while the three styles of coarse-grained instructions (Coarse-Formal, Coarse-Natural, Coarse-Casual) are more concise and vary in length. This linguistic variety ensures our benchmark provides comprehensive coverage.

Navigable Environment Statistics. We also analyze the physical statistics of our 3D environments, as shown in Fig. 2. The distributions for navigable area, room count, and path length are all highly varied, covering a wide range of scenarios rather than a simple uniform spread. The scatter plot and correlation matrix (Fig. 2 a&b) confirm the intuitive positive relationships between these factors: larger areas and scenes with more rooms tend to have longer navigation paths. This broad and varied distribution of scene sizes and complexities ensures that VLNVers presents a significant and diverse challenge for navigation agents.

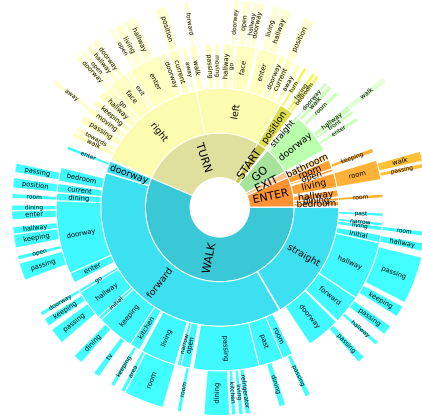


Figure 4: Hierarchy of fine-grained instruction. The white areas denote words with negligible contributions.

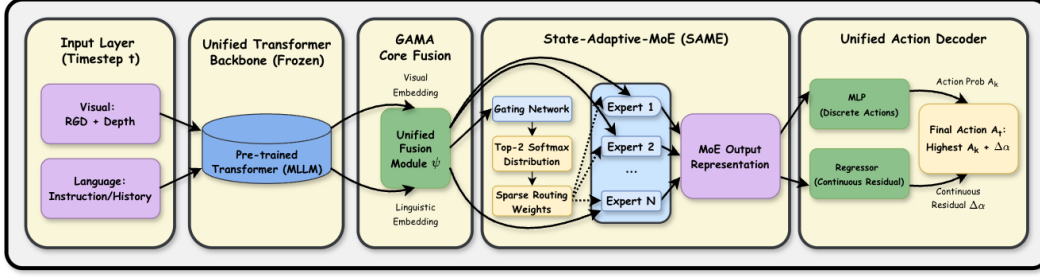


Figure 5: Framework of the proposed General-purpose Agent for Multi-task Navigation (GAMA).

4 GAMA: GENERAL-PURPOSE AGENT FOR MULTI-TASK NAVIGATION

To address the task fragmentation mentioned in Sec. 1, we propose the General-purpose Agent for Multi-task Navigation, dubbed as GAMA. GAMA is a general-purpose framework to handle diverse observation (*i.e.*, input views, FoV) and action spaces (discrete *vs.* continuous), as shown in Fig. 5.

4.1 ARCHITECTURAL OVERVIEW

Unified Transformer Backbone. GAMA’s backbone leverages the pre-trained vision-language models (Tan & Bansal, 2019; Wang et al., 2023b; Li et al., 2024c) to harness their cross-modal aligned embedding and spatio-temporal understanding for sequential tasks, allowing the robust generalization across diverse navigation scenarios.

A unified (and trainable) fusion module, Ψ , then serves as the GAMA-specific core. It integrates representations from two primary data streams at timestep t :

- Visual Stream (V_t): RGB frames and Depth maps are processed by the frozen transformer encoder. These are concatenated and interpolated if necessary, yielding the visual embeddings.
- Language Stream (L_t): The instruction L (*e.g.*, single instruction or dialogue history) is tokenized and encoded by the same frozen MLLM to produce linguistic embeddings.

Subsequently, Ψ fuses the visual and linguistic embeddings to generate a holistic state representation for the subsequent decoder.

Unified Action Decoder. Instead of separate action heads, we treat discrete actions as Dirac δ priors in a continuous space: $p(a|\psi(V_t, L_t)) = \sum_k \alpha_k \cdot \delta(a - a_k)$ where a_k are 4 action primitives including stop and α_k are the corresponding probabilities, computed by an MLP. For continuous control, the model regresses a residual Δa added to the most likely primitive a_k .

4.2 STATE-ADAPTIVE MOE (SAME)

GAMA integrates the *State-Adaptive Mixture-of-Experts* (SAME) (Zhou et al., 2024a), a routing mechanism designed for sequential decision-making in embodied navigation. Navigation tasks typically involve multi-step rollouts in which the relevance of linguistic cues and visual observations varies across timesteps. Conventional token-wise or task-wise MoE formulations do not align well with this temporal structure and often lead to unstable behavior or task interference (Zhou et al., 2024a). SAME instead performs routing along the *time dimension*, enabling expert selection to follow the agent’s state progression.

At each new timestep in which the agent enters a different location, SAME updates the selected experts based on the fused multimodal representation extracted by the policy backbone. The gating network outputs two independent top-2 softmax distributions, producing a sparse combination of experts conditioned on the current observation and instruction. By operating at the state level rather than the token level, SAME aligns the routing mechanism with the sequential nature of navigation while keeping the computation lightweight.

Table 3: Evaluating methods including discrete low-level action (CMA, Seq2Seq), continuous action (RDP), neural implicit representation (HNR) and our unified model.

Methods	Fine-grained Task								Coarse-grained Task							
	Val Seen		Val Unseen		Test				Val Seen		Val Unseen		Test			
	SR↑	SPL↑	SR↑	SPL↑	TL	NE↓	SR↑	SPL↑	SR↑	SPL↑	SR↑	SPL↑	TL	NE↓	SR↑	SPL↑
Human	-	-	-	-	12.13	1.74	86.00	68.60	-	-	-	-	10.23	1.97	77.20	62.50
CMA (2019)	37.35	33.36	31.15	27.92	8.53	5.24	28.83	25.54	32.15	29.06	35.52	32.48	7.53	4.72	31.85	28.67
RDP (2025)	47.28	41.69	48.60	42.72	8.18	4.60	36.38	32.29	41.61	37.53	40.85	35.13	8.35	4.65	36.60	32.21
Seq2Seq (2014)	32.62	30.39	35.03	33.37	5.63	4.79	30.19	28.15	31.91	29.68	33.09	31.00	6.32	5.00	27.17	25.53
HNR (2024b)	36.34	32.10	32.95	29.56	7.98	4.93	32.24	29.03	36.02	33.67	40.92	37.27	7.95	4.62	34.59	30.45
Ours	38.25	34.84	38.69	34.38	5.53	4.52	37.72	33.85	42.45	38.89	48.30	41.50	3.58	4.23	37.20	31.02

This formulation offers two primary benefits:

1. **Compositional Multi-task Behavior.** Experts learn complementary navigation behaviors, and the agent dynamically switches among them within an episode. This reduces cross-task interference and enables effective composition of diverse navigation skills.
2. **Computational Efficiency.** Routing is computed once per timestep on a shared multi-modal feature, avoiding token-wise routing overhead common in sparse MoE layers. This substantially lowers computational cost during long-horizon rollouts.

We refer to Appendix A for more implementation details and architecture design about GAMA.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Baselines. To provide a rigorous benchmark, we conduct a comprehensive evaluation of existing methods, which we group into two primary categories:

- **Classic & Specialized Models:** This group includes specialized VLN models: discrete low-level action (Seq2seq (Sutskever et al., 2014) and CMA (Wang et al., 2019)), continuous action (RDP (Wang et al., 2025)), and neural implicit representation method HNR (Wang et al., 2024b).
- **Foundation Model:** We also evaluate InternNav-N1 (InternRobotics, 2025), NFM (Zhang et al., 2025), and UniNavid (Zhang et al., 2024b) in zero-shot setting.
- **Large Model Agents:** This group tests the generalization of modern large models. We implement two variants with text-summary history (Zhou et al., 2024b; 2025) and text-map history (Chen et al., 2024) in QwenVL3-4B (Yang et al., 2025).

Implementation Details. For a fair comparison, all baseline models are trained strictly in an offline training regime using their default, out-of-the-box experimental settings.

Training Paradigm. We test two kinds of training paradigms: 1) Offline Training (Sec. 5.2): The navigation model is trained to mimic expert actions using the static data; 2) Online Fine-tuning (Sec. 5.5): To adapt to the dynamic simulator, we fine-tune the model using online training. The agent is allowed to explore and interact with the environment, forcing it to learn a robust physics-aware policy.

5.2 BASELINE PERFORMANCE ON VLNVVERSE

We benchmark several methods on VLNVerse using offline training. We pre-render all observations on the trajectory. All experiments are implemented using the PyTorch framework and conducted on NVIDIA RTX 4090 GPUs. The CMA and Seq2Seq models are trained on a single GPU with a batch size of 2, requiring approximately one day to reach convergence. Conversely, the RDP, HNR, and

Table 4: Performance Comparison of Navigation Foundation Model on test split.

Model	TL	NE↓	SR↑	OSR↑	SPL↑	CR↓
<i>Fine-grained Task</i>						
InternNav-N1 (2025)	9.23	5.68	28.95	38.69	25.00	-
NFM (2025)	8.58	4.13	51.59	69.68	32.4	19.35
UniNavid (2024b)	12.35	4.11	45.74	67.02	26.91	12.04
<i>Coarse-grained Task</i>						
InternNav-N1 (2025)	4.00	5.71	17.51	23.32	16.54	-
NFM (2025)	6.52	4.93	38.02	45.93	23.15	24.22
UniNavid (2024b)	10.33	5.05	29.68	42.70	13.47	15.59

Table 5: Performance comparison of VLN agents with Tel-Hop and strict collision detection mechanisms.

Agent	Collision	Fine-grained Task							Coarse-grained Task						
		TL	NE↓	SR↑	OSR↑	SPL↑	nDTW↑	CR↓	TL	NE↓	SR↑	OSR↑	SPL↑	nDTW↑	CR↓
Text-history (2024b)	Tel-Hop	24.22	6.10	19.27	61.46	8.13	52.61	32.13	22.36	5.24	31.25	55.73	7.25	49.60	28.99
	Strict	6.92	6.10	19.27	21.88	14.93	42.66	56.93	6.57	5.75	23.96	28.65	15.34	43.97	52.31
Map-history (2024)	Tel-Hop	34.23	5.62	25.53	55.32	7.57	43.38	27.56	52.68	4.33	49.48	69.79	12.32	46.16	36.16
	Strict	5.43	6.16	16.67	23.44	13.23	42.85	48.20	7.05	5.52	22.92	31.25	14.23	45.55	54.71
Text-history (2024b)+CoT	Tel-Hop	21.22	5.25	28.72	50.53	9.91	44.12	20.70	36.48	4.22	43.75	70.83	9.60	47.01	26.62
	Strict	5.07	6.23	17.19	18.75	14.78	41.96	43.23	6.33	5.53	21.35	29.17	14.14	45.25	38.58
Map-history (2024)+CoT	Tel-Hop	40.92	4.51	42.19	72.40	11.32	45.22	25.30	24.18	4.32	43.23	60.94	15.03	49.68	18.57
	Strict	5.43	5.88	16.67	22.40	14.02	43.13	43.65	5.51	5.36	26.04	31.77	18.61	45.52	33.42

GAMA models are trained across 4 GPUs utilizing `DataParallel`, with a total effective batch size of 4, requiring approximately two days. We optimize all models using AdamW with a learning rate of 1×10^{-4} . The maximum trajectory length is capped at 200 steps.

The results summarized in Tab. 3 highlight the significant challenge posed by the VLNVerse benchmark. On the fine-grained test split, our unified model demonstrates the strongest performance among the evaluated methods, achieving the highest SR at 37.72% and the highest SPL at 33.85%. This is closely followed by the diffusion-based model, RDP (InternRobotics, 2025), which also shows robust results with an SR of 36.38% and an SPL of 32.29%. In contrast, other methods show lower performance. The neural implicit representation model, HNR (Wang et al., 2024b), achieves an SR of 32.24%, while classic discrete-action methods like Seq2Seq (Sutskever et al., 2014) (30.19% SR) and CMA (Wang et al., 2019) (28.83% SR) lag further behind.

We also evaluate large-scale, pre-trained navigation foundation models in Tab. 4. It is noteworthy that while these models have demonstrated strong capabilities on established discrete-action benchmarks (Anderson et al., 2018; Savva et al., 2019), their performance tends to fluctuate when migrated to our more complex, physics-aware continuous environment. Among these methods, NFM (Zhang et al., 2025) achieves the highest success rates at 51.59% and 38.02% for fine-grained and coarse-grained tasks, respectively. UniNavid (Zhang et al., 2024b) posts comparable success rates of 45.74% and 29.68%, while InternNav-N1 (InternRobotics, 2025) shows the lowest performance in this group with success rates of 28.95% and 17.51%.

The overall performance gap suggests our environment is challenging. Traditional discrete-action methods (CMA, Seq2Seq) appear to struggle. Models better suited for continuous control, such as the diffusion-based RDP and our unified model, achieve higher success. Furthermore, the significant drop in SPL for foundation models like InternNav-N1 and UniNavid indicates they struggle with path efficiency in our setting, even when they successfully find the target.

5.3 ZERO-SHOT PERFORMANCE ON VLNVVERSE

We evaluate both text-history and map-history agents (Zhou et al., 2024b; Long et al., 2023; Qiao et al., 2024; Shi et al., 2025a;b; Li et al., 2025a), utilizing QwenVL3-4B (Yang et al., 2025) as the LLM backbone. Following previous works (Qiao et al., 2024; Shi et al., 2025a; Qiao et al., 2025), we randomly sampled 200 test trajectories across all test scenes for zero-shot evaluation.

Table 6: Performance comparison of VLN agents across visual reference, dialogue, and long-horizon navigation tasks.

Agent	TL	NE↓	SR↑	OSR↑	SPL↑	nDTW↑	CR↓	SR1↑	SR2↑	SR3↑
<i>Visual Reference</i>										
Text-history (2024b)	24.43	5.70	29.26	55.32	7.79	47.76	32.26	-	-	-
Map-history (2024)	54.51	4.85	42.55	77.13	5.19	46.29	36.20	-	-	-
<i>Dialogue-based</i>										
Text-history (2024b)	24.57	2.92	84.57	88.30	35.20	60.46	30.92	-	-	-
Map-history (2024)	25.36	4.14	67.02	74.47	30.46	53.62	30.33	-	-	-
<i>Long-horizon</i>										
Text-history (2024b)	39.77	5.68	20.74	-	6.45	33.99	19.56	34.04	12.23	4.79
Map-history (2024)	49.92	5.82	25.5	-	2.13	47.76	33.49	77.13	46.28	10.64

We established two distinct evaluation settings: 1) **Strict**: Navigation occurs under full physical constraints. An episode fails upon a noticeable collision, defined as an obstacle displacing the agent by 0.1m. 2) **Tel-Hop**: The agent teleports between waypoints, bypassing obstacles during transit. Physical constraints are only applied at the destination. If the target is inside an obstacle, the physics engine resolves it by moving the agent to the nearest reachable location, and the episode proceeds. This setting mimics discrete-environment evaluations but still penalizes physically invalid final waypoints.

Separately from these settings, we also investigate the impact of Chain-of-Thought (CoT) (Wei et al., 2022) prompting. This strategy, designed to enhance the LLM’s reasoning process, is applied as an augmentation to agents in both the Strict and Tel-Hop settings.

Our findings reveal a critical gap between idealized and physics-based navigation. As seen in Tab. 5, agent performance is severely degraded in the Strict physics-based setting. For instance, the map-history agent’s success rate drops from 25.5% in the Tel-Hop setting to 16.7% in the Strict setting. This performance gap stems from the fact that these MLLM agents are designed for discrete, collision-free graphs, where waypoint predictions do not account for physical embodiment. The Collision Rate statistics strongly support this: in the Strict setting, agents frequently collide, leading to immediate episode failure.

Strikingly, when we introduce CoT prompting, performance in the Tel-Hop setting improves further, yet performance in the Strict setting remains low. This result strongly confirms that physical collision is the primary bottleneck. While enhanced reasoning improves performance in an idealized, collision-relaxed setting, it is insufficient to overcome the physical-interaction challenge.

These results suggest that while the MLLM’s in-context reasoning possesses sufficient zero-shot capability for high-level navigation logic, current agent models are extremely sensitive to collision-aware environments.

5.4 COMMUNICATIVE AND LONG-HORIZON NAVIGATION

We further evaluated the zero-shot performance of the map-history agent on more complex tasks, building upon the coarse-grained navigation setup. All experiments in this section were conducted in the **Tel-Hop** setting to isolate the agent’s high-level reasoning capabilities from the physical collision bottleneck identified in Sec. 5.3. We tested three distinct scenarios.

Visual Reference Navigation. Here, the agent received a reference image of the target area in addition to the textual instruction. As seen in Tab. 6, this supplementary visual cue did not yield a meaningful performance improvement. The SR remained stagnant at 42.6% (compared to 42.2% for the map-history agent with CoT in Tab. 5). Interestingly, the average Trajectory Length (TL) increased from 40.9 to 54.5. This suggests the MLLM may have struggled to properly ground the visual cue, perhaps searching for a perfect visual match, which prevented it from correctly identifying the stopping condition.

Table 7: Performance of Online fine-tuning on VLN-BERT agent.

VLN-BERT (2021)	TL	NE↓	SR↑	OSR↑	SPL↑	nDTW↑	CR↓
Zero-shot	26.2	6.76	3.4	42.3	1.5	31.3	52.9
Fine-tuned(1 epoch)	23.7	6.13	11.1	53.3	5.7	33.2	31.5
Fine-tuned(10 epochs)	4.28	5.23	25.7	29.7	23.7	49.7	19.5

Dialogue-Based Navigation. In this setting, the agent could query an oracle LLM for environmental information and navigational advice. This intervention, detailed in Tab. 6, led to a massive performance boost, with the map-history agent’s SR jumping from 42.2% to 67.0%.

This dialogue-based result is highly significant and leads to two key conclusions. First, the oracle LLM, which had no prior fine-tuning on our environment, could process the context and provide effective guidance. This demonstrates that the high-quality, realistic rendering and logical structure of VLNVerse are highly compatible with the reasoning capabilities of large models. Second, the ability to engage in dialogue effectively reduces agent confusion and resolves ambiguity, drastically improving the navigation success rate. This strongly supports the value of interactive agents for complex, real-world tasks.

Long-Horizon Navigation. In this task, the agent was given an instruction to navigate to 2~3 goals sequentially. As shown in Tab. 6, the agent performed well at reaching the first goal, achieving an SR1 of 77.1%. However, performance dropped significantly for subsequent goals, with SR2 at 46.3% and SR3 falling to just 10.6%. While the final goal success rate (SR-All) was 25.5%, the sharp decline in sequential success indicates that the model struggles to maintain a complex, multi-stage plan over extended horizons.

5.5 ONLINE FINE-TUNING

Loss Function. Methods based on waypoint prediction (Hong et al., 2022; An et al., 2023; 2022; Wang et al., 2023a; Li et al., 2025b; Wang et al., 2024c) are particularly challenging for purely offline training, as the absence of an interactive feedback mechanism in the new environment prevents the correction of accumulated errors from waypoint prediction, leading to rapid navigation failure. A key feature of VLNVerse is its support for online fine-tuning. To demonstrate this, we investigate the effectiveness in adapting existing models pre-trained on other benchmarks. Specifically, we employ VLN-BERT (Hong et al., 2022) and evaluated its zero-shot performance in VLNVerse. The agent in our experiments is trained by waypoint-based methods (Hong et al., 2022), using Imitation Learning (IL) with a cross-entropy loss on the action probabilities p_t and the oracle action a_t^* . The loss is defined as:

$$\mathcal{L}_{IL} = - \sum_t a_t^* \log(p_t), \quad (1)$$

where the oracle action a_t^* corresponds to the waypoint (★ in Fig. 6) with the shortest geodesic distance to the target among all predicted candidates(● in Fig. 6). The oracle stop is successful when the geodesic distance between the agent and the target(★ in Fig. 6) is less than 1.5 meters and stays in open-space. Note that in rare cases, the waypoint predictor might not be able to return an oracle waypoint that brings the agent closer to the target. In order to allow the agent to explore the environment while learning from teacher actions, we control the agent with schedule sampling (Bengio et al., 2015) to probabilistically sample an action between the oracle and the prediction at each step.

Experimental Setup. The model used in our experiments (VLN-BERT (Hong et al., 2022)) is initialized from the pre-trained PREVALENT model (Hao et al., 2020). Additionally, the weights of our navigation policy are initialized from a model pre-trained with VLN-CE in the Habitat simulator. We select the first 30 scenes from the VLNVerse for online training. This subset selection is motivated by the computational demands inherent to online training, which involves real-time rendering and dynamic scene interactions. To ensure consistency, the validation and testing are also performed using the validation and test splits corresponding to these same 30 scenes. We train the VLN-BERT for 10 epochs with a batch size of 4, taking approximately 15 hours to complete using 4 NVIDIA RTX 3090 GPUs.



Figure 6: Visualization of trajectories and predicted waypoints.

Results. As shown in Tab. 7, due to domain shift, VLN-BERT exhibits poor performance with SR of 3.4%. The performance of VLN-BERT increases to 11.09% SR after only one epoch of brief training conducted on a subset consisting of the first 30 scenes in VLNVerse. We then train it for 10 epochs and achieves 25.7% SR and 23.7% SPL. This result strongly validates the design of our benchmark. It shows that the offline and online training paradigms provided by VLNVerse are complementary and effective.

Agent-Environment Interaction Visualization. Fig. 6 visualizes the agent’s trajectories and the predicted waypoints at each step. As shown by the waypoints (●) in panoramas and the occupancy maps, most of the predictions are nicely positioned at accessible spaces, pointing towards explorable directions around the agent. Thanks to the predicted waypoints, the agent often only needs to make a few decisions to complete a long navigation task. However, in some cases, such as Steps 4 and 5, the waypoint predictor might not be able to return a waypoint that immediately reduces the distance to the target due to environmental constraints (*e.g.*, the target is separated from the agent by a wall). Consequently, the agent must navigate a U-shaped path to bypass the obstacle rather than moving directly towards the target. In order to allow the agent to successfully explore such complex topologies while learning from teacher actions, we control the agent using schedule sampling (Bengio et al., 2015), it probabilistically samples an action between the oracle and the prediction at each step, enabling the agent to break out of local optima and effectively navigate around obstacles. A specific example is shown in Step 6 in Fig. 6, where the agent executes a predicted candidate waypoint(●) rather than the oracle action(★) determined by the geodesic distance.

6 CONCLUSION

This work introduces VLNVerse, a comprehensive, scalable, and physically-grounded benchmark for Vision-Language Navigation. Our framework is built on a decoupled three-layer architecture that separates the responsibilities of the Agent (Simulator Layer), World (Environments Layer), and Benchmark (Task & Dataset Layer). Our primary contribution is a step towards full-stack development in embodied VLN, including (1) a new set of 263 large-scale, fully interactive 3D environments; (2) a physics-aware agent embodiment; (3) a comprehensive evaluation on existing literature; and (4) a unified MoE-based model.

ACKNOWLEDGMENTS

We thank Zihan Wang for the helpful discussion on the HNR implementation, and Jiazhao Zhang for the UniNavid and NFM results. We also appreciate Jian Zhou’s advice on the Isaac Simulator, and Yang Li for his assistance with GPU scheduling.

REFERENCES

- Dong An, Yuankai Qi, Yangguang Li, Yan Huang, Liang Wang, Tieniu Tan, and Jing Shao. Bevbort: Topo-metric map pre-training for language-guided navigation. *arXiv preprint arXiv:2212.04385*, 2022.
- Dong An, Hanqing Wang, Wenguan Wang, Zun Wang, Yan Huang, Keji He, and Liang Wang. Etpnav: Evolving topological planning for vision-language navigation in continuous environments. *arXiv preprint arXiv:2304.03047*, 2023.
- Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3674–3683, 2018.
- Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibao Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.
- Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 15791–15801, 2025.
- Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015.
- Wenzhe Cai, Jiaqi Peng, Yuqiang Yang, Yujian Zhang, Meng Wei, Hanqing Wang, Yilun Chen, Tai Wang, and Jiangmiao Pang. Navdp: Learning sim-to-real navigation diffusion policy with privileged information guidance. *arXiv preprint arXiv:2505.08712*, 2025.
- Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niebner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. In *2017 International Conference on 3D Vision (3DV)*, pp. 667–676. IEEE, 2017.
- Howard Chen, Alane Suhr, Dipendra Misra, Noah Snively, and Yoav Artzi. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12538–12547, 2019.
- Jiaqi Chen, Bingqian Lin, Ran Xu, Zhenhua Chai, Xiaodan Liang, and Kwan-Yee K Wong. Mapgpt: Map-guided prompting for unified vision-and-language navigation. *arXiv preprint arXiv:2401.07314*, 2024.
- Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. History aware multimodal transformer for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 34:5834–5847, 2021.
- Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Think global, act local: Dual-scale graph transformer for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16537–16547, 2022.
- C Chi et al. Diffusion policy: visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.
- Xiangxiang Chu, Limeng Qiao, Xinyang Lin, Shuang Xu, Yang Yang, Yiming Hu, Fei Wei, Xinyu Zhang, Bo Zhang, Xiaolin Wei, et al. Mobilevlm: A fast, reproducible and strong vision language assistant for mobile devices. *arXiv preprint arXiv:2312.16886*, 2023.

- Jiafei Duan, Samson Yu, Hui Li Tan, Hongyuan Zhu, and Cheston Tan. A survey of embodied ai: From simulators to research tasks. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(2):230–244, 2022.
- Xiaofeng Gao, Qiaozi Gao, Ran Gong, Kaixiang Lin, Govind Thattai, and Gaurav S Sukhatme. Dialfred: Dialogue-enabled agents for embodied instruction following. *IEEE Robotics and Automation Letters*, 7(4):10049–10056, 2022.
- Ran Gong, Jiangyong Huang, Yizhou Zhao, Haoran Geng, Xiaofeng Gao, Qingyang Wu, Wensi Ai, Ziheng Zhou, Demetri Terzopoulos, Song-Chun Zhu, et al. Arnold: A benchmark for language-grounded task learning with continuous states in realistic 3d scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 20483–20495, 2023.
- Weituo Hao, Chunyuan Li, Xiujuan Li, Lawrence Carin, and Jianfeng Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13137–13146, 2020.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.
- Haodong Hong, Yanyuan Qiao, Sen Wang, Jiajun Liu, and Qi Wu. General scene adaptation for vision-and-language navigation. *arXiv preprint arXiv:2501.17403*, 2025.
- Yicong Hong, Cristian Rodriguez-Opazo, Qi Wu, and Stephen Gould. Sub-instruction aware vision-and-language navigation. *arXiv preprint arXiv:2004.02707*, 2020.
- Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. A recurrent vision-and-language bert for navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1643–1653, June 2021.
- Yicong Hong, Zun Wang, Qi Wu, and Stephen Gould. Bridging the gap between learning in discrete and continuous environments for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15439–15449, 2022.
- Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, Lewei Lu, Xiaosong Jia, Qiang Liu, Jifeng Dai, Yu Qiao, and Hongyang Li. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.
- Gabriel Ilharco, Vihan Jain, Alexander Ku, Eugene Ie, and Jason Baldridge. General evaluation for instruction conditioned navigation using dynamic time warping. *arXiv preprint arXiv:1907.05446*, 2019.
- InternRobotics. InternNav: InternRobotics’ open platform for building generalized navigation foundation models. <https://github.com/InternRobotics/InternNav>, 2025.
- Muhammad Zubair Irshad, Chih-Yao Ma, and Zsolt Kira. Hierarchical cross-modal agent for robotics vision-and-language navigation. In *2021 IEEE international conference on robotics and automation (ICRA)*, pp. 13238–13246. IEEE, 2021.
- Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. Stay on the path: Instruction fidelity in vision-and-language navigation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1862–1872, 2019.
- Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

- Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision*, pp. 104–120. Springer, 2020.
- Jacob Krantz, Shurjo Banerjee, Wang Zhu, Jason Corso, Peter Anderson, Stefan Lee, and Jesse Thomason. Iterative vision-and-language navigation. *arXiv preprint arXiv:2210.03087*, 2022.
- Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4392–4412, 2020.
- Boqi Li, Siyuan Li, Weiyi Wang, Anran Li, Zhong Cao, and Henry X Liu. Boosting zero-shot vln via abstract obstacle map-based waypoint prediction with topograph-and-visitinfo-aware prompting. *arXiv preprint arXiv:2509.20499*, 2025a.
- Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabriel Levine, Wensi Ai, Benjamin Martinez, et al. Behavior-1k: A human-centered, embodied ai benchmark with 1,000 everyday activities and realistic simulation. *arXiv preprint arXiv:2403.09227*, 2024a.
- Heng Li, Minghan Li, Zhi-Qi Cheng, Yifei Dong, Yuxuan Zhou, Jun-Yan He, Qi Dai, Teruko Mitamura, and Alexander G Hauptmann. Human-aware vision-and-language navigation: Bridging simulation to reality with dynamic human interactions. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024b.
- Yanwei Li, Chengyao Wang, and Jiaya Jia. Llama-vid: An image is worth 2 tokens in large language models. In *European Conference on Computer Vision*, pp. 323–340. Springer, 2024c.
- Zerui Li, Gengze Zhou, Haodong Hong, Yanyan Shao, Wenqi Lyu, Yanyuan Qiao, and Qi Wu. Ground-level viewpoint vision-and-language navigation in continuous environments. *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025b.
- Shubo Liu, Hongsheng Zhang, Yuankai Qi, Peng Wang, Yanning Zhang, and Qi Wu. Aerialvln: Vision-and-language navigation for uavs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15384–15394, 2023.
- Yuxing Long, Xiaoqi Li, Wenzhe Cai, and Hao Dong. Discuss before moving: Visual language navigation via multi-expert discussions. *arXiv preprint arXiv:2309.11382*, 2023.
- Ziqiao Ma, Benjamin VanDerPloeg, Cristian-Paul Bara, Yidong Huang, Eui-In Kim, Felix Gervits, Matthew Marge, and Joyce Chai. Dorothie: Spoken dialogue for handling unexpected situations in interactive autonomous driving agents. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 4800–4822, 2022.
- Dipendra Misra, Andrew Bennett, Valts Blukis, Eyvind Niklasson, Max Shatkhin, and Yoav Artzi. Mapping instructions to actions in 3d environments with visual goal prediction. *arXiv preprint arXiv:1809.00786*, 2018.
- Khanh Nguyen and Hal Daumé III. Help, anna! visual navigation with natural multimodal assistance via retrospective curiosity-encouraging imitation learning. *arXiv preprint arXiv:1909.01871*, 2019.
- Khanh Nguyen, Debadeepta Dey, Chris Brockett, and Bill Dolan. Vision-based navigation with language-based assistance via imitation learning with indirect intervention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12527–12537, 2019.
- Phuong DH Nguyen, Yasmin Kim Georgie, Ezgi Kayhan, Manfred Eppe, Verena Vanessa Hafner, and Stefan Wermter. Sensorimotor representation learning for an “active self” in robots: a model survey. *KI-Künstliche Intelligenz*, 35:9–35, 2021.
- NVIDIA. NVIDIA Isaac Sim. <https://developer.nvidia.com/isaac-sim>, 2021. Accessed: 2025-11-14.

- Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.
- Aishwarya Padmakumar, Jesse Thomason, Ayush Shrivastava, Patrick Lange, Anjali Narayan-Chen, Spandana Gella, Robinson Piramuthu, Gokhan Tur, and Dilek Hakkani-Tur. Teach: Task-driven embodied agents that chat. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 2017–2025, 2022.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans (eds.), *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL <https://aclanthology.org/D14-1162/>.
- Pixar Animation Studios. Universal Scene Description (USD) project. <https://openusd.org/dev/intro.html>, 2021. Accessed: 2025-11-10.
- Xavier Puig, Eric Undersander, Andrew Szot, Mikael Dallaire Cote, Tsung-Yen Yang, Ruslan Partsey, Ruta Desai, Alexander Clegg, Michal Hlavac, So Yeon Min, et al. Habitat 3.0: A co-habitat for humans, avatars, and robots. In *The Twelfth International Conference on Learning Representations*, 2024.
- Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. Reverie: Remote embodied visual referring expression in real indoor environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9982–9991, 2020.
- Yanyuan Qiao, Wenqi Lyu, Hui Wang, Zixu Wang, Zerui Li, Yuan Zhang, Minghui Tan, and Qi Wu. Open-nav: Exploring zero-shot vision-and-language navigation in continuous environment with open-source llms. *arXiv preprint arXiv:2409.18794*, 2024.
- Yanyuan Qiao, Haodong Hong, Wenqi Lyu, Dong An, Siqi Zhang, Yutong Xie, Xinyu Wang, and Qi Wu. Navbench: Probing multimodal large language models for embodied navigation. *arXiv preprint arXiv:2506.01031*, 2025.
- Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9339–9347, 2019.
- Xiangyu Shi, Zerui Li, Wenqi Lyu, Jiatong Xia, Feras Dayoub, Yanyuan Qiao, and Qi Wu. Smartway: Enhanced waypoint prediction and backtracking for zero-shot vision-and-language navigation. *arXiv preprint arXiv:2503.10069*, 2025a.
- Xiangyu Shi, Zerui Li, Yanyuan Qiao, and Qi Wu. Fast-smartway: Panoramic-free end-to-end zero-shot vision-and-language navigation. *arXiv preprint arXiv:2511.00933*, 2025b.
- Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10740–10749, 2020.
- Xinshuai Song, Weixing Chen, Yang Liu, Weikai Chen, Guanbin Li, and Liang Lin. Towards long-horizon vision-language navigation: Platform, benchmark and method. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 12078–12088, 2025.
- Ajay Sridhar, Dhruv Shah, Catherine Glossop, and Sergey Levine. Nomad: Goal masked diffusion policies for navigation and exploration. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 63–70. IEEE, 2024.
- Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

- Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John M. Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel X. Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems*, pp. 251–266, 2021.
- Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 5100–5111, 2019.
- Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In *Conference on Robot Learning*, pp. 394–406, 2020.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- Hanqing Wang, Jiahe Chen, Wensi Huang, Qingwei Ben, Tai Wang, Boyu Mi, Tao Huang, Siheng Zhao, Yilun Chen, Sizhe Yang, et al. Grutopia: Dream general robots in a city at scale. *arXiv preprint arXiv:2407.10943*, 2024a.
- Liuyi Wang, Xinyuan Xia, Hui Zhao, Hanqing Wang, Tai Wang, Yilun Chen, Chengju Liu, Qijun Chen, and Jiangmiao Pang. Rethinking the embodied gap in vision-and-language navigation: A holistic study of physical and visual disparities. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9455–9465, 2025.
- Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6629–6638, 2019.
- Zihan Wang, Xiangyang Li, Jiahao Yang, Yeqi Liu, and Shuqiang Jiang. Gridmm: Grid memory map for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15625–15636, 2023a.
- Zihan Wang, Xiangyang Li, Jiahao Yang, Yeqi Liu, Junjie Hu, Ming Jiang, and Shuqiang Jiang. Lookahead exploration with neural radiance representation for continuous vision-language navigation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13753–13762, 2024b.
- Zihan Wang, Xiangyang Li, Jiahao Yang, Yeqi Liu, and Shuqiang Jiang. Sim-to-real transfer via 3d feature fields for vision-and-language navigation. *arXiv preprint arXiv:2406.09798*, 2024c.
- Zun Wang, Jialu Li, Yicong Hong, Yi Wang, Qi Wu, Mohit Bansal, Stephen Gould, Hao Tan, and Yu Qiao. Scaling data generation in vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12009–12020, 2023b.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.
- Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. In *International Conference on Learning Representations*, 2019.
- Fei Xia, Amir R Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9068–9079, 2018.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

- Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 11975–11986, 2023.
- Beichen Zhang, Pan Zhang, Xiaoyi Dong, Yuhang Zang, and Jiaqi Wang. Long-clip: Unlocking the long-text capability of clip. In *European conference on computer vision*, pp. 310–325. Springer, 2024a.
- Jiazhao Zhang, Kunyu Wang, Shaoan Wang, Minghan Li, Haoran Liu, Songlin Wei, Zhongyuan Wang, Zhizheng Zhang, and He Wang. Uni-navid: A video-based vision-language-action model for unifying embodied navigation tasks. *arXiv preprint arXiv:2412.06224*, 2024b.
- Jiazhao Zhang, Anqi Li, Yunpeng Qi, Minghan Li, Jiahang Liu, Shaoan Wang, Haoran Liu, Gengze Zhou, Yuze Wu, Xingxing Li, et al. Embodied navigation foundation model. *arXiv preprint arXiv:2509.12129*, 2025.
- Chongyang Zhao, Yuankai Qi, and Qi Wu. Mind the gap: Improving success rate of vision-and-language navigation by revisiting oracle success routes. In *Proceedings of the 31st ACM International Conference on Multimedia*, pp. 4349–4358, 2023.
- Gengze Zhou, Yicong Hong, Zun Wang, Chongyang Zhao, Mohit Bansal, and Qi Wu. Same: Learning generic language-guided visual navigation with state-adaptive mixture of experts. *arXiv preprint arXiv:2412.05552*, 2024a.
- Gengze Zhou, Yicong Hong, and Qi Wu. Navgpt: Explicit reasoning in vision-and-language navigation with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 7641–7649, 2024b.
- Gengze Zhou, Yicong Hong, Zun Wang, Xin Eric Wang, and Qi Wu. Navgpt-2: Unleashing navigational reasoning capability for large vision-language models. In *European Conference on Computer Vision*, pp. 260–278. Springer, 2025.
- Qinhong Zhou, Sunli Chen, Yisong Wang, Haozhe Xu, Weihua Du, Hongxin Zhang, Yilun Du, Joshua B Tenenbaum, and Chuang Gan. Hazard challenge: Embodied decision making in dynamically changing environments. In *The Twelfth International Conference on Learning Representations*, 2024c.
- Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. Soon: Scenario oriented object navigation with graph-based exploration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12689–12699, 2021.
- Wang Zhu, Hexiang Hu, Jiacheng Chen, Zhiwei Deng, Vihan Jain, Eugene Ie, and Fei Sha. Baby-Walk: Going farther in vision-and-language navigation by taking baby steps. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2539–2556. Association for Computational Linguistics, 2020.
- Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pp. 3357–3364. IEEE, 2017.

APPENDIX

A MODEL DETAILS & TRAINING RECIPE

In this section, we provide comprehensive implementation details for the models and training pipelines discussed in the main paper. We first elaborate on the architecture of our unified baseline, GAMA, followed by a description of the comparative baseline methods. Finally, we present the specific experimental setups and hyperparameter configurations for both offline and online training phases.

A.1 GAMA: GENERAL-PURPOSE AGENT FOR MULTI-TASK NAVIGATION

Our unified baseline GAMA integrates a hierarchical navigation backbone (Wang et al., 2024b) with a novel state-adaptive navigation policy (Zhou et al., 2024a). In the following, we detail the implementation of the hierarchical representation backbone in Sec. A.1.1, followed by the state-adaptive navigation policy in Sec. A.1.2

A.1.1 HIERARCHICAL NEURAL RADIANCE BACKBONE

Unlike standard recurrent policies that rely solely on current observations, GAMA adopts the lookahead exploration strategy from HNR (Wang et al., 2024b). It utilizes a hierarchical neural radiance representation to hallucinate semantic representations of future environments. The backbone consists of three key stages:

Feature Cloud Encoding: The agent maintains a feature cloud \mathcal{M} that stores fine-grained visual semantics and 3D spatial information. At step t , it extracts grid features $\mathbf{g}_{t,j}$ from the observation using a vision encoder (*e.g.*, CLIP-ViT) and projects them into 3D space using depth $\mathbf{d}_{t,j}$ and camera intrinsics \mathbf{K} :

$$P_{t,j} = [\mathbf{d}_{t,j} \mathbf{R}^{-1} \mathbf{K}^{-1} [h, w, 1]^T - \mathbf{T}]^T, \quad (2)$$

where $P_{t,j}$ is the 3D coordinate. The cloud is updated as $\mathcal{M}_t = \mathcal{M}_{t-1} \cup \{[\mathbf{g}_{t,j}, P_{t,j}, \theta_{t,j}, s_{t,j}]\}$, including orientation θ and scale s .

Future Prediction via Volume Rendering: To predict the feature map of a candidate lookahead view, the model employs neural volume rendering. For a sampled point \mathcal{P}_n along a camera ray, it retrieves the K -nearest features from \mathcal{M} using a KD-Tree. To ensure translational invariance, relative positional embeddings are computed:

$$\mathbf{q}_k = \text{LN}(\mathbf{W}_1 [P_k^{rel}, \theta_k^{rel}, s_k]), \quad (3)$$

where P^{rel} and θ^{rel} are relative coordinates and orientation. These are aggregated to produce a density σ_n and latent vector \mathbf{r}_n . The final region feature $\mathbf{R}_{h,w}$ is computed via integration:

$$\mathbf{R}_{h,w} = \sum_{n=1}^N \tau_n (1 - \exp(-\sigma_n \Delta_n)) \mathbf{r}_n, \quad \tau_n = \exp\left(-\sum_{i=1}^{n-1} \sigma_i \Delta_i\right). \quad (4)$$

These rendered features allow the agent to “see” potential future states without physical movement.

Topological Graph Construction: The predicted future views are organized into a topological graph containing visited nodes, candidate nodes, and lookahead nodes. The graph is encoded using Graph-Aware Self-Attention (GASA), which incorporates the geodesic distance matrix E into the attention mechanism:

$$\text{GASA}(\mathcal{V}) = \text{Softmax}\left(\frac{\mathcal{V} \mathbf{W}_q (\mathcal{V} \mathbf{W}_k)^T}{\sqrt{d}} + E \mathbf{W}_e\right) \mathcal{V} \mathbf{W}_v. \quad (5)$$

This graph representation \mathcal{V} serves as the input to the navigation policy.

A.1.2 MULTIMODAL NAVIGATION POLICY

Building upon the topological representations from the backbone, GAMA employs a State-Adaptive Mixture of Experts (SAME) to execute precise navigation actions aligned with language instructions.

Instruction-Vision Alignment: At each timestep, the agent receives the visual features \hat{v}_t (derived from the current view) and updates the topological map $\hat{\mathcal{G}}_t$. To align the visual context with the language instruction \mathcal{W} , a local cross-modal encoder excites visual features conditioned on text via Cross-Attention:

$$\text{CrossAttn}(\hat{v}_t, \hat{\mathcal{W}}) = \text{Softmax} \left(\frac{\hat{v}_t W_q (\hat{\mathcal{W}} W_k)^T}{\sqrt{d}} \right) \hat{\mathcal{W}} W_v. \quad (6)$$

The output embedding for the current view is \hat{v}'_t . Similarly, a global cross-modal encoder encodes the map to yield map node embeddings $\hat{v}'_{g,i}$:

$$\hat{\mathcal{G}}'_t = \text{CrossAttn}(\hat{\mathcal{G}}_t, \hat{\mathcal{W}}). \quad (7)$$

State-Adaptive Navigation Policy: In the previous HNR (Wang et al., 2024b), navigation scores are computed via static Feed-Forward Networks. GAMA replaces this with the SAME mechanism to handle task fragmentation. First, a multimodal routing feature $\mathbf{x}_r^{\text{multi}}$ is computed to capture the alignment between the visual context and the instruction. We utilize the global representation of the current monocular view aligned with the instruction’s [CLS] token:

$$\mathbf{x}_r^{\text{multi}} = W_m \left[\hat{v}_t^{\text{CLS}} ; \hat{\mathcal{W}}^{\text{CLS}} \right], \quad (8)$$

where W_m is a projection layer. The router \mathcal{R} predicts activation probabilities for N experts:

$$\mathcal{P}(\mathbf{x}_r) = \text{Softmax}(W_r \mathbf{x}_r^{\text{multi}}). \quad (9)$$

The local observation features \hat{v}'_t and global map features $\hat{v}'_{g,i}$ are then processed by the top- k selected experts. The navigation scores are calculated as:

$$\begin{aligned} s^l &= \sum_{k \in \mathcal{T}} \mathcal{P}(\mathbf{x}_r)_k \cdot f_k^{\text{local}}(\hat{v}'_t), \\ s^g_i &= \sum_{k \in \mathcal{T}} \mathcal{P}(\mathbf{x}_r)_k \cdot f_k^{\text{global}}(\hat{v}'_{g,i}). \end{aligned} \quad (10)$$

Finally, the global and local scores are fused using a learnable parameter σ_t to determine the action:

$$s_i = \sigma_t s^l + (1 - \sigma_t) s^g_i. \quad (11)$$

To ensure diverse expert usage, we apply a load balancing loss $\mathcal{L}_{\text{balance}} = N \sum \mathcal{F}_i \mathcal{D}_i$ during training.

A.2 BASELINE MODELS

To evaluate the effectiveness of our approach, we compare it against a comprehensive set of baselines ranging from traditional recurrent architectures to recent foundation models. We categorize these baselines into three groups based on their action spaces and architectural paradigms.

A.2.1 RECURRENT AND GRAPH-BASED METHODS

Sequence-to-Sequence (Seq2Seq). This baseline, adapted from (Krantz et al., 2020), represents a foundational VLN architecture. It comprises three main modules:

1. An **instruction encoder** that processes the language input ($I = \{w_i\}_{i=1}^L$) using an LSTM on GLoVe embeddings (Pennington et al., 2014).
2. An **observation encoder** that uses a pre-trained ResNet50 (He et al., 2016) for RGB visual features (V_t) and a separate pre-trained model (from point-goal navigation) for depth features (D_t).
3. A **recurrent policy decoder** (GRU) that integrates these inputs to predict the next action.

At each timestep t , the GRU updates its hidden state h_t by combining the representations of the current visual inputs and the static instruction. The action a_t is then predicted via a linear layer over the hidden state.

$$h_t = \text{GRU}([V_t, D_t, I], h_{t-1}), \quad (12)$$

$$a_t = \arg \max_a \text{softmax}(W_a h_t + b_a). \quad (13)$$

Cross-modal Attention (CMA). The CMA model extends the Seq2Seq architecture by incorporating a more sophisticated dual-GRU attention mechanism. Instead of a single recurrent unit, CMA employs two:

- A **first GRU** tracks the observational state. It takes the visual inputs (V_t, D_t) and the *previous action* a_{t-1} (as a 32-dimensional embedding) to produce a state h_t^{1st} .

$$h_t^{1st} = \text{GRU}([V_t, D_t, a_{t-1}], h_{t-1}^{1st}). \quad (14)$$

- This state h_t^{1st} is then used as a query in a scaled dot-product attention mechanism to dynamically attend to the instruction features (\hat{I}_t) , visual features (\hat{V}_t) , and depth features (\hat{D}_t) .

$$\hat{I}_t = \text{Attn}(I, h_t^{1st}), \quad (15)$$

$$\hat{V}_t = \text{Attn}(V_t, h_t^{1st}), \quad \hat{D}_t = \text{Attn}(D_t, h_t^{1st}). \quad (16)$$

- A **second GRU** acts as the decision-making module. It receives a comprehensive set of inputs: the attended features $(\hat{I}_t, \hat{V}_t, \hat{D}_t)$, the previous action a_{t-1} , and the hidden state of the first GRU h_t^{1st} .
- The output of this second GRU, h_t^{2nd} , is used to predict the final action a_t .

$$h_t^{2nd} = \text{GRU}([\hat{I}_t, \hat{V}_t, \hat{D}_t, a_{t-1}, h_t^{1st}], h_{t-1}^{2nd}), \quad (17)$$

$$a_t = \arg \max_a \text{softmax}(W_a h_t^{2nd} + b_a). \quad (18)$$

Hierarchical Neural Radiance (HNR). Unlike standard recurrent policies that rely solely on current observations, HNR (Wang et al., 2024b) introduces a lookahead exploration strategy. It utilizes a pre-trained Hierarchical Neural Radiance representation model to predict (hallucinate) the semantic representations of future environments without physical movement. The framework consists of three key stages:

- **Feature Cloud Encoding:** HNR maintains a feature cloud \mathcal{M} that stores fine-grained visual semantics and 3D spatial information. At step t , it extracts grid features $\mathbf{g}_{t,j}$ using CLIP-ViT-B/32 and projects them into 3D space using depth $\mathbf{d}_{t,j}$ and camera intrinsics \mathbf{K} :

$$P_{t,j} = [\mathbf{d}_{t,j} \mathbf{R}^{-1} \mathbf{K}^{-1} [h, w, 1]^T - \mathbf{T}]^T, \quad (19)$$

where $P_{t,j}$ is the 3D coordinate. The cloud is updated as $\mathcal{M}_t = \mathcal{M}_{t-1} \cup \{[\mathbf{g}_{t,j}, P_{t,j}, \theta_{t,j}, s_{t,j}]\}$, including orientation θ and scale s .

- **Future Prediction via Volume Rendering:** To predict the feature map of a future candidate view, HNR employs neural volume rendering. For a sampled point \mathcal{P}_n along a camera ray, the model retrieves the K -nearest features from \mathcal{M} using a KD-Tree. To ensure translational invariance, it computes relative positional embeddings:

$$\mathbf{q}_k = \text{LN}(\mathbf{W}_1 [P_k^{rel}, \theta_k^{rel}, s_k]), \quad (20)$$

where P^{rel} and θ^{rel} are the relative coordinates and orientation between the query point and the stored features. These are aggregated via an MLP to produce a density σ_n and latent vector \mathbf{r}_n . The final region feature $\mathbf{R}_{h,w}$ is computed via integration:

$$\mathbf{R}_{h,w} = \sum_{n=1}^N \tau_n (1 - \exp(-\sigma_n \Delta_n)) \mathbf{r}_n, \quad \tau_n = \exp\left(-\sum_{i=1}^{n-1} \sigma_i \Delta_i\right). \quad (21)$$

A hierarchical encoder then processes these region features to obtain a holistic view representation.

- **Lookahead Planning:** The predicted future views are organized into a topological graph containing visited nodes, candidate nodes, and lookahead nodes. The graph is encoded using Graph-Aware Self-Attention (GASA), which incorporates the geodesic distance matrix E into the attention mechanism:

$$\text{GASA}(\mathcal{V}) = \text{Softmax}\left(\frac{\mathcal{V} \mathbf{W}_q (\mathcal{V} \mathbf{W}_k)^T}{\sqrt{d}} + E \mathbf{W}_e\right) \mathcal{V} \mathbf{W}_v. \quad (22)$$

Finally, the model evaluates path branches by max-pooling the predicted scores of candidate nodes and their connected lookahead nodes: $S^{path} = \max([S^{candidate}, S^{lookahead}])$.

A.2.2 VLM-BASED DISCRETE PLANNERS

Uni-NaVid. Uni-NaVid (Zhang et al., 2024b) reformulates the navigation task as a video-to-text generation problem using a Large Language Model (Vicuna-7B). Given an instruction \mathcal{I} and an ego-centric video history $\mathcal{O}_T = \{\mathbf{x}_1, \dots, \mathbf{x}_T\}$, the model predicts a sequence of $k = 4$ future discrete actions.

To handle long-horizon video efficiently, Uni-NaVid introduces an **Online Visual Token Merging** mechanism inspired by the Atkinson-Shiffrin memory model. Visual features are first extracted via an EVA-CLIP encoder, $\mathbf{X}_{1:T} = \text{Encoder}(\mathbf{x}_{1:T})$, where each frame yields N_x tokens. These tokens are then categorized into three groups with different compression rates using a Grid Pooling operation:

$$\mathbf{X}_{1:T} = \begin{cases} \mathbf{X}_{\text{curr}} = \text{GridPool}(\mathbf{X}_t, \alpha_{\text{curr}}), & \text{if } t = T \\ \mathbf{X}_{\text{short}} = \text{GridPool}(\mathbf{X}_t, \alpha_{\text{short}}), & \text{if } t \in [T - B, T) \\ \mathbf{X}_{\text{long}} = \text{GridPool}(\mathbf{X}_t, \alpha_{\text{long}}), & \text{if } t \in [1, T - B) \end{cases} \quad (23)$$

where $\alpha_{\text{curr}} = 2$, $\alpha_{\text{short}} = 8$, and $\alpha_{\text{long}} = 16$ control the spatial pooling resolution, and $B = 64$ represents the short-term buffer size.

To enable efficient online inference, the model recursively updates these memory banks. New frames are processed and integrated into the short-term memory, while older short-term tokens are compressed into long-term memory:

$$\mathbf{X}_{\text{curr} \rightarrow \text{short}} = \text{GridPool}(\mathbf{X}_{\text{curr}}, \alpha_{\text{short}} / \alpha_{\text{curr}}), \quad (24)$$

$$\mathbf{X}_{\text{short} \rightarrow \text{long}} = \text{GridPool}(\mathbf{X}_{\text{short}}, \alpha_{\text{long}} / \alpha_{\text{short}}). \quad (25)$$

To prevent the long-term memory from growing linearly, Uni-NaVid merges temporally adjacent tokens based on cosine similarity. If the similarity between a candidate token and the existing memory exceeds a threshold $\tau = 0.95$, they are fused:

$$\mathbf{X}_{\text{long}} = \frac{1}{K + 1} (K \mathbf{X}_{\text{long}} + \mathbf{X}_{\text{short} \rightarrow \text{long}}) \quad \text{s.t.} \quad \cos(\mathbf{X}_{\text{long}}, \mathbf{X}_{\text{short} \rightarrow \text{long}}) > \tau, \quad (26)$$

where K is the count of previously merged frames.

Finally, all merged visual tokens are projected via a two-layer MLP, $P_V(\cdot)$, to align with the LLM’s embedding space. The input to the LLM is constructed as:

$$\text{Input} = [\mathbf{E}_{\text{long}}^V, \mathbf{E}_{\text{short}}^V, \mathbf{E}_{\text{curr}}^V, \langle \text{NAV} \rangle, \mathbf{E}_{\text{instr}}^L], \quad (27)$$

where $\langle \text{NAV} \rangle$ is a task indicator token. The model outputs action tokens autoregressively: $\{\mathbf{E}_T^A, \dots, \mathbf{E}_{T+3}^A\}$.

A.2.3 CONTINUOUS TRAJECTORY GENERATION MODELS

Recurrent Diffusion Policy (RDP). Drawing inspiration from the success of diffusion policies in manipulation (Chi et al., 2023) and simpler navigation tasks (Sridhar et al., 2024; Bar et al., 2025; Cai et al., 2025), the RDP baseline adapts this generative paradigm to VLN. This approach utilizes a diffusion-based generative head to enable continuous, multi-step trajectory predictions.

The architecture processes ego-centric RGB-D inputs, where RGB images and language instructions are encoded via LongCLIP (Zhang et al., 2024a), while the depth stream is processed by a pre-trained ResNet50. Feature alignment between the visual and linguistic modalities is established through two multi-head, multi-layer cross-modal attention modules (Vaswani et al., 2017).

At the core of RDP is a Transformer-based diffusion decoder conditioned on the fused cross-modal features c_t . In contrast to discrete planners, RDP predicts a continuous action sequence for the next T steps, denoted as $\{\Delta x_t, \Delta y_t, \Delta yaw_t\}_{t=1}^T$, corresponding to relative displacement and yaw. The model follows the DDPM framework (Ho et al., 2020) for training and sampling, utilizing an iterative denoising process:

$$a_t^{k-1} = \alpha \cdot (a_t^k - \gamma \epsilon_\theta(c_t, a_t^k, k) + \mathcal{N}(0, \mu^2 I)), \quad (28)$$

where k represents the denoising step, ϵ_θ is the noise prediction network, and α, γ, μ serve as noise schedule functions.

To address the specific challenges of VLN, RDP incorporates two key mechanisms:

1. **History Management:** A recurrent GRU module is employed to maintain and update the observation history, allowing the model to capture long-range dependencies effectively.
2. **Stop Prediction:** Since standard diffusion models are not inherently designed for discrete stop/continue decisions, an auxiliary MLP prediction head $\mathcal{S}_{stop}(c_t)$ is added. This head predicts a continuous *stop progression* value \hat{p}_{stop} , which evolves from 0 (start) to 1 (goal).

The total loss function combines the standard diffusion denoising objective with the stop progression error:

$$\mathcal{L}_{RDP} = \text{MSE}(\epsilon^k, \epsilon_\theta(c_t, a_t^0 + \epsilon^k, k)) + \lambda \cdot \text{MSE}(\mathcal{S}_{stop}(c_t), \hat{p}_{stop}), \quad (29)$$

where \hat{p}_{stop} represents the ground-truth stop progress and λ is a weighting hyperparameter.

Navigation Foundation Model (NFM). Proposed by (Zhang et al., 2025), NFM is a generalist navigation policy capable of handling cross-embodiment and cross-task scenarios (*e.g.*, drones, cars, indoor robots). It extends a standard VLM architecture to support multi-view inputs and continuous trajectory prediction.

- **Dual-Scale Visual Encoding:** To capture both semantic semantics and fine-grained details, NFM concatenates features from two pre-trained encoders, DINOv2 (Oquab et al., 2023) and SigLIP (Zhai et al., 2023). To manage token counts, it applies a Grid Pooling strategy with two resolutions:

$$\mathbf{V}^{\text{fine/coarse}} = \text{GridPool}(\mathbf{V}, s), \quad s \in \left\{ \frac{64}{P}, \frac{4}{P} \right\}, \quad (30)$$

where fine-grained tokens ($64 \times C$) are used for the most recent observation to ensure precision, while coarse-grained tokens ($4 \times C$) are used for historical frames to save memory.

- **Temporal-Viewpoint Indicator (TVI) Tokens:** To enable the LLM to distinguish between different camera views and time steps in a multi-view setup, NFM adds specialized learnable embeddings to the visual tokens. For a token at time t and view angle ϕ , the indicator \mathbf{E}_{TVIT} is computed as:

$$\mathbf{E}_{\text{TVIT}} = \mathbf{E}_{\text{Base}} + \mathcal{P}_{\text{time}}(\text{TimePE}(t)) + \mathcal{P}_{\text{angle}}(\text{AnglePE}(\phi)), \quad (31)$$

where TimePE and AnglePE are sinusoidal positional encodings processed by MLPs (\mathcal{P}). This allows the model to maintain geometric awareness ($0 \equiv 2\pi$) and temporal order simultaneously.

- **Budget-Aware Temporal Sampling (BATS):** To handle long-horizon videos within a fixed token budget B_{token} , NFM replaces uniform sampling with an exponential decay strategy inspired by the “forgetting curve.” The sampling probability $P(t)$ for a frame at time t relative to the current time T is:

$$P(t) = (1 - \epsilon)e^{k(t-T)/T} + \epsilon, \quad (32)$$

where ϵ is a lower bound and k is a decay rate solved numerically to satisfy the token budget. This ensures high sampling density for recent observations while retaining sparse historical context.

- **Continuous Trajectory Prediction:** Unlike discrete planners, NFM predicts a sequence of continuous waypoints. The LLM outputs a hidden state E_T^A , which is decoded by a 3-layer MLP (\mathcal{A}_θ) into normalized coordinates. These are then scaled by a task-specific factor α_{task} (*e.g.*, for drone vs. car speeds):

$$\tau_T = \{\mathbf{a}_1, \dots, \mathbf{a}_M\} = \alpha_{\text{task}} \cdot \mathcal{A}_\theta(E_T^A), \quad (33)$$

where $\mathbf{a} \in \mathbb{R}^4$ represents (x, y, z, yaw) .

InternNav-N1. InternNav-N1 (InternRobotics, 2025) proposes a dual-system architecture inspired by the “System 1 vs. System 2” cognitive theory. It decouples navigation into low-frequency reasoning and high-frequency execution to handle long-horizon planning and dynamic obstacle avoidance simultaneously.

- **System 2 (Reasoning Planner):** Built upon a pre-trained VLM (Qwen-VL-2.5 7B (Bai et al., 2025)), System 2 operates at a low frequency (2Hz). It interprets instructions and historical observations to predict a mid-term goal. Initially formulated as a *Pixel Goal* (u, v) on the image plane, this is evolved during joint-tuning into a *Latent Plan*—a set of compact, learnable tokens that bridge the VLM and the policy without geometric ambiguity.
- **System 1 (Agile Executor):** System 1 is a lightweight, multi-modal goal-conditioned diffusion policy running at 30Hz. It takes real-time observations and the asynchronous goal from System 2 to generate continuous trajectories. To ensure robust goal conditioning, System 1 is trained with a *Goal Alignment* objective. Specifically, it encodes both image-goals I_g and pixel-goal masks M_g (derived from coordinates c_g) into embeddings z_{img} and z_{pix} , enforcing them to align with the ground-truth point-goal p_g :

$$\mathcal{L}^{goal} = \frac{1}{N} \sum_{i=1}^N \|\text{MLP}(z_{img}) - p_g\|^2 + \frac{1}{N} \sum_{i=1}^N \|\text{MLP}(z_{pix}) - p_g\|^2. \quad (34)$$

- **Hierarchical Joint Training:** The model employs a two-stage curriculum. First, systems are pre-trained separately using explicit pixel/point goals. Second, they are jointly fine-tuned using latent plans. The overall objective for System 1 combines action generation, critic prediction, and goal alignment:

$$\mathcal{L}^{system1} = \alpha \cdot \mathcal{L}^{act} + \beta \cdot \mathcal{L}^{critic} + \gamma \cdot \mathcal{L}^{goal}, \quad (35)$$

where \mathcal{L}^{act} is the diffusion noise prediction loss and \mathcal{L}^{critic} estimates trajectory safety. This asynchronous design allows the agent to perform collision-free traversal while maintaining long-horizon semantic consistency.

A.2.4 SIMULATOR CONFIGURATION

For a fair comparison to previous works, agents in our experiments are set up with the standard dimensions for R2R-CE. Furthermore, while we trained a specialized waypoint predictor using our collected data for our MLLM Zero-shot benchmarks, we employ the original pre-trained waypoint predictor in online-training experiments to strictly align with established baselines. On the other hand, to facilitate the use of the same waypoints predictor and the powerful pre-trained depth-encoder (Wijmans et al., 2019) in the two benchmarks, as well as to decrease the rendering cost, we adjust the camera parameters in VLNVerse. Some key configurations are listed here, where the commented numbers are the original values.

Online-training Configuration	
SIMULATOR:	TRAINING:
AGENT:	IL:
HEIGHT: 1.50	BATCH_SIZE: 4
RADIUS: 0.30	MAX_TRAJ_LEN: 35
ISAAC_SIM:	EPOCHS: 20
ENABLE_COLLISION: True	SCHEDULE_RATIO: 0.85
COLLISION_THRESH: 0.10	DECAY_TIME: 4
ALLOW_SLIDING: True	LR: CosineAnnealingLR
RGB_SENSOR:	ENV:
WIDTH: 224	NUM_ENVS: 4
HEIGHT: 224	NUM_GPUS: 4
HFOV: 90	
DEPTH_SENSOR:	
WIDTH: 256	
HEIGHT: 256	
HFOV: 90	

B DETAILED METRIC DEFINITION

We evaluate all the models mentioned in Appendix A using a complex set of metrics, the details and definitions can be found in Tab. 8

Table 8: Detailed definitions of evaluation metrics.

Metric	Detailed Definition
Trajectory Length (TL)	The average total distance in meters traversed by the agent from its starting position to its final stopping position across all episodes.
Navigation Error (NE)	The average distance in meters between the agent’s final stopping position and the target goal location. This is calculated for <i>all</i> episodes, regardless of their success status. A lower NE indicates better overall navigation accuracy.
Success Rate (SR)	The percentage of episodes where the agent issues the <code>STOP</code> command within a predefined threshold distance (<i>e.g.</i> , 3 meters) of the target goal location. This is a binary metric (1 for success, 0 for failure) calculated for each episode. For long-horizon tasks, this metric is same as SR_{All} .
Oracle Success Rate (OSR)	The percentage of episodes where the <i>optimal</i> path from the agent’s final stopping position to the goal is shorter than the success threshold (<i>e.g.</i> , 3 meters). This measures “recoverability” and assesses whether the agent stopped in a location from which the goal was realistically reachable.
Success weighted by Path Length (SPL)	A metric evaluating both task completion and efficiency: $\frac{1}{N} \sum_{i=1}^N S_i \frac{L_i}{\max(P_i, L_i)}$, where N is the total episodes, S_i is binary success, L_i is the shortest path length (oracle), and P_i is the agent’s path length. This penalizes successful episodes completed via inefficient, long paths.
Normalized Dynamic Time Warping (nDTW)	A path fidelity metric measuring the similarity between the agent’s generated trajectory and the reference (oracle) path. It computes the optimal alignment between the two sequences of 3D points and normalizes by the reference path length. A higher nDTW (closer to 1.0) indicates the agent’s trajectory closely matched the shape and structure of the optimal path.
SR_n (Success Rate for n-th goal)	In long-horizon tasks, SR_n measures the conditional success rate of reaching the n -th goal. For $n > 1$, this is the percentage of episodes where the agent successfully reached goal n , <i>given</i> that it had already successfully reached all preceding goals $(1, \dots, n - 1)$. SR_1 measures the non-conditional success rate of the first goal.
Collision Rate (CR)	A physics-awareness metric defined as the average number of collision events per navigation step. Calculated by dividing the total collisions by the total navigation actions taken. A lower CR indicates a safer agent with better low-level obstacle avoidance.

C A CLOSE LOOK AT VLNVVERSE

In this section, we first visualize the environment to highlight the topological complexity and rendering quality of VLNVerse. Following this, we elaborate on the data generation pipeline, describing the three-agent system (Describer, Verifier, and Synthesizer). We conclude by presenting visualized examples of various task instances constructed using this pipeline.

C.1 VISUALIZATION OF VLNVVERSE ENVIRONMENT

To complement the quantitative statistics presented in the main text, we provide a qualitative visual analysis of the VLNVverse environments in this section. We showcase the Bird’s eye view (BEV) maps and first-person snapshots to empirically demonstrate the structural diversity and high-fidelity rendering that distinguish our dataset from previous works.

Structural Diversity. Fig. 8 displays a curated selection of BEV visualization from our dataset. These maps highlight the topological complexity of our environments. Specifically, VLNVverse features multi-room layouts with varied connectivity graphs. This structural diversity ensures that agents are rigorously tested on long-horizon planning and complex obstacle avoidance.

Visual Fidelity. Fig. 9 presents a gallery of first-person RGB snapshots captured from the agent’s perspective. These images demonstrate the photorealistic rendering quality enabled by the NVIDIA Isaac Sim engine and our hand-crafted USD assets. Note the high-resolution textures, realistic lighting effects (*e.g.*, dynamic shadows), and fine-grained object details. Such visual fidelity is crucial for bridging the sim-to-real gap, as it compels the agent’s vision encoder to process realistic visual data rather than simplified textures.

C.2 DATA GENERATION PIPELINE

C.2.1 VISUAL OBSERVATION

Based on the collision-free trajectories sampled via A* on the dilated occupancy maps, we generate high-fidelity visual data using NVIDIA Isaac Sim. To ensure rendering stability and eliminate visual artifacts, we implement a rendering warm-up mechanism, where the simulator performs 20 physics and rendering steps before capturing the final frame at each waypoint. The visual configuration differs slightly between offline data and online data:

- **Offline Training Data:** We capture the ego-centric front-view RGB images along the trajectory. The camera is configured with a resolution of 224×224 and a Field of View (FoV) of 90° . We simultaneously record aligned depth maps of size 256×256 , applying a distance filter to clip and normalize pixels exceeding a depth of 10 meters.
- **Online Simulation:** To support more complex spatial awareness during active navigation, the agent is equipped with a panoptic camera rig. At each step, we capture a full 360° view by sampling 12 discrete snapshots at 30° yaw intervals, maintaining the same resolution (224×224) and FoV (90°) as the offline setup.

C.2.2 NAVIGATION INSTRUCTION

Then, we generate the navigation instructions for the sampled paths. We designed a collaborative multi-agent pipeline that fuses structured environment data with natural visual perception. The process follows a Generate-Verify-Synthesize workflow. First, we use the scene-graph prior from the simulator metadata to initialize the factual skeleton. However, while accurate, this prior often lacks visual nuance and relational context. Therefore, we employ a **Describer** (Agent 1) to analyze the rendered images. To filter out visual hallucinations from the Describer, a **Verifier** (Agent 2) cross-checks the visual description against the factual prior. Finally, a **Synthesizer** (Agent 3) aggregates the verified information to generate the final instruction, tailoring the output to specific task requirements and linguistic styles (*e.g.*, formal vs. casual), as follows:

Describer. This agent serves as the primary bridge between the visual environment and linguistic instructions. Leveraging a powerful MLLM, the Describer translates raw pixel data into semantic language. Crucially, its behavior is conditioned on the task definition, operating in two distinct modes:

- **Sequential Description for Fine-grained Navigation:** For tasks requiring step-by-step guidance, the Describer takes the chronological sequence of egocentric RGB frames from the sampled path as input. We employ a strict prompt engineering strategy to ensure the output is navigable. Specifically, the agent is constrained to use the second-person imperative

voice (e.g., “Turn left”, not “The agent turns left”) to mimic natural human commands. Furthermore, the prompt explicitly enforces the inclusion of four key navigational elements: (1) *Landmarks*, (2) *Spatial Information*, (3) *Action Verbs*, and (4) a precise *End Point* description. This ensures the generated text is not merely a video caption, but a functional set of actionable instructions.

- **Target Profiling for Coarse-grained Navigation:** For goal-oriented tasks, the Describer shifts focus from the trajectory to the destination state. It analyzes the visual observation of the target object to verify and enrich the scene-graph prior. The agent is prompted to extract specific visual attributes (e.g., color, material, state like “open/closed”) and, critically, to articulate the *spatial relationship* between the target and nearby reference objects (e.g., “the cup *on* the nightstand”). This ensures that the final instruction (“Find the red cup on the nightstand”) is visually grounded and unambiguous.

Prompt for Describer (Coarse-grained Instruction): Analyze the provided image, presumably from room [room_id]^a. Focus specifically on the [target_object_name]. Describe its visual attributes (e.g., color, material if obvious) and its current state (e.g., open/closed, full/empty, on/off if applicable and visible). Also, describe its spatial relationship to the [reference_object_name] if visible and relevant. Mention its relationship to any other significant nearby objects as well. If the target object or reference object isn’t clearly visible, the relationship is ambiguous, or the image itself is problematic (e.g., black, blurry), state that clearly (e.g., “Target object [target_object_name] not clearly visible.” or “Image is unclear.”). Respond ONLY with the concise description or the unclear status. Be factual.

^aroom_id, target_object_name, reference_object_name, target_object_name are from metadata.

Prompt for Describer (Fine-grained Instruction): You are an expert navigation assistant. Your goal is to analyze a **sequence of images**, which represents a continuous first-person path, and generate a clear, step-by-step instruction for a person or a robot to follow that exact path. Your highest priority is the accuracy of the actions.

The images are provided in chronological order and should be treated as frames from a video. The instruction **MUST** be a direct command written in the second-person imperative voice (e.g., “Walk forward,” “Turn left”). You are telling an agent what to do.

Your instruction must include these four elements: 1. **Landmarks:** Refer to areas, rooms, or furniture the agent moves through (e.g., “enter the living room”, “move towards the bathroom”, “pass by a couch”). 2. **Spatial Information:** Identify the direction of landmarks relative to the agent (e.g., “with the table on your left”, “the chairs are on the right-hand side”). 3. **Actions:** Detail the agent’s movements using command verbs (e.g., “turn left at the hallway”, “walk straight past the sofa”, “go through the doorway”). 4. **End Point:** Clearly describe the final stopping position (e.g., “Stop near the sink in front of you”, “Stay in front of the TV”, “Stop to the left of the bed”).

EXAMPLE OF A PERFECT INSTRUCTION: “Walk past a dining table on the left and a living room on the right, then turn right into a hallway. Proceed straight down the hallway, and then turn left to enter a bathroom, stopping in front of the window.”

IMPORTANT RULES: - **DO NOT** use third-person descriptive words like “Walks,” “Moves,” “Enters,” or “Proceeds.” - **DO NOT** narrate what is happening like a video caption. Give direct commands. - Avoid using “Move backward.”

Now, generate the navigation instruction for the following image sequence.

Verifier. While the Describer provides visual richness, it is susceptible to hallucination. Conversely, the scene-graph prior is factually rigid but blind to dynamic visual nuances (e.g., object states). The Verifier acts as a *logic-driven* fusion module to reconcile these two data streams. It receives a batch of rigid, template-based instructions (derived from the prior) alongside the Describer’s visual caption. To integrate them, we implement a robust conditional fusion logic:

- **Visual Priority for Nuance:** The agent first evaluates the informativeness of the visual caption. If the vision provides specific, physically plausible spatial relationships or dynamic attributes (e.g., “the door is *open*”, “the *red* cup”) that differ from the generic prior, the

agent is explicitly instructed to **prioritize** the visual evidence. This ensures the instruction reflects the actual rendered reality rather than stale metadata.

- **Factual Fallback for Safety:** If the visual input is deemed ambiguous (*e.g.*, marked as “unclear” or “not visible”) or describes an implausible relationship, the agent triggers a fallback mechanism, reverting to the spatial relationships defined by the scene-graph Prior.

This two-step logic effectively filters out visual noise while preserving the rich, grounded details necessary for natural instruction generation:

Prompt for Verifier: Your task is to analyze 10 rigid text instructions AND an image caption describing the scene. The 10 text instructions share the same semantic goal. The caption provides a targeted description of the goal object, its state/attributes, and its spatial relationships based on an image. ****Fusion Rule for Spatial Relationship & Details:****

- Compare the spatial relationship described in the IMAGE CAPTION (specifically between the target and reference objects) with the one consistently described in the TEXT instructions.
 - ****IF**** the caption is informative (not “unclear”, “not visible”, etc.) AND describes a relationship between the correct objects AND this relationship seems physically plausible AND it differs from the text relationship (especially if the text relationship seems implausible): ****PRIORITIZE** the spatial relationship from the IMAGE CAPTION. ****** Incorporate relevant attributes/state details from the caption (*e.g.*, color, open/closed) naturally into the generated instructions.
 - ****ELSE**** (caption is uninformative, describes wrong objects, relationship is unclear, or matches the text): ****Use** the spatial relationship from the TEXT instructions. ****** You may still incorporate object attributes/state details from the caption if available and relevant.
****[Text Instructions]**** instructions_text^a ****[/Text Instructions]****
****[Image Caption (Targeted Description)]**** image_caption^b ****[/Image Caption]****

^afrom metadata

^bfrom Describer

Synthesizer. The final stage of our pipeline is the Synthesizer, responsible for transforming the verified semantic content into diverse natural language. Its primary objective is linguistic style adaptation, ensuring the benchmark covers a wide spectrum of user interaction patterns. The Synthesizer operates differently based on the task granularity:

- **Coarse-grained Tasks:** For coarse-grained navigation, where the user intent can vary significantly in tone, the agent is prompted to generate three distinct variations for every episode: (1) *Formal* (precise, objective commands), (2) *Natural* (polite, conversational phrasing), and (3) *Casual* (colloquial, fragmented speech). This stylistic diversity prevents the agent from overfitting to a single linguistic pattern.
- **Fine-grained Tasks:** For step-by-step navigation, where clarity and immediacy are crucial, the Synthesizer is restricted to a single natural style. This ensures that the complex instructions remain clear and easy to follow without the ambiguity.

Finally, the agent enforces a strict JSON output format to ensure the generated data can be programmatically parsed and validated without manual intervention.

Prompt for Synthesizer: You are an expert in natural language generation.

****Core Task:**** Generate three new, high-quality, and distinct instructions (Formal, Natural, Casual) in English based primarily on the TEXT instructions’ goal (action, target object, reference object).

****Output Styles:****

1. ****Formal:**** Precise, objective, complete sentences. Use details sparingly.
2. ****Natural:**** Clear, polite, conversational. Use relevant details naturally.
3. ****Casual:**** Very informal, colloquial, uses fragments. Use key details concisely.

You MUST return ONLY a valid JSON object (and nothing else) with the following structure:

“formal”: “The formal instruction you generated.”,
 “natural”: “The natural instruction you generated.”,
 “casual”: “The casual instruction you generated.”

Robustness of Navigation Instruction. We further assessed the rationality of our three-agent framework by testing it on both GPT-4o and Gemini-2.5-fast. In a blind study, human volunteers were unable to distinguish between instructions generated by different backbones. This underscores the effectiveness of our prompt templates, which successfully guide distinct models to converge on high-quality and fact-based outputs. It demonstrates that our method effectively suppresses model-specific hallucinations by grounding generation in factual priors.

C.3 VISUALIZATION OF TASK INSTANCES

We present qualitative data examples in Fig. 7. Each data entry is constructed around a visual observation sequence (film strips) and paired with instructions at varying granularity. The *Vis.-Ref.* column highlights the modality for visual-reference navigation, providing a cropped target image as the navigational cue. These single-stage episodes serve as the fundamental units for our advanced tasks. We construct long-horizon navigation by chaining these coarse-grained segments, and enable dialogue-based navigation by incorporating an MLLM Oracle (*i.e.*, QwenVL-3 (Yang et al., 2025)) for real-time assistance.

D CLARIFICATION TO HUMAN STUDY

To establish a robust baseline for the proposed tasks, we conducted a comprehensive human study on the test split. This study encompasses both the *Fine-grained* and *Coarse-grained* tasks. Below, we detail the participant selection, experimental interface, data recording mechanism, and ethical considerations.

Participants and Ethics Statement. To ensure the results are representative of general users and free from expert bias, we recruited participants with no prior specific knowledge of Vision Language Navigation (VLN). Strict ethical guidelines were followed during the recruitment and testing phases. All participants provided informed consent prior to the study. To ensure fairness and the validity of the evaluation, participants were only exposed to scenes they had not previously encountered, preventing any advantage gained from prior memorization of the environment layout.

Experimental Design and Workload. To maintain high attention levels and ensure data quality, we controlled the workload for each tester. Each participant was assigned a single, randomized subset of the testing set, consisting of approximately 50 episodes. On average, participants required between 30 to 60 seconds to evaluate a single episode. This duration was deemed optimal to balance efficient data collection with the prevention of user fatigue.

Interface and Interaction. The study was conducted using the Isaac simulator. Upon loading the corresponding data, the navigation instruction was presented to the participant via a pop-up interface. Participants utilized standard keyboard controls to adjust their view and navigate within the 3D environment based on the textual guidance. The termination condition for an episode was user-determined; participants were instructed to close the simulator application once they believed they had successfully reached the target destination described in the instruction.

Data Recording and Evaluation. We implemented a high-frequency logging system to capture precise trajectory data. The agent’s position and heading were recorded every 50 milliseconds and stored in a CSV file. Performance evaluation was conducted by comparing the recorded human trajectory against the ground-truth path. Since the distribution of episodes was balanced across participants, the final human performance reported in this paper is calculated by averaging the scores of all testers within each respective task (Fine-grained and Coarse-grained).

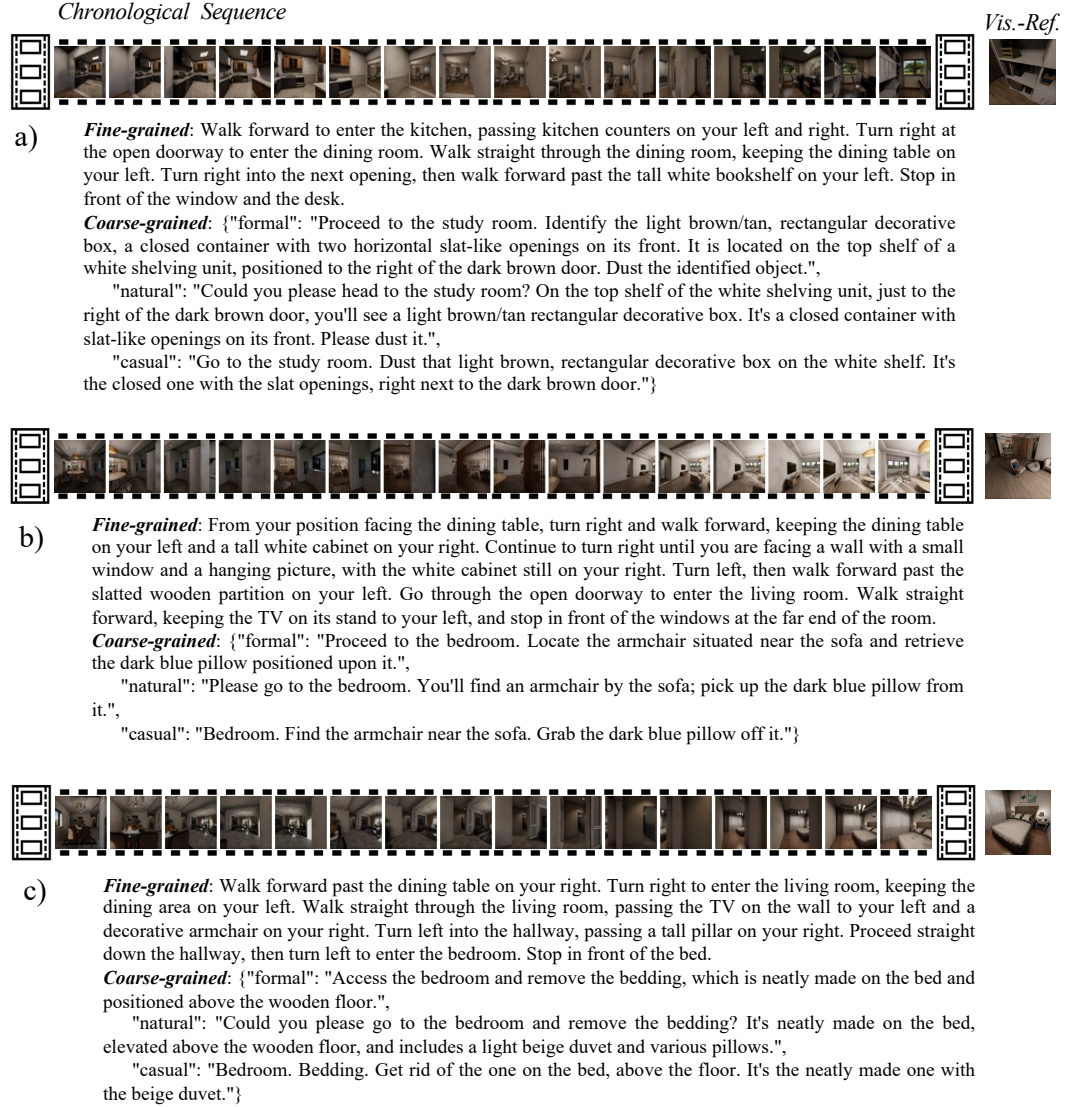


Figure 7: Visualization of sample data representing different task granularities. Each entry consists of a chronological sequence of RGB observations and a visual reference target (*Vis.-Ref.*). (a-c) display three distinct navigation episodes. For each episode, we provide: (1) Fine-grained instructions: Step-by-step navigational guidance describing the path; (2) Coarse-grained instructions: High-level goal descriptions with object interactions, available in three stylistic variations (formal, natural, and casual).





Figure 8: Bird's eye view of VLNVerser environment.





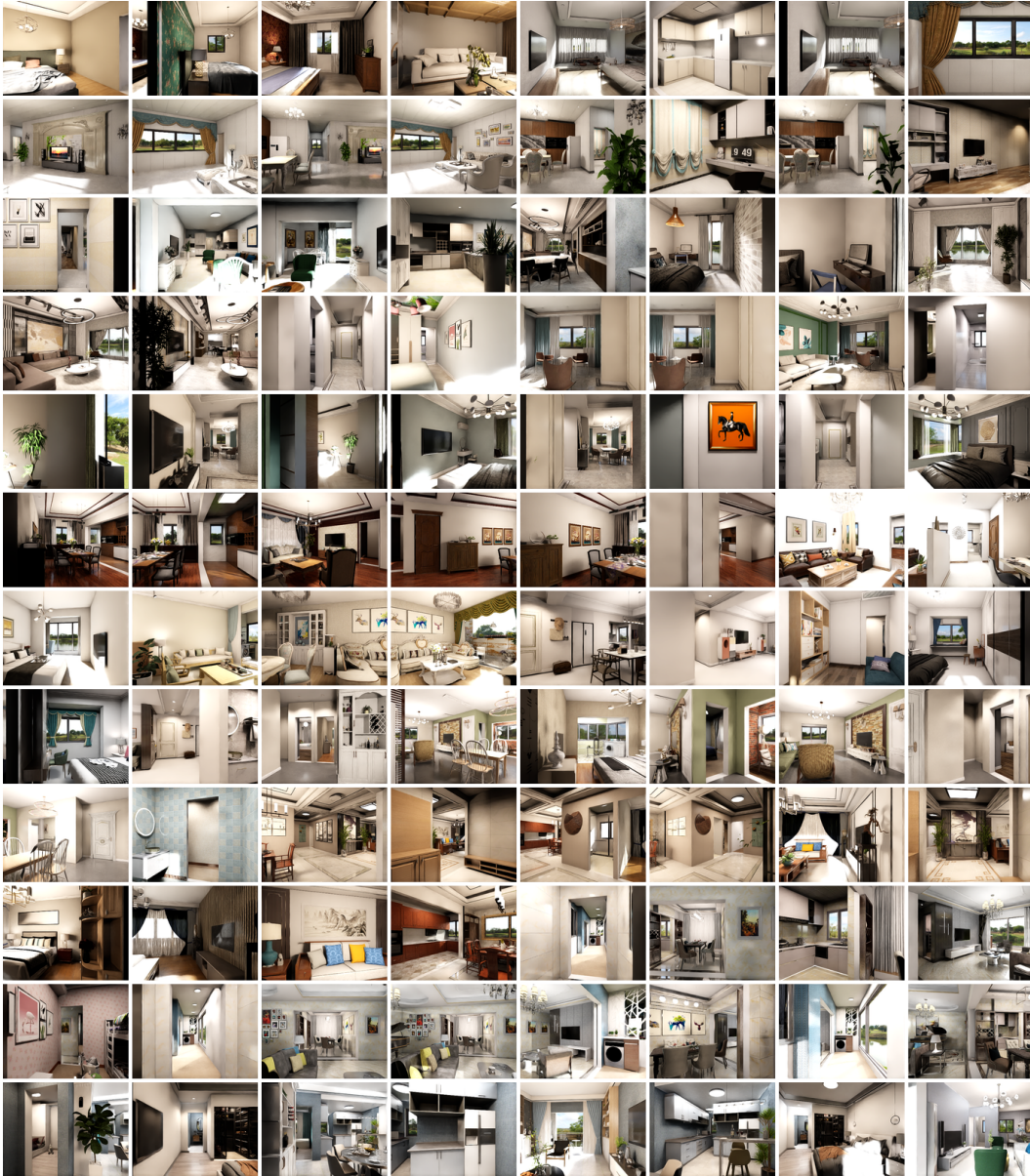


Figure 9: Snapshots of VLNVerse. (Zoom in for details.)