SIMPLEVLN: TOWARDS SIMPLE AND FRONTIER VISION-LANGUAGE NAVIGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-and-Language Navigation (VLN) requires embodied agents to ground language in visual observations and make long-horizon decisions. Existing methods often address these demands through task-specific architectures. As multimodal large language models (MLLMs) become increasingly capable, an important question remains: ***Have we truly unlocked the potential of native MLLMs for VLN before introducing task-specific architecture?*** We present **SimpleVLN**, an architecture-preserving framework developed through studies of navigation formulation, data construction, and optimization. At the formulation level, SimpleVLN combines single-turn decision inputs with fixed-budget global–local memory to efficiently use navigation history without unbounded context growth, while action chunks enable efficient output of multiple primitive actions per model invocation. For data construction, we combine expert demonstrations with policy-dependent DAgger experience and develop a scalable synthesis pipeline that generates large-scale tasks for reinforcement learning. At the optimization level, a progressive IL–DAgger–RL pipeline advances from action imitation to task-level outcome optimization. Comprehensive experiments validate the effectiveness of these three aspects. We instantiate SimpleVLN across multiple MLLM backbones and model sizes, demonstrating its generality. SimpleVLN achieves 76.7% and 80.1% success rates on the R2R-CE and RxR-CE splits, respectively, substantially outperforming existing methods. These results show that the navigation capabilities of native MLLMs can be effectively unlocked without task-specific architectural designs.

1 INTRODUCTION

Vision-and-Language Navigation (VLN) requires embodied agents to follow natural-language instructions in unseen environments (Anderson et al., 2018; Ku et al., 2020), combining visual understanding, language grounding, spatial reasoning, and long-horizon decision making. Existing methods have advanced through task-specific designs for perception and spatial reasoning (Zeng et al., 2026), graph-based modeling (Chen et al., 2022), and hierarchical or dual-system decision making (Wei et al., 2026). These architectures are effective, while multimodal large language models (MLLMs) offer strong general-purpose visual understanding, reasoning, and instruction following (Bai et al., 2025). However, the performance of augmented MLLM systems reflects both the backbone and task-specific components, leaving the capability of a native MLLM alone unclear. This raises an underexplored question: *Have we truly unlocked the potential of native MLLMs for VLN before introducing task-specific architecture?* To answer it, we investigate how far native MLLMs can be advanced for VLN without architectural modifications.

Preserving the architecture still requires adapting the MLLM to sequential navigation. We develop **SimpleVLN** by studying three aspects: formulation, data construction, and optimization (Figure 1). First, at the formulation level, we combine single-turn decision inputs with fixed-budget global–local memory to prevent unbounded context growth while efficiently using navigation history. Action chunking further enables efficient action output by predicting multiple primitive actions per model invocation. Second, for data construction, we combine expert demonstrations with policy-dependent experience collected through DAgger (Ross et al., 2011). We further develop a scalable task-synthesis pipeline to generate large-scale navigation tasks for reinforcement learning. Third, at the optimization level, we investigate a progressive IL–DAgger–RL training pipeline. IL initializes the policy through

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

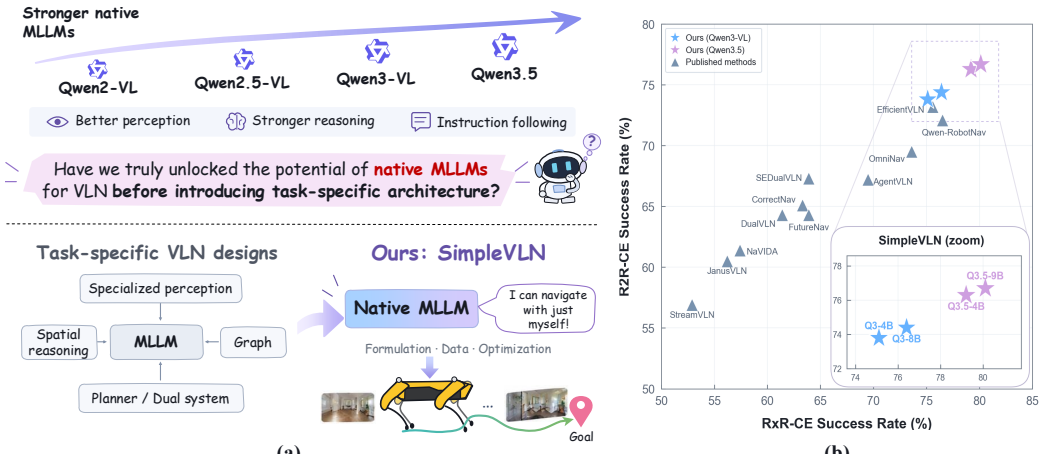


Figure 1: **Motivation and performance comparison of SimpleVLN**. (a) Motivation for using native MLLMs in VLN. (b) Comparison with existing methods on R2R-CE and RxR-CE.

action imitation, DAgger adapts it to policy-induced states, and RL directly optimizes task-level outcomes, progressively improving navigation accuracy.

Comprehensive experiments validate the effectiveness of these three aspects. We instantiate SimpleVLN across multiple MLLM backbones and model sizes, demonstrating its generality. SimpleVLN achieves 76.7% and 80.1% SR on the R2R-CE and RxR-CE *val-unseen* splits, respectively, substantially outperforming existing methods. These results demonstrate that the navigation capabilities of native MLLMs can be effectively unlocked without task-specific architectural designs.

Our contributions are threefold: (1) We introduce SimpleVLN, an architecture-preserving framework that adapts native MLLMs to VLN through single-turn decision inputs, managed visual history, and action-chunk prediction, without task-specific architectural modules. (2) We construct training data from expert demonstrations, policy-dependent DAgger experience, and scalable RL task synthesis, and develop a progressive IL–DAgger–RL pipeline for improving navigation accuracy. (3) We instantiate SimpleVLN across multiple backbone families and model sizes, achieving strong performance on R2R-CE and RxR-CE, while extensive experiments validate the effectiveness of its formulation, data construction, and optimization.

2 Related Works

Vision-and-Language Navigation. Vision-and-Language Navigation (VLN) requires agents to follow natural-language instructions grounded in visual observations (Anderson et al., 2018; Ku et al., 2020). While early VLN operates over predefined discrete viewpoints, VLN-CE (Krantz et al., 2020) extends navigation to continuous environments with low-level actions.

Multimodal Large Language Models for VLN. Recent VLN methods leverage multimodal large language models (MLLMs) for vision–language understanding (Zhang et al., 2024b; Cheng et al., 2024; Zhang et al., 2024a), while introducing streaming contexts (Wei et al., 2025), 3D geometry encoders (Zeng et al., 2026), dual-system architectures (Wei et al., 2026; Huang et al., 2026; Xin et al., 2026; Team, 2025), or explicit chain-of-thought reasoning (Lin et al., 2025; Wang et al., 2026) for long-horizon navigation. In contrast, SimpleVLN directly adapts a pretrained MLLM without architectural modifications, retaining compatibility with existing MLLM inference frameworks.

Training Paradigms for Navigation. Imitation learning trains navigation policies from expert trajectories (Wei et al., 2025; Zhang et al., 2025), but suffers from distribution shift under policy execution. DAgger (Ross et al., 2011) mitigates this issue using supervision on model-induced states (Wei et al., 2025; Zeng et al., 2026), while RL (Zhang et al., 2025; Hu et al., 2026) directly optimizes task-level outcomes. SimpleVLN combines them in a progressive IL-DAgger-RL pipeline from expert supervision to task-completion optimization.

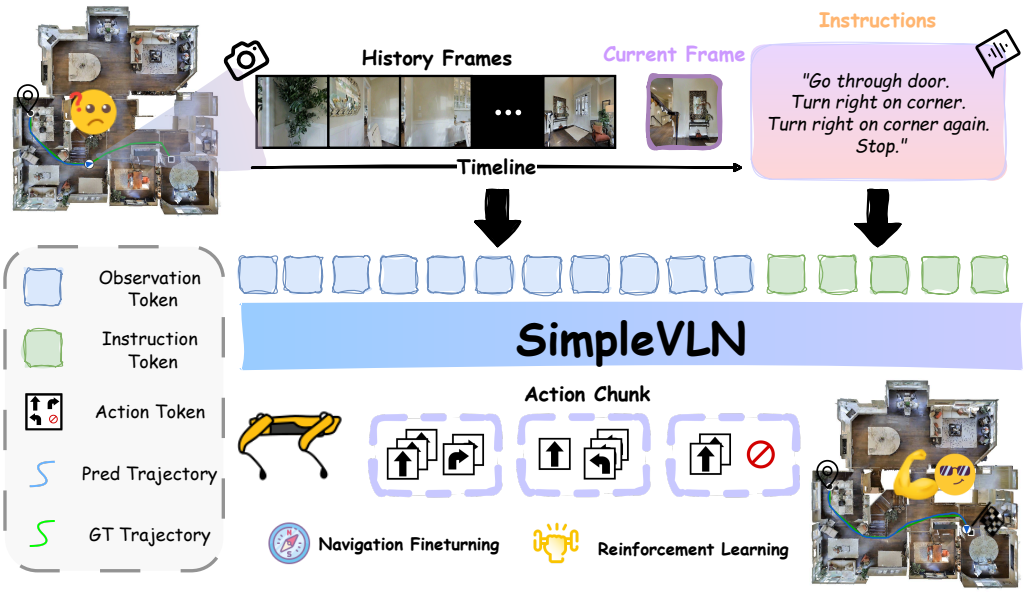


Figure 2: **Overview of SimpleVLN.** At each decision, sampled history frames, current observation, and the instruction are fed to an MLLM, which directly predicts an action chunk for navigation.

3 SIMPLEVLN

3.1 OVERVIEW

SimpleVLN directly adapts off-the-shelf MLLMs (Team et al., 2026; Bai et al., 2025) to VLN without architectural modifications. We study native-MLLM navigation from three perspectives: navigation formulation, data construction, and a progressive optimization strategy that shifts from action imitation to outcome-driven learning through environment interaction.

As illustrated in Figure 2, given a natural-language instruction $\mathcal{I}$, at each primitive environment step k, the agent observes an egocentric RGB image x_k and executes an action u_k from

$$\mathcal{A} = \{\textsc{Forward}(0.25\,\text{m}), \textsc{TurnLeft}(15^\circ), \textsc{TurnRight}(15^\circ), \textsc{Stop}\}.$$

The episode terminates when STOP is executed or the primitive-action budget $S_{\max}$ is reached, and succeeds if the agent stops within 3 m of the target (Krantz et al., 2020).

3.2 DECISION-LEVEL NAVIGATION FORMULATION

We formulate navigation around model decision steps rather than primitive environment steps. At each decision, SimpleVLN reconstructs a bounded context from the instruction, current observation, and sampled history, and uses the native MLLM to predict navigation actions. We first describe the construction of the decision context and then introduce the output action representation.

Single-turn decision input. At each model decision, we construct an independent input rather than append the latest observation and predicted action to the preceding dialogue. Let $t = 1, 2, \ldots$ index model decisions, and let n_t denote the number of primitive actions executed before decision t, with $n_1 = 0$. The current observation is therefore $o_t = x_{n_t}$. The input consists of the instruction $\mathcal{I}$, the current observation o_t, and a sampled visual history H_t; previous dialogue turns and past actions are not included. We denote the resulting visual context by $s_t = (o_t, H_t)$.

Global–local history selection. Before decision t, the primitive-step observation history $(x_0, \ldots, x_{n_t-1})$ grows with environment interaction. Keeping all observations enlarges the input, whereas keeping only recent frames loses earlier route context. We therefore represent history at two complementary temporal scales: local history preserves recent continuity, while global history provides sparse coverage of earlier primitive-step observations. Let L_t contain up to N observations immediately preceding o_t, and let G_t contain up to M observations uniformly sampled from the

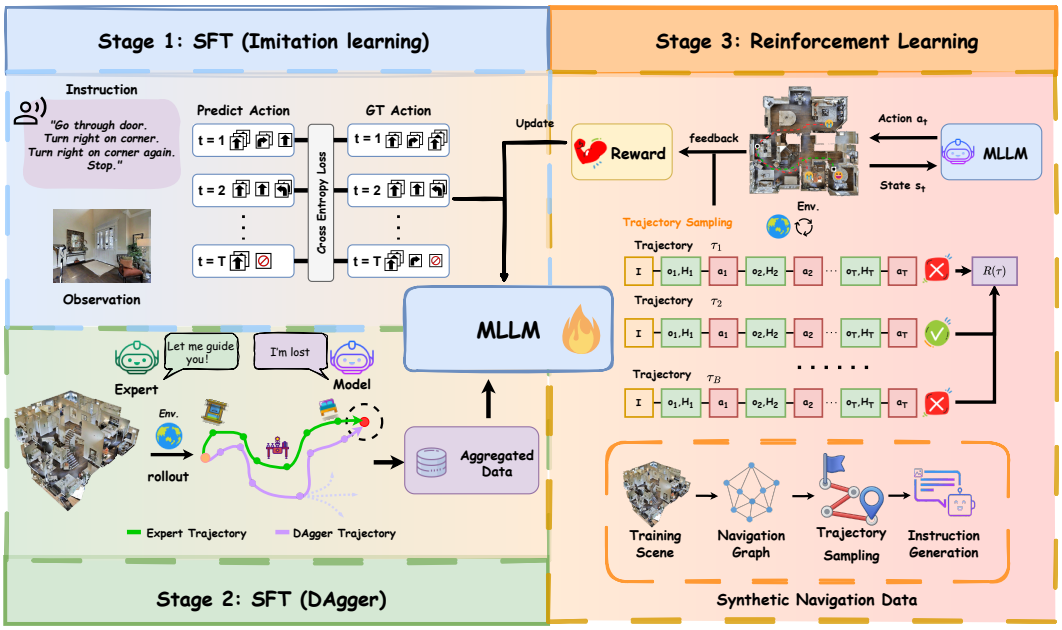


Figure 3: **Training data pipeline of SimpleVLN.** SimpleVLN first learns from expert demonstrations, then collects and aggregates corrective trajectories through DAgger. For RL, it synthesizes navigation episodes from sampled start–goal paths and generated instructions, then optimizes the policy through environment interaction and outcome feedback.

earlier history:

$$L_t = (x_k)_{k=\max(0,n_t-N)}^{n_t-1}, \qquad m_t = \min\{M, \max(0, n_t - N)\},$$
$$G_t = (x_{i_j})_{j=1}^{m_t}, \qquad H_t = G_t \,\|\, L_t. \tag{1}$$

Here, $\|$ denotes sequence concatenation. For $m_t > 0$, the sampled indices satisfy $0 \le i_1 < \cdots < i_{m_t} < \max(0, n_t - N)$, preserving chronological order. When fewer than M earlier observations are available, all are retained. The current observation o_t is provided separately. Thus, each decision uses at most $N + M + 1$ visual inputs, independently of episode length.

Action chunking. On the output side, a single model decision can specify multiple primitive actions. Rather than predicting one primitive action per decision, we follow NaVIDA (Zhu et al., 2026) and predict an action chunk a_t, defined as an ordered sequence of primitive actions. Let K_t denote the number of primitive actions in the chunk; then $a_t = (u_{n_t}, \ldots, u_{n_t+K_t-1})$, and the navigation policy predicts $a_t \sim \pi_\theta(\cdot \mid \mathcal{I}, s_t) = \pi_\theta(\cdot \mid \mathcal{I}, o_t, H_t)$, where π_θ denotes the native MLLM navigation policy. The environment executes the actions in a_t sequentially, producing an observation after each. These observations are retained for later history construction. The MLLM is queried again when the chunk finishes, unless STOP or the primitive-action budget terminates the episode.

3.3 Navigation Data Construction

We construct training data for each stage of post-training. Expert demonstrations provide action supervision, while DAgger adds policy-dependent states encountered during model execution. At the RL stage, reusing the same episodes used for supervised learning would not expand the task distribution or provide genuinely new interaction experience. We therefore synthesize new navigation tasks in the training scenes.

Expert demonstration coverage. We pool trajectories from R2R (Anderson et al., 2018), RxR (Ku et al., 2020), and ScaleVLN (Wang et al., 2023) to broaden expert behavior coverage. To match the decision-level inference interface in Section 3.2, each trajectory is converted into supervised samples with $(\mathcal{I}, s_t)$ as input and the corresponding expert action chunk $a_t^\star$ as the target, where $s_t = (o_t, H_t)$.

Policy-dependent experience via DAgger. Broader expert demonstrations do not eliminate the distribution gap between expert trajectories and states visited by the current policy. We therefore

Table 1: **Comparison with state-of-the-art methods** on the R2R-CE and RxR-CE *val-unseen* splits. † and * denote SimpleVLN before and after RL, respectively.

Method	Size	Observation				R2R-CE val-unseen				RxR-CE val-unseen			
		Pano.	Odo.	Depth	S.RGB	NE↓	OS↑	SR↑	SPL↑	NE↓	SR↑	SPL↑	nDTW↑
NaVid (Zhang et al., 2024b)	7B				✓	5.47	49.1	37.4	35.9	8.41	23.8	21.2	—
Uni-NaVid (Zhang et al., 2024a)	7B				✓	5.58	53.3	47.0	42.7	6.24	48.7	40.9	—
NaVILA (Cheng et al., 2024)	7B				✓	5.22	62.5	54.0	49.0	6.77	49.3	44.0	58.8
ActiveVLN (Zhang et al., 2025)	3B				✓	5.31	58.3	50.1	43.7	5.84	50.7	41.2	58.1
StreamVLN (Wei et al., 2025)	7B				✓	4.98	64.2	56.9	51.9	6.22	52.9	46.0	61.9
NavFoM (Zhang et al., 2026a)	7B				✓	5.01	64.9	56.2	51.2	5.51	57.4	49.4	60.2
JanusVLN (Zeng et al., 2026)	7B				✓	4.78	65.2	60.5	56.8	6.06	56.2	47.5	62.1
NaVIDA (Zhu et al., 2026)	3B				✓	4.32	69.5	61.4	54.7	5.23	57.4	49.6	67.0
CorrectNav (Yu et al., 2026b)	7B				✓	4.24	67.5	65.1	62.3	4.09	69.3	63.3	75.2
DualVLN (Wei et al., 2026)	7B				✓	4.05	70.7	64.3	58.5	4.58	61.4	51.8	70.0
FutureNav (Zhang et al., 2026c)	4B				✓	4.24	70.2	65.4	61.3	4.60	60.9	52.8	68.0
FutureNav (Zhang et al., 2026c)	8B				✓	4.29	69.2	64.3	60.3	4.26	63.9	54.8	69.1
AgentVLN (Xin et al., 2026)	3B			✓	✓	3.88	73.5	67.2	64.7	3.92	69.5	61.3	74.6
OmniNav (Xue et al., 2026)	3B	✓				3.74	74.6	69.5	66.1	3.77	73.6	62.0	—
SEDualVLN (Huang et al., 2026)	7B				✓	3.75	73.7	67.3	62.5	4.12	63.9	52.4	72.8
Qwen-RobotNav (Zhang et al., 2026b)	4B				✓	4.22	73.6	66.9	60.5	4.15	71.3	61.5	68.6
Qwen-RobotNav (Zhang et al., 2026b)	8B				✓	4.36	72.7	65.7	59.6	4.16	73.4	63.5	69.9
Qwen-RobotNav (Zhang et al., 2026b)	4B	✓				3.80	77.2	69.5	63.6	3.80	75.2	65.0	71.9
Qwen-RobotNav (Zhang et al., 2026b)	8B	✓				3.53	78.5	72.1	66.6	3.58	76.5	65.7	72.5
Efficient-VLN (Zheng et al., 2025)	8B				✓	3.29	78.2	73.2	65.0	3.18	75.6	60.0	-
SimpleVLN† (Qwen2.5-VL)	3B				✓	5.70	63.3	51.6	44.9	5.20	50.3	42.0	57.6
SimpleVLN† (Qwen2.5-VL)	7B				✓	4.87	68.4	59.1	52.0	5.31	58.9	49.2	62.9
SimpleVLN† (Qwen3-VL)	4B				✓	4.26	74.3	65.4	58.8	4.79	63.4	54.1	66.5
SimpleVLN* (Qwen3-VL)	4B				✓	3.39	79.4	73.8	66.7	3.29	75.1	65.3	71.5
SimpleVLN† (Qwen3-VL)	8B				✓	4.40	73.3	64.2	57.0	4.73	64.2	54.3	66.2
SimpleVLN* (Qwen3-VL)	8B				✓	2.93	79.2	74.4	69.5	3.11	76.4	66.4	77.4
SimpleVLN† (Qwen3.5)	4B				✓	3.76	77.5	67.1	59.2	4.32	65.6	55.3	67.8
SimpleVLN* (Qwen3.5)	4B				✓	2.79	80.1	76.3	72.3	2.61	79.2	69.7	80.3
SimpleVLN† (Qwen3.5)	9B				✓	3.59	76.9	69.2	61.2	4.48	64.0	54.2	67.0
SimpleVLN* (Qwen3.5)	**9B**				✓	**2.63**	**84.1**	**76.7**	**72.6**	**2.51**	**80.1**	**69.9**	**79.8**

use a deviation-based DAgger (Ross et al., 2011) strategy. We compare the current policy trajectory with the ground-truth (GT) trajectory: when the deviation is large, we select the expert trajectory; otherwise, we retain the policy trajectory. Selected trajectories are converted into decision-level supervised samples and aggregated with the demonstrations.

Task synthesis for RL. Because the R2R, RxR, and ScaleVLN training episodes are already used for supervised learning, we synthesize new RL tasks in their training scenes rather than reuse the annotated episodes, excluding val-unseen scenes. For each source, we sample start–goal pairs in different rooms from the navigation graph, with goals associated with semantically labelled objects. An MLLM generates navigation instructions from reference-path waypoints describing turns, rooms, salient objects, and egocentric views. The reference path supports task construction rather than providing an action target.

3.4 Progressive Optimization

Figure 3 summarizes the three-stage optimization of SimpleVLN. First, imitation learning uses expert demonstrations to learn navigation actions. Next, DAgger-based fine-tuning adapts the policy to states encountered during its own execution while retaining action-level supervision. Finally, reinforcement learning optimizes task-level outcomes through environment interaction on synthesized episodes, shifting the learning signal from demonstrated actions to outcome feedback.

Imitation learning from expert actions. Let $\mathcal{D}$ denote a supervised dataset of tuples $(\mathcal{I}, s_t, a_t^\star)$, where $a_t^\star$ denotes the target action chunk for decision state s_t. We minimize the negative log-likelihood of the complete target chunk: $\mathcal{L}_{\mathrm{SFT}} = -\mathbb{E}_{(\mathcal{I},s_t,a_t^\star)\sim\mathcal{D}}\left[\log \pi_\theta\big(a_t^\star \mid \mathcal{I}, s_t\big)\right]$. This objective teaches the policy to reproduce demonstrated action chunks, learning from expert supervision.

Policy-dependent supervised adaptation. We next train on the aggregation of demonstration and DAgger data from Section 3.3. The objective $\mathcal{L}_{\mathrm{SFT}}$ is unchanged, while $\mathcal{D}$ additionally includes target action chunks $a_t^\star$ derived from the selected policy-dependent trajectories.

Table 2: **Comparison with state-of-the-art methods on inference latency and navigation performance.** Latency is measured on R2R-CE using the same NVIDIA A100 GPU, and measures model execution only for one action-chunk decision, excluding preprocessing, environment execution, and other non-model overhead. SimpleVLN uses Qwen3-VL-4B as its backbone.

Method	Size	Latency (ms)↓	R2R-CE				RxR-CE			
			NE↓	OS↑	SR↑	SPL↑	NE↓	SR↑	SPL↑	nDTW↑
StreamVLN (Wei et al., 2025)	7B	300	4.98	64.2	56.9	51.9	6.22	52.9	46.0	61.9
JanusVLN (Zeng et al., 2026)	7B	906	4.78	65.2	60.5	56.8	6.06	56.2	47.5	62.1
DualVLN (Wei et al., 2026)	7B	683	4.05	70.7	64.3	58.5	4.58	61.4	51.8	70.0
FutureNav (Zhang et al., 2026c)	4B	1141	4.24	70.2	65.4	61.3	4.60	60.9	52.8	68.0
SimpleVLN	**4B**	**293**	**3.39**	**79.4**	**73.8**	**66.7**	**3.29**	**75.1**	**65.3**	**71.5**

Table 3: **Navigation interface and memory design.** Ablations of action chunking, single-turn navigation, and observation sampling using Qwen3-VL 4B fine-tuned on R2R. The default design of SimpleVLN is shaded.

Action format	NE↓	OS↑	SR↑
Primitive action	5.2	54.5	45.6
Action chunk	**5.5**	**61.2**	**53.6**

(a) **Action chunking.** *Primitive action* predicts one atomic action per decision; *action chunk* predicts multiple atomic actions per decision and executes them in order, stopping early if the episode terminates.

Dialogue format	NE↓	OS↑	SR↑
Multi-turn	5.9	55.2	35.7
Single-turn	**5.5**	**61.2**	**53.6**

(b) **Single-turn navigation.** *Multi-turn* accumulates observations and actions in the dialogue; *single-turn* reconstructs each input from the instruction, current observation, and sampled fixed-budget history.

Sampling scheme	NE↓	OS↑	SR↑
Chunk	6.1	55.2	46.3
Primitive (all)	5.8	57.5	51.9
Primitive (8)	**5.5**	**61.2**	**53.6**

(c) **Observation sampling.** *Chunk* samples boundaries; *Primitive (all)* keeps all primitive-step observations; *Primitive (8)* uniformly samples eight global history frames, excluding local and current frames.

Outcome optimization through reinforcement learning. Because navigation is ultimately evaluated by task completion rather than agreement with a particular action sequence, we optimize task-level outcomes on the synthesized navigation tasks. Using the synthesized episodes described in Section 3.3, the policy interacts with the environment and sequentially predicts decision-level action chunks,

$$a_t \sim \pi_\theta\big(\cdot \mid \mathcal{I}, s_t\big), \qquad \tau = \big((s_t, a_t)\big)_{t=1}^{T}. \tag{2}$$

Here, T is the actual number of model-level decisions before the episode terminates, distinct from the primitive-action budget $S_{\max}$. We use a binary task-level reward to make task completion the explicit objective, without requiring agreement with a reference action sequence. Success follows the stopping and distance criteria in Section 3.1:$R(\tau) = \mathbb{I}(\text{success}) \in \{0, 1\}$. We maximize the expected task reward over the trajectory distribution $p_\theta(\tau)$ induced by policy–environment interaction on synthesized episodes:$\max_\theta \ \mathcal{J}_{\mathrm{RL}}(\theta) = \mathbb{E}_{\tau \sim p_\theta}\left[R(\tau)\right]$. Unlike imitation learning, this objective does not require agreement with a particular reference action sequence: any trajectory that successfully completes the task is rewarded. Training therefore progresses from expert action supervision, through policy-dependent supervised adaptation, to episode-level task-completion feedback.

4 Experiments

4.1 Experimental Setup

Benchmarks and metrics. We evaluate SimpleVLN on the R2R-CE (Anderson et al., 2018) and RxR-CE (Ku et al., 2020) val-unseen splits in Matterport3D with Habitat (Krantz et al., 2020), reporting NE, OS, SR, SPL, and nDTW, with SR as the primary metric.

Implementation details. We use Qwen3-VL (4B/8B) (Bai et al., 2025) and Qwen3.5 (4B/9B) (Team et al., 2026). IL uses SFT for 3 epochs on R2R (Anderson et al., 2018) and RxR (Ku et al., 2020), with the vision encoder frozen and a peak learning rate of 2×10^{-5}. We then train 1 epoch on collected DAgger data with R2R/RxR, followed by 2 epochs on their mixture with ScaleVLN (Wang et al., 2023). Main-result RL uses DAPO (Yu et al., 2026a) for 40 steps with batch size 256, 16 rollouts per prompt, learning rate 1×10^{-6}, KL coefficient 0, and clip ratios $0.30/0.20$.

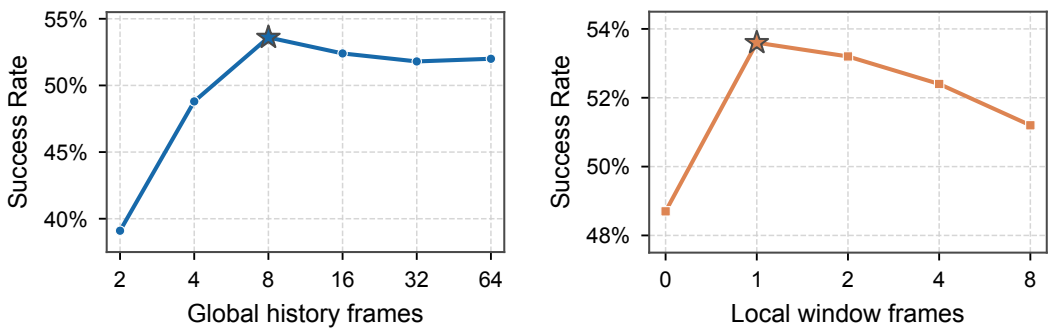


Figure 4: **Memory budget.** Effect of global and local memory sizes on R2R-CE *val-unseen* SR. We vary M with $N = 1$ (left) and N with $M = 8$ (right); stars indicate the default configuration.

4.2 Comparison with State-of-the-Art

Main results. Table 1 compares SimpleVLN with prior methods and reports results across multiple backbone families and model sizes, including pre-RL Qwen2.5-VL (3B/7B) variants. SimpleVLN reaches 76.7 SR on R2R-CE and 80.1 SR on RxR-CE using only single-RGB observations and the native MLLM architecture, outperforming existing methods. RL consistently improves SR across evaluated backbones; for Qwen3-VL 4B, SR increases from 65.4 to 73.8 on R2R-CE and from 63.4 to 75.1 on RxR-CE. Table 2 shows that SimpleVLN requires 293 ms per action-chunk decision, comparable to StreamVLN and faster than the other compared methods.

4.3 Formulation

We investigate the navigation formulation introduced in Section 3.2. Unless otherwise specified, these experiments use Qwen3-VL 4B fine-tuned on R2R and evaluated on the R2R-CE *val-unseen* split.

Action chunking. Table 3a evaluates action chunking as an alternative to primitive-action prediction. Action chunking raises SR from 45.6 to 53.6, an 8.0-point gain, supporting it as an effective output representation beyond reducing model invocation frequency.

Single-turn decision input. Table 3b replaces accumulated multi-turn dialogue with a reconstructed single-turn input and sampled visual history, raising SR from 35.7 to 53.6. Although multi-turn dialogue can retain a fuller interaction history, continually appending observations and actions creates an overly long context that current MLLMs cannot effectively exploit for navigation.

Observation sampling. Table 3c examines which observations to retain. Sampling only chunk boundaries yields 46.3 SR, retaining all primitive-step observations yields 51.9, and uniformly sampling eight global history frames yields 53.6. The eight-frame budget covers global history only, excluding local history and the current observation. These results favor compact, uniformly sampled global history over retaining every frame.

Memory budget. Figure 4 tests whether increasing the global or local history budget improves navigation accuracy. With $N = 1$, SR peaks at $M = 8$ and then declines as the global budget grows, suggesting that excessive global history introduces redundant or less relevant observations that dilute useful context. With $M = 8$, retaining one local observation gives the highest SR, indicating that additional nearby frames provide largely redundant information. We therefore use $M = 8$ and $N = 1$: up to eight global and one local history frame, plus the current observation.

4.4 Data Construction

We investigate the effects of the training data constructed in Section 3.3. All experiments in this subsection use Qwen3-VL 4B and report results on the R2R-CE and RxR-CE *val-unseen* splits.

Expert demonstration coverage. Table 4 tests whether broadening expert demonstration coverage improves navigation. Adding RxR to R2R raises SR from 53.6 to 56.7 on R2R-CE and from 50.3 to 54.5 on RxR-CE. Including ScaleVLN further increases SR to 57.9 and 55.3, respectively. These results show that combining R2R and RxR provides clear in-domain gains, while ScaleVLN further improves generalization to unseen environments.

Table 4: **Demonstration and DAgger data ablation** on the R2R-CE and RxR-CE *val-unseen* splits.

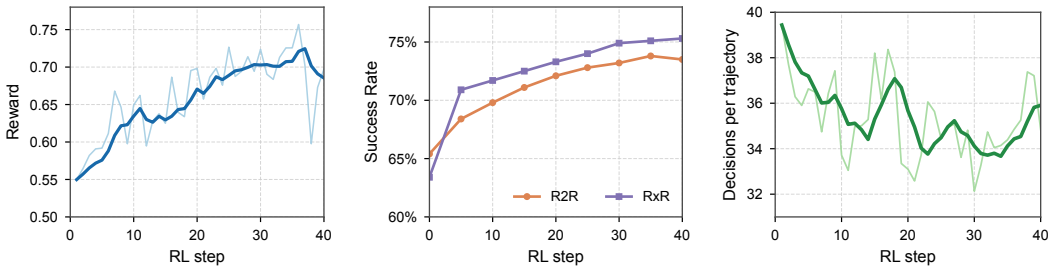
R2R	RxR	ScaleVLN	DAgger	R2R-CE				RxR-CE			
				NE↓	OS↑	SR↑	SPL↑	NE↓	SR↑	SPL↑	nDTW↑
✓				5.53	61.2	53.6	48.1	6.45	50.3	42.0	57.6
✓	✓			5.28	63.4	56.7	51.3	5.75	54.5	46.7	62.4
✓	✓	✓		4.90	68.4	57.9	50.9	5.27	55.3	48.1	63.6
✓	✓	✓	✓	**4.26**	**74.3**	**65.4**	**58.8**	**4.79**	**63.4**	**54.1**	**66.5**

Table 5: **Synthesized RL data ablation** on the R2R-CE and RxR-CE *val-unseen* splits. Dataset columns identify the training-scene sources used to synthesize RL episodes.

R2R	RxR	ScaleVLN	R2R-CE				RxR-CE			
			NE↓	OS↑	SR↑	SPL↑	NE↓	SR↑	SPL↑	nDTW↑
Pre-RL baseline			4.26	74.3	65.4	58.8	4.79	63.4	54.1	66.5
✓			3.91	74.0	68.5	63.2	4.31	65.8	55.3	68.3
	✓		3.76	71.8	67.5	59.3	3.98	68.5	59.5	68.8
		✓	4.17	70.9	66.6	61.3	4.34	66.3	56.2	67.3
✓	✓		3.75	74.1	69.4	63.8	3.77	69.7	60.8	68.1
✓	✓	✓	**3.74**	**73.1**	**70.1**	**65.6**	**3.82**	**71.0**	**62.1**	**68.4**

Policy-dependent experience via DAgger. We next assess whether policy-dependent data improves over the combined expert demonstrations. Adding DAgger yields the largest gains in Table 4: 7.5 SR points on R2R-CE and 8.1 on RxR-CE, reaching 65.4 and 63.4. This shows the value of complementing static demonstrations with trajectories collected by the learned policy.

Task resynthesis for RL. Table 5 evaluates RL tasks synthesized from the R2R, RxR, and ScaleVLN training scenes. These names denote scene pools for synthesis, not reused annotated episodes. All variants start from the same IL–DAgger checkpoint and use GRPO (Shao et al., 2024) with batch size 32 and eight rollouts per prompt. Each pool improves SR over the pre-RL baseline, while their mixture performs best, reaching 70.1 on R2R-CE and 71.0 on RxR-CE.



(a) Rollout reward. (b) Success rate on R2R and RxR. (c) Decisions per trajectory.

Figure 5: **RL training dynamics.** Overall, rollout reward and evaluation success rate increase, while the average number of action-chunk decisions per trajectory decreases.

4.5 Optimization

We investigate how RL configurations affect navigation accuracy and how the model's navigation behavior evolves during training. Table 6 uses Qwen3-VL 4B and a mixture of synthetic episodes constructed from the R2R, RxR, and ScaleVLN training scenes.

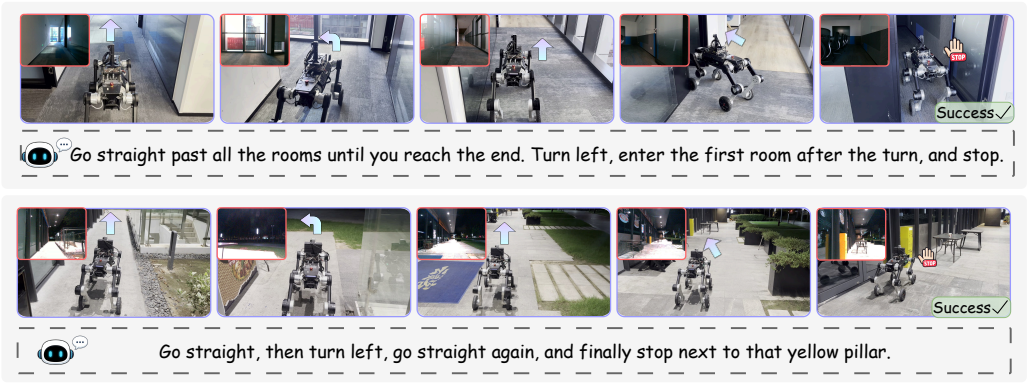
Reward design. Under GRPO with batch size 32 and eight rollouts per prompt, we compare SPL and binary rewards. Both assign zero to failures; successful episodes receive either the SPL score or a reward of one. The binary reward raises SR from 68.9 to 70.1 on R2R-CE and from 67.7 to 71.0 on RxR-CE, supporting simple task-completion feedback.

RL configuration. With the success reward fixed, DAPO (Yu et al., 2026a) reaches 71.5/73.7 SR, compared with 70.1/71.0 for GRPO. Both ablations use a batch size of 32, compared with 256 for the main-results configuration in Section 4.1. These results favor the evaluated DAPO configuration, although we do not attribute the gain solely to the optimization algorithm.

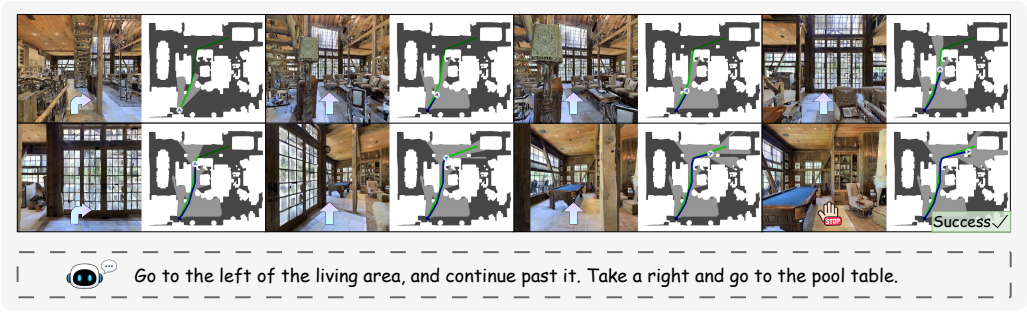
Table 6: **Reward design and RL optimization** on the R2R-CE and RxR-CE *val-unseen* splits with Qwen3-VL 4B. The SPL reward is zero for failed episodes and equals the ratio of the ground-truth path length to the executed path length for successful episodes; the success reward is binary.

RL configuration	R2R-CE				RxR-CE			
	NE↓	OS↑	SR↑	SPL↑	NE↓	SR↑	SPL↑	nDTW↑
Pre-RL baseline	4.26	74.3	65.4	58.8	4.79	63.4	54.1	66.5
GRPO + $\mathrm{SPL}(\tau)$	3.80	73.9	68.9	63.1	4.13	67.7	58.3	67.3
GRPO + $\mathbb{I}(\mathrm{success})$	3.74	73.1	70.1	65.6	3.82	71.0	62.1	68.4
DAPO + $\mathbb{I}(\mathrm{success})$	**3.45**	**76.6**	**71.5**	**65.3**	**3.39**	**73.7**	**64.0**	**69.4**

RL training dynamics. Figure 5 shows that, despite fluctuations, rollout reward and evaluation SR trend upward, while the average number of action-chunk decisions per trajectory gradually decreases. This suggests that RL improves navigation performance while also increasing decision efficiency.



(a) Qualitative results of SimpleVLN in the real world.



(b) Qualitative results of SimpleVLN on Simulation.

Figure 6: **SimpleVLN in real-world and simulated environments.** SimpleVLN follows diverse instructions and reaches the specified goals; arrows indicate navigation directions.

4.6 Real World Experiments

We further evaluate SimpleVLN in real environments using a Daystar quadruped robot. The model runs onboard on an NVIDIA Jetson Thor GPU. The robot is equipped with an Intel RealSense D450 RGB-D camera for egocentric perception. Figure 6 shows that SimpleVLN successfully follows multi-step instructions involving forward movement, turns, room entry, and landmark-based stopping, while maintaining coherent instruction-following behavior over extended trajectories.

5 Conclusion

We presented SimpleVLN, a framework for adapting native MLLMs to VLN. SimpleVLN combines single-turn decision inputs, global–local memory, and action chunking with training data constructed from expert demonstrations, policy-dependent DAgger experience, and synthesized RL tasks. A progressive IL–DAgger–RL pipeline further shifts learning from action imitation to task-level outcome optimization. SimpleVLN achieves strong navigation performance, demonstrating that native MLLM navigation capabilities can be effectively unlocked without task-specific architectural designs.

AI USE STATEMENT

Generative AI was used in two capacities. First, as part of the research method, an MLLM generated the instructions for the synthetic reinforcement-learning episodes described in Section 3.3. Second, a generative AI assistant was used to improve the wording and readability of the manuscript, including drafting and copy-editing text. The authors reviewed and revised all AI-assisted text, verified the associated technical claims and citations, and independently checked the reported experimental results. Generative AI was not treated as an author and did not determine the conclusions of the paper. The authors take full responsibility for the final manuscript and all artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work conducts visual language navigation experiments and does not involve new human-subject studies or the collection of personal data. We use existing datasets in accordance with their licenses and terms of use. Our quantitative benchmark evaluation is conducted in simulation, while we additionally provide qualitative real-world robot demonstrations. Real-world deployment of navigation systems may raise safety and privacy concerns and should therefore include appropriate safeguards and human oversight.

REPRODUCIBILITY STATEMENT

We describe the navigation formulation, memory construction, action space, data construction, and optimization objectives in Section 3.1– Section 3.4. Section 4.1 specifies the evaluated benchmarks and metrics, model backbones, training schedule, learning rates, batch sizes, rollout counts, and reinforcement-learning hyperparameters; the remaining experimental subsections report controlled formulation, data, reward, and optimizer ablations. To facilitate reproduction, we will release the training and evaluation code, configuration files, data-generation prompts, synthetic-task metadata, and model checkpoints. Resources derived from R2R, RxR, ScaleVLN, and Matterport3D will be distributed only as permitted by their licenses; where redistribution is restricted, we will provide scripts and instructions for authorized users to reconstruct the required data.

REFERENCES

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 *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 3674–3683. IEEE, 2018.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

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 *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16516–16526. IEEE, 2022.

An-Chieh Cheng, Yandong Ji, Zhaojing Yang, Zaitian Gongye, Xueyan Zou, Jan Kautz, Erdem Bıyık, Hongxu Yin, Sifei Liu, and Xiaolong Wang. Navila: Legged robot vision-language-action model for navigation. *arXiv preprint arXiv:2412.04453*, 2024.

Yue Hu, Avery Xi, Qixin Xiao, Seth Isaacson, Henry X Liu, Ram Vasudevan, and Maani Ghaffari. Longnav-r1: Horizon-adaptive multi-turn rl for long-horizon vla navigation. *arXiv preprint arXiv:2602.12351*, 2026.

Jingzhi Huang, Junkai Huang, Wenxuan Song, Haoyang Yang, Hailong Huang, Haoang Li, and Yi Wang. Sedualvln: A spatially-enhanced dual-system for vision-language navigation. *arXiv preprint arXiv:2605.17249*, 2026.

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.

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.

Bingqian Lin, Yunshuang Nie, Ziming Wei, Jiaqi Chen, Shikui Ma, Jianhua Han, Hang Xu, Xiaojun Chang, and Xiaodan Liang. Navcot: Boosting llm-based vision-and-language navigation via learning disentangled reasoning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(7):5945–5957, 2025.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y.K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

InternNav Team. InternVLA-N1: An open dual-system navigation foundation model with learned latent plans, 2025.

Qwen Team et al. Qwen3.5: Towards native multimodal agents, february 2026. *URL https://qwen.ai/blog*, 2026.

Shuo Wang, Yongcai Wang, Wanting Li, Xudong Cai, Yucheng Wang, Maiyue Chen, Zhizhong Su, Deying Li, and Zhaoxin Fan. Aux-think: Exploring reasoning strategies for data-efficient vision-language navigation. *Advances in Neural Information Processing Systems*, 38:30892–30916, 2026.

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 *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11975–11986. IEEE, 2023.

Meng Wei, Chenyang Wan, Xiqian Yu, Tai Wang, Yuqiang Yang, Xiaohan Mao, Chenming Zhu, Wenzhe Cai, Hanqing Wang, Yilun Chen, et al. Streamvln: Streaming vision-and-language navigation via slowfast context modeling. *arXiv preprint arXiv:2507.05240*, 2025.

Meng Wei, Chenyang Wan, Peng Peng, Xiqian Yu, Yuqiang Yang, Delin Feng, Wenzhe Cai, Chenming Zhu, Tai Wang, Jiangmiao Pang, et al. Ground slow, move fast: A dual-system foundation model for generalizable vision-language navigation. In *International Conference on Learning Representations*, volume 2026, pp. 12380–12396, 2026.

Zihao Xin, Wentong Li, Yixuan Jiang, Ziyuan Huang, Bin Wang, Piji Li, Jianke Zhu, Jie Qin, and Shengjun Huang. Agentvln: Towards agentic vision-and-language navigation. In *European Conference on Computer Vision*, pp. 114–132. Springer, 2026.

Xinda Xue, Junjun Hu, Minghua Luo, Shichao Xie, Jintao Chen, Zixun Xie, Quan Kuichen, Wei Guo, Zedong Chu, Mu Xu, et al. Omninav: A unified framework for prospective exploration and visual-language navigation. 2026:106216–106231, 2026.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *Advances in Neural Information Processing Systems*, 38:113222–113244, 2026a.

Zhuoyuan Yu, Yuxing Long, Zihan Yang, Chengyan Zeng, Hongwei Fan, Jiyao Zhang, and Hao Dong. Correctnav: Self-correction flywheel empowers vision-language-action navigation model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18737–18745, 2026b.

Shuang Zeng, Dekang Qi, Xinyuan Chang, Feng Xiong, Shichao Xie, Xiaolong Wu, Shiyi Liang, Mu Xu, and Xing Wei. Janusvln: Decoupling semantics and spatiality with dual implicit memory for vision-language navigation. In *International Conference on Learning Representations*, volume 2026, pp. 33001–33026, 2026.

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*, 2024a.

Jiazhao Zhang, Kunyu Wang, Rongtao Xu, Gengze Zhou, Yicong Hong, Xiaomeng Fang, Qi Wu, Zhizheng Zhang, and He Wang. Navid: Video-based vlm plans the next step for vision-and-language navigation. *arXiv preprint arXiv:2402.15852*, 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. In *International Conference on Learning Representations*, volume 2026, pp. 127293–127322, 2026a.

Jiazhao Zhang, Gengze Zhou, Hale Yin, Yiyang Huang, Zixing Lei, Qihang Peng, Haoqi Yuan, Jie Zhang, Xudong Guo, Xiaoyue Chen, et al. Qwen-robotnav technical report: A scalable navigation model designed for an agentic navigation system. *arXiv preprint arXiv:2606.18112*, 2026b.

Lingfeng Zhang, Zeying Gong, Xiaoshuai Hao, Haoxiang Fu, Qiang Zhang, Mingliang Zhou, Hangjun Ye, Xiaojun Liang, Junwei Liang, and Wenbo Ding. Futurenav: Unified world-action modeling for vision-and-language navigation. *arXiv preprint arXiv:2606.30367*, 2026c.

Zekai Zhang, Weiye Zhu, Hewei Pan, Xiangchen Wang, Rongtao Xu, Xing Sun, and Feng Zheng. Activevln: Towards active exploration via multi-turn rl in vision-and-language navigation. *arXiv preprint arXiv:2509.12618*, 2025.

Duo Zheng, Shijia Huang, Yanyang Li, and Liwei Wang. Efficient-vln: A simple yet strong baseline for efficient vision-language navigation. *arXiv preprint arXiv:2512.10310*, 2025.

Weiye Zhu, Zekai Zhang, Xiangchen Wang, Hewei Pan, Teng Wang, Tiantian Geng, Rongtao Xu, and Feng Zheng. Navida: Vision-language navigation with inverse dynamics augmentation. *arXiv preprint arXiv:2601.18188*, 2026.