CITYLOOP: SANDBOX FOR TOOL-GROUNDED URBAN NAVIGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Urban navigation requires an embodied agent to connect a user request with global navigation guidance, local visual perception, continuous action, and eventual arrival. Existing research, however, typically studies these components in isolation, making it difficult to evaluate how they work together over complete routes. We introduce CityLoop, a unified request-to-arrival sandbox for tool-grounded urban navigation. CityLoop connects online navigation, 2D Gaussian Splatting (2DGS) simulation, and physical robots through a shared policy interface, while providing controllable localization perturbations and geographically aligned supervision from map and street-view data. Systematic evaluation in CityLoop reveals three limitations of existing models: unreliable execution of navigation instructions even under clean conditions, weak coordination between navigation guidance and egocentric visual evidence, especially when the two disagree, and compounding errors over long-horizon closed-loop execution. To address these limitations, we introduce RouteGround, a capability-structured training framework that develops navigation-context understanding, visual grounding, conflict-aware decision making through counterfactual tool–vision conflicts, and full-route execution. RouteGround improves keypoint decision accuracy over the strongest baseline from 47.6% to 78.4% with accurate localization and from 26.0% to 59.0% under medium localization errors, achieves 46.0% and 32.0% success in complete 2DGS rollouts under clean and medium-error guidance, respectively, and succeeds in 6 of 20 preliminary physical trials after zero-shot transfer.

1 INTRODUCTION

Embodied agents operating in real cities must navigate over spatial scales far beyond their instantaneous perceptual field. A first-person camera provides local road geometry and obstacle information, but cannot directly reveal distant destinations or the full structure of a city-scale route. In practice, long-range navigation therefore benefits from combining local visual perception with external navigation services that provide destination and route information. Yet successful urban navigation involves considerably more than either route planning or local control alone: an agent must interpret a user request, identify an appropriate destination, obtain and follow a route, continuously relate navigation guidance to its current observations, and determine when it has arrived. This makes urban navigation a naturally request-to-arrival, closed-loop system problem.

Prior work has made substantial progress on individual parts of this problem. Street View environments support goal reaching and language-guided navigation over city-scale panorama graphs (Mirowski et al., 2018; Schumann et al., 2024); web-scale videos and human trajectories teach local urban behavior (Liu et al., 2025; Mei et al., 2026); geographic hints and public routes condition long-range and micromobility control (Shah & Levine, 2022; Li et al., 2025); task-specific methods target the last meters of a route or long-horizon social assistance (Zhao et al., 2026; Zhang et al., 2026); vision-language-action models bring navigation policies to physical robots (Cheng et al., 2025); and simulators and benchmarks scale urban environments or evaluate navigation behavior (Liu et al., 2026; Windecker et al., 2025). Together, these efforts have established key building blocks for urban navigation, including realistic environments, scalable data sources, capable policies, and physical platforms. Yet they are developed under different task formulations, interfaces, and evaluation protocols, and the few systems that span the full loop are task-specific pipelines vali-

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.

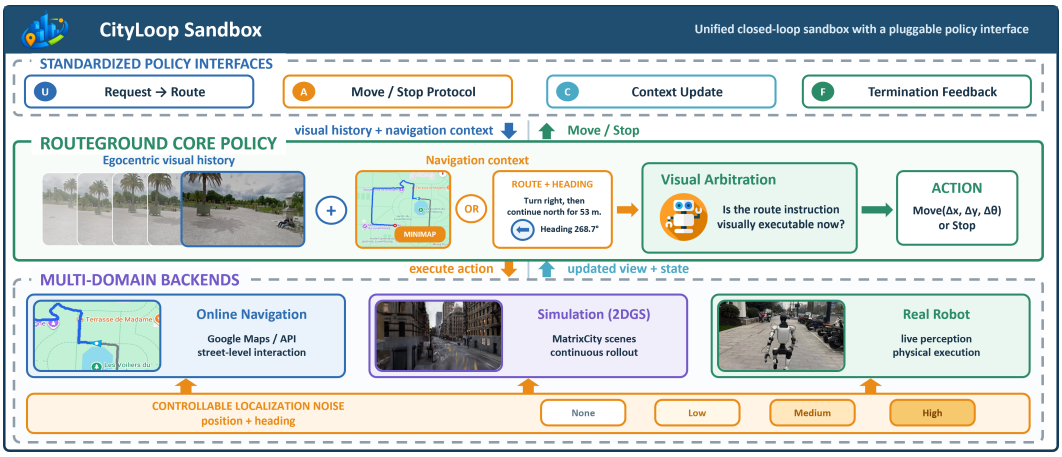

Figure 1: CityLoop sandbox and RouteGround interaction. CityLoop unifies request-to-route initialization, move/stop execution, context updates, and termination feedback, and executes RouteGround's actions online, in 2DGS simulation, or on a real robot, with optional pose perturbation.

dated through limited trials (Zhang et al., 2026; Wang et al., 2025). As a result, it remains difficult to systematically study how request resolution, global navigation guidance, local perception, sequential decision making, and physical execution interact within a single complete navigation loop.

To enable such study, we introduce CityLoop, a unified sandbox for request-to-arrival urban navigation (Figure 1). It resolves explicit or intent-based requests into destinations and routes, then runs a policy through one interface on online navigation platforms, 2DGS simulation, or a physical robot, refreshing the navigation context after every action until the agent stops. Because routes, street-level panoramas, and destinations share one geographic frame, CityLoop also converts navigation states into verifiable supervision drawn from map and street-view data rather than videos, and benchmarks each stage from POI resolution to closed-loop and physical execution. Finally, it can perturb the reported position and heading while leaving the physical state unchanged, enabling matched evaluation under localization errors.

Evaluating existing multimodal and navigation-specialized models within this unified setting reveals three recurring limitations. First, reliably following navigation instructions remains difficult even with accurate localization: models often fail to turn navigation context into the correct action at decision-critical states. Second, they struggle to use navigation guidance and egocentric evidence jointly, especially when localization errors make the two disagree; in Figure 2, a globally valid turn cue arrives before the turning point. Third, errors that look local compound over complete routes by changing subsequent observations. Successful urban navigation thus requires understanding global guidance, grounding it in the current scene, arbitrating between inconsistent sources, and composing reliable decisions over time.

Motivated by these findings, we introduce RouteGround, a capability-structured training framework for tool-grounded urban navigation. It first learns to interpret navigation context and route structure, then grounds decisions at visually critical states such as intersections and arrivals. Its *Counterfactual Conflict Training* perturbs the pose behind the navigation context while keeping the oracle action tied to the true physical state, so the policy must judge whether the suggested maneuver is executable instead of copying the tool. Finally, complete closed-loop trajectories teach the policy to compose these capabilities over long routes and recover from earlier decisions.

To determine whether RouteGround improves not only isolated decisions but also end-to-end navigation, we evaluate it from decision-critical states to 2D Gaussian Splatting (2DGS) (Huang et al., 2024) closed-loop rollouts and physical robot deployment. With accurate localization, RouteGround raises keypoint Decision Accuracy from 47.6% for the strongest baseline to 78.4%; under medium localization errors, it reaches 59.0%, whereas no baseline exceeds 26.0%. In complete 2DGS rollouts, where the best baseline completes at most 4.0% of routes, RouteGround achieves Success Accuracy of 46.0% and 32.0% under clean and medium-error guidance, and it transfers zero-shot to a physical robot, succeeding in 6 of 20 preliminary trials. Together, these results suggest that a uni-

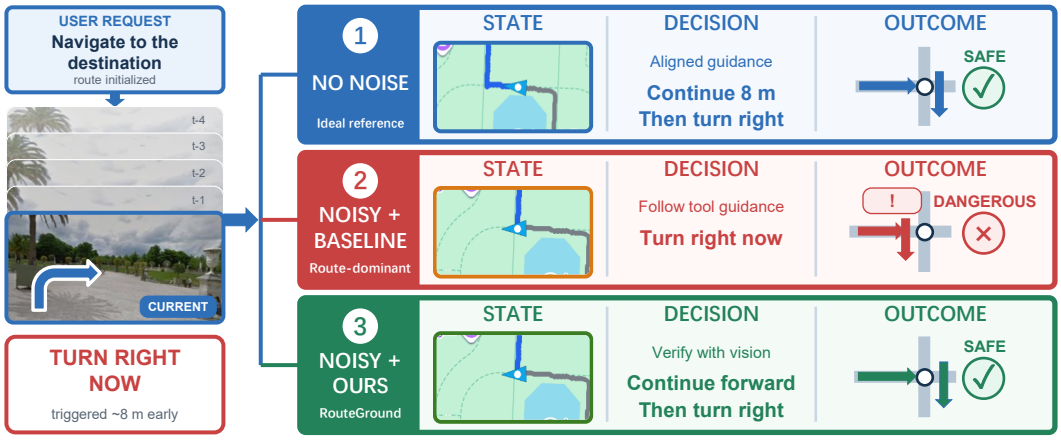

Figure 2: Turn execution under GPS offset. With accurate GPS, the agent reaches the turning point before turning. When GPS offset triggers the cue early, a route-dominant baseline turns prematurely, whereas RouteGround uses egocentric history to continue safely to the correct turn.

fied request-to-arrival framework exposes limitations that isolated subtasks hide, while capability-structured training substantially improves complete urban navigation.

Our contributions are twofold:

- **CityLoop:** a unified request-to-arrival platform that connects online navigation, 2DGS simulation, and physical robots through one policy interface, converts map and street-view data into verifiable supervision, and benchmarks each stage from POI resolution to closed-loop and physical execution, including under controlled localization errors.
- **RouteGround:** a capability-structured training framework that develops navigation-context understanding, visual grounding, conflict-aware decision making through counterfactual conflicts, and long-horizon closed-loop execution, improving keypoint decisions, closed-loop success, and zero-shot robot deployment.

2 Related Work

Table 1 situates CityLoop and RouteGround among representative urban-navigation policies, benchmarks, and environments. We review the two closest lines of work: urban policy learning and closed-loop navigation environments.

Urban Visual Navigation and Route-Conditioned Control. StreetLearn introduced city-scale goal-conditioned interaction over Street View (Mirowski et al., 2018), while Touchdown, Retouchdown, and VELMA grounded route language in urban panoramas (Chen et al., 2019; Mehta et al., 2020; Schumann et al., 2024). Recent policies scale supervision from web video and human trajectories (Liu et al., 2025; Mei et al., 2026), condition control on public routes (Li et al., 2025), or study prior-free instruction following over the last meters of a route (Zhao et al., 2026). SocialNav and ABot-N0 broaden this line toward socially aware and multi-goal navigation (Chen et al., 2026; Chu et al., 2026), while S2E adds interactive reinforcement learning after video pretraining (He et al., 2026). NaviTrace instead evaluates image-space navigation traces without closed-loop execution (Windecker et al., 2025), while NavTrust benchmarks robustness to perturbations in visual observations and instructions (Jiang et al., 2026). These works expand supervision, task scope, physical deployment, and robustness evaluation, but do not make localization-induced disagreement between public-route guidance and egocentric evidence a controlled object of both evaluation and learning.

Closed-Loop Navigation Environments. Gibson and Habitat provide repeatable embodied evaluation in photorealistic simulation (Xia et al., 2018; Savva et al., 2019); StreetLearn and Touchdown support interaction over real city-scale panorama graphs (Mirowski et al., 2018; Chen et al., 2019). MetaUrban and URBAN-SIM provide scalable procedurally constructed urban simulation (Wu et al., 2025a;b), whereas Vid2Sim and UrbanVerse convert videos into interactive training en-

Table 1: Representative urban-navigation paradigms. POI search: resolving a place or intent request into a destination. GPS/map robustness: trained or evaluated under localization or route errors. Long-route E2E: complete long-horizon execution. Navigation map: a map given to the policy.

Work	POI search	GPS/map robustness	Long-route E2E	Navigation map	Language instruction
StreetLearn (Mirowski et al., 2018)	–	–	✓	–	–
ViKiNG (Shah & Levine, 2022)	–	✓	✓	✓	–
VELMA (Schumann et al., 2024)	–	–	✓	–	✓
CityWalker (Liu et al., 2025)	–	–	–	–	–
UrbanNav (Mei et al., 2026)	–	–	–	–	✓
UrbanVLA (Li et al., 2025)	–	✓	✓	–	✓
Walk With Me (Zhang et al., 2026)	✓	–	✓	–	✓
BridgeNav (Zhao et al., 2026)	✓	–	–	–	✓
S2E (He et al., 2026)	–	–	–	–	–
NaviTrace (Windecker et al., 2025)	–	–	–	–	✓
UrbanVerse (Liu et al., 2026)	–	–	✓	–	–
CityLoop + RouteGround	✓	✓	✓	✓	✓

vironments (Xie et al., 2025; Liu et al., 2026). These platforms improve scale, realism, or sim-to-real transfer. However, they do not provide a unified way to study public route navigation across online navigation, simulation, and physical deployment under localization errors. To address this gap, CityLoop connects the three domains through a shared interface and uses controllable position and heading perturbations to create mismatches between route guidance and visual observations.

3 Task Formulation and the CityLoop Environment

We first formalize tool-grounded urban navigation from request to arrival, and then present CityLoop and the navigation system built upon it. Figure 1 shows how a unified policy interface connects RouteGround's decisions to online navigation, 2DGS simulation, and real-world deployment, with optional perturbation of the reported position and heading.

3.1 Tool-Grounded Urban Navigation

To make the above problem precise, we define tool-grounded urban navigation from request initialization to final arrival. Given a user request q and an initial pose p_0, the system resolves a destination $\hat{g}$ and obtains a global route r from a navigation tool. Destination resolution is an upstream initialization step: the closed-loop route-execution phase begins only when $\hat{g}$ satisfies the request. Let $\mathcal{G}(q)$ denote the set of valid destinations for q, and let $d_T = d(p_T, \hat{g})$; complete task success is

$$\mathrm{Success}_{\mathrm{task}} = \begin{cases} 0, & \hat{g} \notin \mathcal{G}(q), \\ S_{\mathrm{nav}}, & \hat{g} \in \mathcal{G}(q), \end{cases} \tag{1}$$
$$S_{\mathrm{nav}} = \mathbb{1}[d_T \le \tau_{\mathrm{arrive}}]\,\mathbb{1}[a_T = \mathrm{Stop}],$$

Thus, a valid destination is a prerequisite for navigation, and the episode succeeds only if the agent actively outputs Stop within the arrival threshold τ_{arrive}, set to 5 m in all experiments; the policy interface used during route execution is defined next.

Once the destination and route are initialized, navigation is a partially observable sequential decision process with a shared policy interface. At step t, the latent state s_t and policy input X_t are

$$s_t = (p_t, \hat{g}, r, \rho_t, e_t), \tag{2}$$
$$X_t = \left(I, V_{t-K+1:t}, C_t, \bar{A}_{t-K+1:t-1}\right), \tag{3}$$

where p_t is the current pose, ρ_t is route progress, and e_t is the backend state; these variables are not exposed directly to the policy. In X_t, I is the task instruction, $V_{t-K+1:t}$ is a bounded egocentric visual history, C_t is the current navigation context, and $\bar{A}_{t-K+1:t-1}$ records the corresponding executed-action history; the window length is $K = 5$ in all experiments. The navigation context has two supported forms:

$$C_t = \begin{cases} M_t, & \text{minimap mode}, \\ (H_t, R_t), & \text{heading–route mode}, \end{cases} \tag{4}$$

where M_t is a local minimap, H_t is the current heading, and R_t describes the remaining route. RouteGround outputs only a high-level move or stop action,

$$a_t \in \left\{ \text{Move}\left(x_t^{\text{right}}, y_t^{\text{forward}}, \Delta\theta_t\right), \text{Stop} \right\}. \tag{5}$$

This interface specifies the information consumed and produced by the policy; the next subsection defines how CityLoop preserves this interface across deployment domains.

3.2 CityLoop: A Unified Closed-Loop Environment

To support deployment across heterogeneous urban backends, CityLoop applies the same closed-loop navigation task to online navigation platforms, simulation platforms, and real-world deployment. Their shared interaction is

$$(s_{t+1}, V_{t+1}, C_{t+1}, f_t) = \text{Step}_d(s_t, \bar{a}_t), \quad d \in \{\text{Online}, \text{Simulation}, \text{Real}\}. \tag{6}$$

where $\bar{a}_t$ is the executed action and f_t is task feedback. All three domains share destination and route semantics, visual history, the two navigation-context modes, the move/stop protocol, and termination feedback, so the policy interacts with each domain through the same decision interface. Online navigation platforms provide maps, routes, and panoramic observations aligned with real streets; simulation platforms provide controllable rendering and state transitions; and the real-world interface receives live perception and localization from a physical platform. This policy-level contract leaves backend dynamics domain-specific while requiring every domain to regenerate the next navigation context from its latest position and heading.

Real-world localization is updated online and can be inaccurate. Accordingly, after every action, CityLoop refreshes the navigation context from the latest backend state and can perturb its reported position and heading at four severity levels, while leaving the physical trajectory and visual observations unchanged. This isolates the effect of localization errors across backends and provides the environment on which the complete navigation system is built.

3.3 Request-to-Arrival Navigation System

With the CityLoop environment established, we build a request-to-arrival navigation system to complete the task through its unified interaction interface. Given a user request, the system queries online navigation APIs to filter candidate POIs, resolve the destination, and obtain the corresponding global route and initial navigation context. During route execution, it combines the current and four preceding egocentric views with the latest navigation context and executed-action history, and RouteGround predicts a move or stop action. The selected action is issued through the CityLoop interface, which advances the active backend and returns the next egocentric observation, updated navigation context, and task feedback. Repeating this loop connects API-based request handling with visually grounded navigation until the agent stops at the destination.

4 RouteGround: Capability-Structured Counterfactual Training

CityLoop exposes verifiable closed-loop states, but the central learning problem is to teach a policy when tool guidance is visually executable. We address this problem with RouteGround, organizing its supervision into four complementary capability groups. Figure 3 makes the training signal explicit by pairing a representative data card in each group with its policy input and target output.

4.1 CityLoop-Grounded Data Construction

CityLoop's value for learning extends beyond reproducing navigation: it turns decision evidence that is difficult to annotate at scale in physical deployment into explicit supervision. At each closed-loop state, it aligns the bounded egocentric history and navigation context with route-relative position, progress, and an oracle move/stop action. Because panoramic viewpoints, route geometry, and destinations are registered in a shared latitude–longitude coordinate system, the resulting supervision

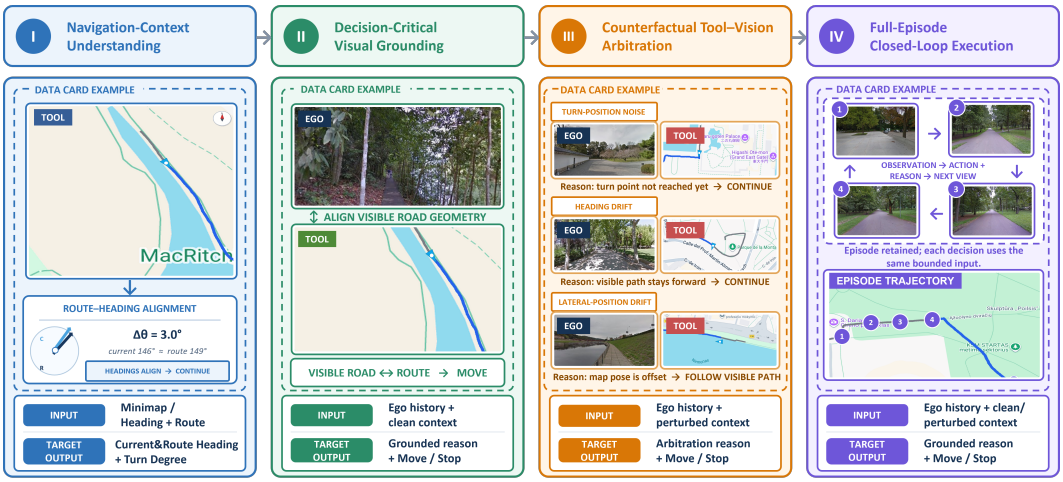

Figure 3: Training data and supervision targets for RouteGround. Each column shows a representative data card together with its policy input and target output. The four groups supervise route-context understanding, visual grounding at critical states, arbitration under counterfactual tool–vision conflicts, and consistent decisions over complete closed-loop episodes.

remains geographically aligned with the corresponding real-world streets. These aligned state–action pairs provide reliable behavior labels, but an action alone does not specify the evidence on which the decision should rely.

An oracle action tells the model what to do, but does not require it to connect route guidance, egocentric observations, and visible road structure before acting. We therefore augment each fixed action with a short evidence-grounded reason to encourage an explicit observation–comparison–decision process. The reason is generated through a constrained template from CityLoop ground truth and the already fixed oracle action, keeping every decision fact verifiable and action-consistent. To diversify the reason's linguistic realization, `gemini-3.1-flash-lite` (Google DeepMind, 2026) is used for constrained rewriting.

4.2 Capability-Structured Training Data

A homogeneous sample distribution is unlikely to cover route semantics, visual grounding, tool conflict, and long-horizon execution equally well, so we organize the constructed samples into four complementary capability groups:

$$\mathcal{D}_{\text{train}} = \{\mathcal{D}_{\text{context}}, \mathcal{D}_{\text{grounding}}, \mathcal{D}_{\text{counterfactual}}, \mathcal{D}_{\text{episode}}\}. \tag{7}$$

All four groups use the same structured policy interface and supervised fine-tuning objective, but differ in the decision situations and capabilities they emphasize; sample counts, replay ratios, and optimization settings are reported in supplementary Tables S3 and S4. In particular, context and grounding supervision establish route-semantic understanding and visual alignment, providing a clean reference against which tool–vision conflicts can be defined.

The first two groups respectively establish navigation-context understanding and local grounding at decision-critical states. $\mathcal{D}_{\text{context}}$ teaches the model to interpret position, heading, and route structure from a minimap, or current heading, remaining route, and next-turn semantics in heading–route mode, and map either representation to the shared action space. $\mathcal{D}_{\text{grounding}}$ focuses on turns, intersections, and arrival states, where route semantics must be associated with visible road geometry. Because the tool and oracle action usually agree in these clean samples, however, a policy can still minimize its loss by over-relying on the tool without learning the true correspondence between egocentric evidence and the minimap's road topology.

Counterfactual supervision explicitly breaks this tool-copying shortcut by exposing the model to samples in which navigation context conflicts with the current visual evidence. A counterfactual sample takes the form $(V_t, \widetilde{C}_t, a_t^*)$, where position or heading perturbations alter the visible navigation context, while the oracle action remains determined by the underlying physical state and visible

Table 2: Main results. POI: accuracy on 500 place requests (N/A: navigation policies without a tool-calling interface). Nav.: navigation context as a minimap image or, if unsupported, route text. All methods share the five-frame egocentric history, perturbations, and action budgets.

Model	**Nav.**	**POI**	**Keypoint Decision (%)**				**2DGS Closed-Loop (%)**			
	input	**Acc. (%)**	**None**	**Low**	**Med.**	**High**	**None**	**Low**	**Med.**	**High**
Proprietary Multimodal Models (API)										
GPT-5.6-sol (OpenAI, 2026)	Image	94.8	47.6	36.3	15.4	18.3	2.0	0	0	0
Claude-Sonnet-5 (Anthropic, 2026)	Image	**95.9**	16.5	19.4	10.3	17.2	0	0	0	0
Gemini-3.6-Flash (Google, 2026)	Image	93.3	24.9	19.0	10.6	15.0	0	0	0	0
Open-Source Multimodal Models										
Qwen3-VL-4B (Bai et al., 2025)	Image	91.6	15.4	17.2	12.8	15.4	0	0	0	0
Qwen3-VL-8B (Bai et al., 2025)	Image	92.1	23.1	25.6	23.4	23.8	4.0	0	0	0
Released Navigation Models										
NaVid (Zhang et al., 2024)	Text	N/A	25.3	26.7	24.9	26.4	0	0	0	0
NaVILA (Cheng et al., 2025)	Text	N/A	25.3	25.3	24.9	25.6	0	0	0	0
StreamVLN (Wei et al., 2026)	Text	N/A	25.3	23.4	24.2	26.0	0	0	0	0
OmniVLA-edge (Hirose et al., 2026)	Text	N/A	26.0	26.0	26.0	26.0	0	0	0	0
LeLaN (Hirose et al., 2024)	Text	N/A	20.5	20.1	20.5	20.5	0	0	0	0
Fine-Tuned Qwen3-VL-4B (Ours)										
RouteGround	Image	92.1	**78.4**	**60.1**	**59.0**	**44.0**	**46.0**	**26.0**	**32.0**	**18.0**

road geometry. By making route guidance inconsistent with the current street, these samples require the policy to judge whether the suggested maneuver is executable instead of directly copying the tool instruction. This supervision converts unconditional route following into visually conditioned tool arbitration, the central training mechanism of RouteGround, while representative full episodes extend the learned behavior beyond isolated conflicts.

Decision-critical and counterfactual samples train local choices, but cannot fully show how one action changes subsequent observations and accumulates into drift, loops, or premature stopping. $\mathcal{D}_{\text{episode}}$ therefore preserves representative complete closed-loop episodes spanning initialization, continuous motion, intersections, tool conflicts, recovery, late-route states, and active stopping. Each episode remains a physically continuous data and sampling unit, while every training input still follows the bounded interface in Equation 3 and exposes only recent egocentric and executed action history. Together, the four groups cover route understanding, visual grounding, conditional tool arbitration, and continuous closed-loop execution.

5 Experiments

We evaluate whether RouteGround improves decision-critical grounding under clean and corrupted guidance, whether these gains persist through complete 2DGS routes, and whether the same policy transfers zero-shot to a physical robot. Controlled training variants further isolate the contributions of context understanding, visual grounding, counterfactual arbitration, and full-episode supervision.

5.1 Experimental Setup

Deployment domains. CityLoop instantiates the same request-to-arrival policy interface across Google-based online navigation, MatrixCity 2DGS simulation (Li et al., 2023; Huang et al., 2024), and physical deployment on a Unitree G1. These domains respectively support controlled keypoint evaluation, complete-route rollouts, and zero-shot real-robot validation while sharing the policy inputs, action semantics, context refresh, and stopping criterion defined in Section 3. Backend, rendering, and robot configurations are provided in the supplementary material.

RouteGround fine-tuning. RouteGround fine-tunes Qwen3-VL-4B-Instruct (Bai et al., 2025) with 4-bit LoRA (Hu et al., 2022) by introducing the Stage I–IV data sequentially, with each stage initialized from the preceding adapter. All controlled variants share the same backbone and policy contract; per-stage data sizes and schedules are listed in supplementary Table S3, and optimization and hardware settings in Table S4.

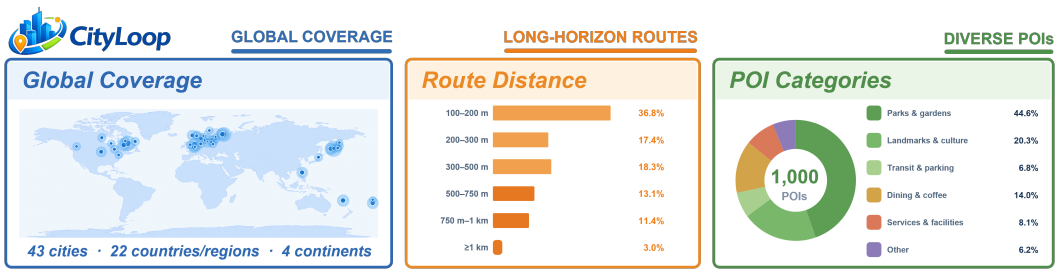

Figure 4: Diversity of the CityLoop benchmark: 43 cities in 22 countries or regions across four continents, with route-distance and POI-category distributions of the 1,000-route evaluation pool.

Compared methods. We compare against the proprietary multimodal, open-source multimodal, and released navigation models listed in Table 2. All methods receive the same five-frame egocentric history, and the navigation context follows the minimap setting of RouteGround; released navigation policies that cannot take a map instead receive the same route as text, and are adapted through their native interfaces to the shared action protocol. All methods use non-oracle inputs, matched perturbations, and identical action budgets and stopping criteria; navigation specialists are therefore reported as adapted cross-domain references rather than fully matched baselines. Model-specific prompts, checkpoints, and output mappings are given in the supplementary material.

CityLoop benchmark. The CityLoop source corpus spans 43 cities, 22 countries or regions, and four continents; Figure 4 summarizes its geographic coverage and the distance and destination diversity of the 1,000-route evaluation pool. Destination resolution is not the main bottleneck for tool-using models, which exceed 91% on the 500-request set; we thus focus on closed-loop route execution. Open-loop evaluation applies four matched perturbation levels to 1,000 base states, while closed-loop evaluation injects perturbations online over 50 2DGS routes. Training and evaluation are separated at the route level, and we report keypoint Decision Accuracy and 2DGS closed-loop Success Accuracy. Dataset composition, perturbation ranges, and metric details are provided in the supplementary material.

5.2 Main Results

Table 2 shows that route semantics remain difficult even with accurate localization; most baselines almost always predict forward (supplementary Table S10). The best baseline accuracy is 47.6% under clean guidance and 26.0% under medium localization errors, whereas RouteGround achieves 78.4% and 59.0%, respectively. Its advantage over larger and navigation-specialized models indicates that the gain comes from learned route grounding and conflict arbitration rather than scale alone.

The same advantage persists when local decisions are composed into complete routes. The best baseline obtains 4.0% clean 2DGS Success Accuracy and none succeeds under medium errors, while RouteGround reaches 46.0% and 32.0%. Thus, checking guidance against the visible scene keeps local errors from compounding into route failure under our protocol (Figure 5).

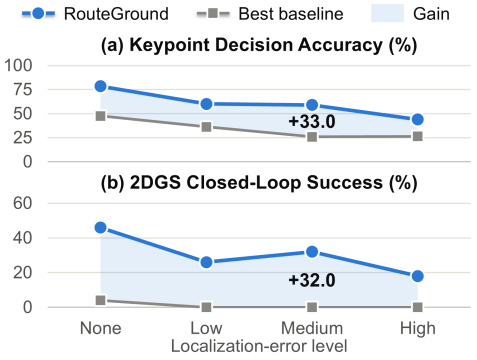

Figure 5: Robustness across localization-error levels. Each baseline point is the strongest non-RouteGround result at that level.

Table 3 identifies where these capabilities arise. Critical-state grounding is strong under clean guidance but brittle under medium errors (89.2% versus 16.6%); counterfactual conflicts raise the latter to 61.0%. Complete-route supervision then improves clean/medium 2DGS success from 30.0%/20.0% to 46.0%/32.0%. Stage III therefore learns tool–vision arbitration, while Stage IV improves its temporal composition; Stage I is not applicable, as it provides no action supervision. In

Table 3: Sequential curriculum ablation. Rows add context, grounding, conflict, and complete-route data.

Training Variant	Keypoint Decision		2DGS Closed-Loop	
	None	Med.	None	Med.
I: Context	N/A	N/A	N/A	N/A
I–II: + Grounding	89.2	16.6	32.0	14.0
I–III: + Conflicts	83.6	61.0	30.0	20.0
I–IV: + Full routes	**78.4**	**59.0**	**46.0**	**32.0**

Table 4: Input ablation of RouteGround. Ego.: current egocentric view; Hist.: four preceding views; Nav.: minimap or route text.

Ego.	Hist.	Nav.	Keypoint Decision		2DGS Closed-Loop	
			None	Med.	None	Med.
✓	✓	Image	78.4	**59.0**	46.0	**32.0**
✓	✓	Text	73.3	49.1	**58.0**	26.0
–	–	Image	88.3	33.3	50.0	0.0
✓	–	Image	**88.6**	37.0	52.0	12.0

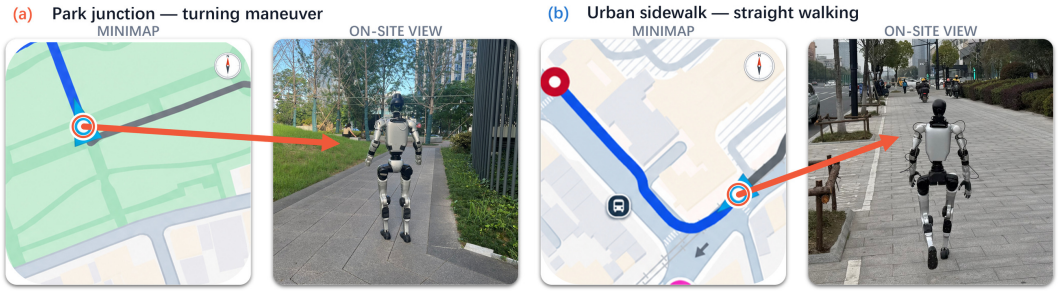

Figure 6: Zero-shot deployment on a Unitree G1. Representative robot observations are paired with their navigation states along a route.

Table 4, route text in place of the minimap still yields 73.3% clean accuracy, far above navigation models given the same text. Removing the egocentric views or their history helps under clean guidance (keypoint 88.3%/88.6%, 2DGS 50.0%/52.0%) but fails under medium errors, cutting keypoint accuracy from 59.0% to 33.3%/37.0% and 2DGS success from 32.0% to 0.0%/12.0%; the policy thus relies on vision when guidance is wrong.

5.3 Real-World Robot Deployment

Online navigation and 2DGS provide controlled evaluation, whereas physical deployment tests whether the same CityLoop interface remains operational under live sensing, localization updates, robot dynamics, and safety constraints. We deploy the final minimap policy zero-shot on a Unitree G1, without fine-tuning on robot observations or the test routes. The protocol contains 20 trials over five landmark and multi-waypoint routes averaging 1.0 km. The MOVE/STOP output is executed by the Unitree SDK2 high-level locomotion interface (Unitree Robotics, 2024). RouteGround inference runs on the robot's edge computer, while navigation context—including GPS, route guidance, and the minimap—is transmitted over a 5G hotspot. Six trials succeed (30%); route, success, intervention, and latency statistics are given in supplementary Table S9. Figure 6 visualizes synchronized robot observations and navigation states from a representative deployment. These trials evaluate physical transfer rather than claim a large-scale real-world benchmark.

6 Conclusion

Toward end-to-end urban navigation from request to arrival, we introduced CityLoop, a unified closed-loop sandbox that connects online navigation, 2DGS simulation, and physical deployment through one policy interface and turns map and street-view data into verifiable supervision. Evaluating existing models in CityLoop exposed weak route grounding, sensitivity to localization errors, and errors that compound over complete routes. RouteGround addresses these problems with counterfactual tool–vision conflicts and a four-stage curriculum, reaching 59.0% keypoint accuracy under medium errors and 46.0%/32.0% 2DGS Success Accuracy under clean/medium settings; zero-shot Unitree G1 trials provide initial evidence of physical transfer. Broader real-world evaluation across more cities, persistent localization biases, and safety-aware control remain future work.

AI USE STATEMENT

We used large language models (LLMs) to help draft and polish the text (rephrasing, grammar, and flow), format the LaTeX source, and search for related work. Beyond the `gemini-3.1-flash-lite` rewriting of training reasons described in Section 4.1, LLMs were not involved in research ideation, methodology, or experimental design. The authors verified all AI-assisted content and take full responsibility for this paper.

REFERENCES

Anthropic. Introducing Claude Sonnet 5. https://www.anthropic.com/news/claude-sonnet-5, 2026. Accessed 2026-07-29.

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. URL https://arxiv.org/abs/2511.21631. Preprint.

Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, 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. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Chen_TOUCHDOWN_Natural_Language_Navigation_and_Spatial_Reasoning_in_Visual_Street_CVPR_2019_paper.html.

Ziyi Chen, Yingnan Guo, Zedong Chu, Minghua Luo, Yanfen Shen, Mingchao Sun, Junjun Hu, Shichao Xie, Yang Kuan, Pei Shi, Zhining Gu, Lu Liu, Honglin Han, Xiaolong Wu, Mu Xu, and Yu Zhang. SocialNav: Training human-inspired foundation model for socially-aware embodied navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28796–28806, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Chen_SocialNav_Training_Human-Inspired_Foundation_Model_for_Socially-Aware_Embodied_Navigation_CVPR_2026_paper.html.

An-Chieh Cheng, Yandong Ji, Zhaojing Yang, Zaitian Gongye, Xueyan Zou, Jan Kautz, Erdem Biyik, Hongxu Yin, Sifei Liu, and Xiaolong Wang. NaVILA: Legged robot vision-language-action model for navigation. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.018. URL https://www.roboticsproceedings.org/rss21/p018.html.

Zedong Chu, Shichao Xie, Xiaolong Wu, Yanfen Shen, Minghua Luo, Zhengbo Wang, Fei Liu, Xiaoxu Leng, Junjun Hu, Mingyang Yin, et al. ABot-N0: Technical report on the VLA foundation model for versatile embodied navigation. *arXiv preprint arXiv:2602.11598*, 2026. URL https://arxiv.org/abs/2602.11598. Preprint.

Google. Gemini 3.6 Flash. https://ai.google.dev/gemini-api/docs/models/gemini-3.6-flash, 2026. Accessed 2026-07-29.

Google DeepMind. Gemini 3.1 Flash-Lite model card. https://deepmind.google/models/model-cards/gemini-3-1-flash-lite/, 2026. Published 2026-03-03. Accessed 2026-07-29.

Honglin He, Yukai Ma, Brad Squicciarini, Wayne Wu, and Bolei Zhou. From seeing to experiencing: Scaling navigation foundation models with reinforcement learning. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=0c7nAZjyr5.

Noriaki Hirose, Catherine Glossop, Ajay Sridhar, Dhruv Shah, Oier Mees, and Sergey Levine. LeLaN: Learning a language-conditioned navigation policy from in-the-wild videos. In *Proceedings of the 8th Conference on Robot Learning*, 2024. URL https://arxiv.org/abs/2410.03603.

Noriaki Hirose, Catherine Glossop, Dhruv Shah, and Sergey Levine. OmniVLA: An omni-modal vision-language-action model for robot navigation. In *IEEE International Conference on Robotics and Automation*, 2026. URL https://arxiv.org/abs/2509.19480.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 Conference Papers*. Association for Computing Machinery, 2024. doi: 10.1145/3641519.3657428. URL https://surfsplatting.github.io/.

Huaide Jiang, Yash Chaudhary, Yuping Wang, Zehao Wang, Raghav Sharma, Manan Mehta, Yang Zhou, Lichao Sun, Zhiwen Fan, Zhengzhong Tu, and Jiachen Li. Navtrust: Benchmarking trustworthiness for embodied navigation. *arXiv preprint arXiv:2603.19229*, 2026. URL https://arxiv.org/abs/2603.19229. Preprint.

Anqi Li, Zhiyong Wang, Jiazhao Zhang, Minghan Li, Yunpeng Qi, Zhibo Chen, Zhizheng Zhang, and He Wang. Urbanvla: A vision-language-action model for urban micromobility. *arXiv preprint arXiv:2510.23576*, 2025. URL https://arxiv.org/abs/2510.23576. Preprint.

Yixuan Li, Lihan Jiang, Linning Xu, Yuanbo Xiangli, Zhenzhi Wang, Dahua Lin, and Bo Dai. Matrixcity: A large-scale city dataset for city-scale neural rendering and beyond. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3205–3215, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Li_MatrixCity_A_Large-scale_City_Dataset_for_City-scale_Neural_Rendering_and_Beyond_ICCV_2023_paper.html.

Mingxuan Liu, Honglin He, Elisa Ricci, Wayne Wu, and Bolei Zhou. UrbanVerse: Scaling urban simulation by watching city-tour videos. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=HE6j2jtjII.

Xinhao Liu, Jintong Li, Yicheng Jiang, Niranjan Sujay, Zhicheng Yang, Juexiao Zhang, John Abanes, Jing Zhang, and Chen Feng. Citywalker: Learning embodied urban navigation from web-scale videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6875–6885, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Liu_CityWalker_Learning_Embodied_Urban_Navigation_from_Web-Scale_Videos_CVPR_2025_paper.html.

Harsh Mehta, Yoav Artzi, Jason Baldridge, Eugene Ie, and Piotr Mirowski. Retouchdown: Releasing touchdown on streetlearn as a public resource for language grounding tasks in street view. In *Proceedings of the Third International Workshop on Spatial Language Understanding*, pp. 56–62. Association for Computational Linguistics, 2020. URL https://aclanthology.org/2020.splu-1.8/.

Yanghong Mei, Yirong Yang, Longteng Guo, Qunbo Wang, Ming-Ming Yu, Xingjian He, Wenjun Wu, and Jing Liu. Urbannav: Learning language-guided embodied urban navigation from web-scale human trajectories. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18505–18513, 2026. doi: 10.1609/aaai.v40i22.38916. URL https://ojs.aaai.org/index.php/AAAI/article/view/38916.

Piotr Mirowski, Matthew Koichi Grimes, Mateusz Malinowski, Karl Moritz Hermann, Keith Anderson, Denis Teplyashin, Karen Simonyan, Koray Kavukcuoglu, Andrew Zisserman, and Raia Hadsell. Learning to navigate in cities without a map. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/e034fb6b66aacc1d48f445ddfb08da98-Abstract.html.

OpenAI. GPT-5.6 Sol. https://developers.openai.com/api/docs/models/gpt-5.6-sol, 2026. Accessed 2026-07-29.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A platform for embodied AI research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9339–9347, 2019. URL https://openaccess.thecvf.com/content_ICCV_2019/html/Savva_Habitat_A_Platform_for_Embodied_AI_Research_ICCV_2019_paper.html.

Raphael Schumann, Wanrong Zhu, Weixi Feng, Tsu-Jui Fu, Stefan Riezler, and William Yang Wang. VELMA: Verbalization embodiment of LLM agents for vision and language navigation in street view. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 18924–18933, 2024. doi: 10.1609/aaai.v38i17.29858. URL https://ojs.aaai.org/index.php/AAAI/article/view/29858.

Dhruv Shah and Sergey Levine. ViKiNG: Vision-based kilometer-scale navigation with geographic hints. In *Proceedings of Robotics: Science and Systems*, 2022. doi: 10.15607/RSS.2022.XVIII.019. URL https://www.roboticsproceedings.org/rss18/p019.html.

Unitree Robotics. Unitree SDK2. https://github.com/unitreerobotics/unitree_sdk2, 2024. Software repository. Accessed 2026-07-29.

Junhui Wang, Dongjie Huo, Zehui Xu, Yongliang Shi, Yimin Yan, Yuanxin Wang, Chao Gao, Yan Qiao, and Guyue Zhou. OpenBench: A new benchmark and baseline for semantic navigation in smart logistics. In *IEEE International Conference on Robotics and Automation*, 2025. URL https://arxiv.org/abs/2502.09238.

Meng Wei, Chenyang Wan, Xiqian Yu, Tai Wang, Yuqiang Yang, Xiaohan Mao, Chenming Zhu, Wenzhe Cai, Hanqing Wang, Yilun Chen, Xihui Liu, and Jiangmiao Pang. StreamVLN: Streaming vision-and-language navigation via slowfast context modeling. In *IEEE International Conference on Robotics and Automation*, 2026. URL https://arxiv.org/abs/2507.05240.

Tim Windecker, Manthan Patel, Moritz Reuss, Richard Schwarzkopf, Cesar Cadena, Rudolf Lioutikov, Marco Hutter, and Jonas Frey. NaviTrace: Evaluating embodied navigation of vision-language models. *arXiv preprint arXiv:2510.26909*, 2025. URL https://arxiv.org/abs/2510.26909. Preprint.

Wayne Wu, Honglin He, Jack He, Yiran Wang, Chenda Duan, Zhizheng Liu, Quanyi Li, and Bolei Zhou. Metaurban: An embodied AI simulation platform for urban micromobility. In *International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=kFsWpSxkFz.

Wayne Wu, Honglin He, Chaoyuan Zhang, Jack He, Seth Z. Zhao, Ran Gong, Quanyi Li, and Bolei Zhou. Towards autonomous micromobility through scalable urban simulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27553–27563, 2025b. URL https://openaccess.thecvf.com/content/CVPR2025/html/Wu_Towards_Autonomous_Micromobility_through_Scalable_Urban_Simulation_CVPR_2025_paper.html.

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. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Xia_Gibson_Env_Real-World_CVPR_2018_paper.html.

Ziyang Xie, Zhizheng Liu, Zhenghao Peng, Wayne Wu, and Bolei Zhou. Vid2Sim: Realistic and interactive simulation from video for urban navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1581–1591, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Xie_Vid2Sim_Realistic_and_Interactive_Simulation_from_Video_for_Urban_Navigation_CVPR_2025_paper.html.

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. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. URL https://arxiv.org/abs/2402.15852.

Lingfeng Zhang, Xiaoshuai Hao, Xizhou Bu, Yingbo Tang, Hongsheng Li, Jinghui Lu, Xiu-shen Wei, Jiayi Ma, Yu Liu, Jing Zhang, Hangjun Ye, Xiaojun Liang, Long Chen, and Wenbo Ding. Walk With Me: Long-horizon social navigation for human-centric outdoor assistance. *arXiv preprint arXiv:2604.26839*, 2026. URL https://arxiv.org/abs/2604.26839. Preprint.

Yuxiang Zhao, Yirong Yang, Yanqing Zhu, Yanfen Shen, Chiyu Wang, Zhining Gu, Pei Shi, Wei Guo, and Mu Xu. Bridging the indoor-outdoor gap: Vision-centric instruction-guided embodied navigation for the last meters. *arXiv preprint arXiv:2602.06427*, 2026. URL https://arxiv.org/abs/2602.06427. Preprint.

Table S1: Online-navigation backend configuration.

Configuration	Value
POI service	Google Places API (New)
Route service	Routes API v2 `ComputeRoutes`, pedestrian mode
Visual source	Street View metadata and panorama tiles
Context renderer	Google Static Maps
Cached-data window	April–July 2026
Panorama tiles	Zoom 2; 4×2 tile grid, cropped to panorama metadata
Egocentric rendering	1280×800; 90° horizontal FOV; 0° pitch
Minimap source	512×512, scale 2, roadmap, zoom 14–17
Minimap crop	North-up, fixed 320 m span; 80 m future and 30 m past route context

Table S2: 2DGS rendering and closed-loop execution configuration.

Configuration	Value
Scene identifier	`matrixcity_street_clean`
2DGS checkpoint	MatrixCity Street, iteration 30,000 point cloud
Renderer version	CityGaussian revision `2704734`
Image resolution	1280×720
Camera height	1.6 m
Camera configuration	75° horizontal FOV, 0° pitch, 20 m look-ahead
Action coordinates	Metric egocentric $(x, y, \Delta\theta)$
Maximum translation	5 m per high-level step
State validation	Scene walkability map; invalid moves are projected to the active route
Episode budget	$\lceil d_{\text{route}}/1.2\,\text{m} \rceil$

A Backend, Rendering, and Robot Configurations

This section records only the implementation configurations deferred by the deployment-domain paragraph of the main paper.

A.1 Online Navigation Backend

Each panorama ID defines one graph node. Latitude and longitude are converted to a local metric frame using $111{,}000$ m per latitude degree and the longitude correction $\cos(\text{latitude})$. An executed translation is matched to an outgoing edge no longer than 5 m when its direction and distance errors are at most 45° and 2 m, respectively; among valid edges, the minimum $0.8e_{\text{dir}} + 0.2e_{\text{dist}}$ candidate is used. Consecutive repeated route points are removed before rendering the context.

A.2 2DGS Rendering and Execution

The simulation backend uses a MatrixCity Street scene (Li et al., 2023) represented with 2D Gaussian Splatting (Huang et al., 2024). Table S2 records the renderer and execution settings not specified in the main paper.

A.3 Physical Robot Configuration

Only the Unitree SDK2 high-level locomotion interface is used (Unitree Robotics, 2024); the stock low-level gait controller is left unchanged. A human safety operator may pause or override execution for traffic or regulatory hazards, and every such event is counted as an intervention.

Table S3: Training data introduced at each incremental stage. "Replay" denotes earlier critical-state or conflict examples retained while introducing the named capability.

Stage	New	Replay	Total	Training schedule
I	6,000	0	6,000	3 epochs, 2×10^{-4}
I–II	6,188	0	6,188	3 epochs, 2×10^{-4}
I–III	5,500	1,100	6,600	3 epochs, 2×10^{-4}
I–IV	12,000	1,504	13,504	1 epoch, 3×10^{-5}

Table S4: RouteGround optimization and hardware configuration.

Configuration	Value
Base checkpoint	`Qwen3-VL-4B-Instruct`
Quantization	4-bit NF4, double quantization, BF16 compute
Adaptation	LoRA (Hu et al., 2022), rank 16, $\alpha = 32$, dropout 0.05
Target modules	`q/k/v/o_proj`, `gate/up/down_proj`
Image pixels	50,176–401,408
Optimizer	Paged 8-bit AdamW
Learning rate	Stage-specific; Table S3
Batch / accumulation	1 / 8 steps on one GPU (effective batch 8)
Warmup / weight decay	0.03 / 0
Scheduler	Linear decay
Random seeds	42 for training shuffle and held-out sampling
Hardware	Two RTX 4090 GPUs (24 GB each); one GPU per training run
Checkpoint selection	Final adapter after the fixed epoch budget

B Training Composition, Optimization, and Hardware

Table S3 gives the data introduced and replayed at each incremental action-training stage. Stage I contains context-only examples and is therefore not evaluated as an action policy in Table 3 of the main paper.

C Model Prompts, Checkpoints, and Output Mappings

The model-specific adapters used for the methods reported in Table 2 of the main paper are summarized in Table S5. All adapters exclude oracle state and future observations.

Prompt templates. General multimodal models receive the user request, five egocentric frames ordered from $t-4$ to t, the current navigation context, and executed action history. The prompt asks for exactly one JSON object:

```
{"action_type":"move", "x_right_m":x,
"y_forward_m":y, "delta_heading_deg":theta,
"reason":"..."}
or {"action_type":"stop","reason":"..."}.
```

It defines right and forward as positive, limits $\sqrt{x^2+y^2}$ to 5 m, and requires visual walkability and destination evidence before acting or stopping. Navigation specialists receive the same five frames when supported and the following route-text template:

```
Follow the pedestrian navigation route using the visual
observations.
The reported localization may be inaccurate, so use the
visual history
to decide the next physical navigation action.
Reported current heading: {heading} degrees. {remaining
route}
```

Table S5: Model endpoints and interface adapters. Each interface cell follows standardized input → native output; N/A denotes models prompted to emit the shared high-level action protocol directly.

Model	Endpoint / checkpoint	Standardized input → native output	Protocol adaptation
General multimodal models			
GPT-5.6-sol (OpenAI, 2026)	`gpt-5.6-sol`	Five-frame history + minimap → $(x, y, \Delta\theta)$ or STOP	N/A (direct output)
Claude-Sonnet-5 (Anthropic, 2026)	`claude-sonnet-5`	Five-frame history + minimap → $(x, y, \Delta\theta)$ or STOP	N/A (direct output)
Gemini-3.6-Flash (Google, 2026)	`gemini-3.6-flash`	Five-frame history + minimap → $(x, y, \Delta\theta)$ or STOP	N/A (direct output)
Qwen3-VL-4B/8B (Bai et al., 2025)	`Qwen3-VL-4B/8B-Instruct`	Five-frame history + minimap → $(x, y, \Delta\theta)$ or STOP	N/A (direct output)
Navigation models			
NaVid (Zhang et al., 2024)	`Jzzhang/NaVid/navid-7b-full-224-video-fps-1-grid-2-r2r-rxr-training-split`	Five-frame history + route text → text action with distance or angle	Native motion translated to $(x, y, \Delta\theta)$ or STOP
NaVILA (Cheng et al., 2025)	`navila-llama3-8b-8f`	Five-frame history + route text → mid-level text action with distance or angle	Native motion translated to $(x, y, \Delta\theta)$ or STOP
StreamVLN (Wei et al., 2026)	`mengwei0427/StreamVLN_Video_qwen_1_5_r2r_rxr_envdrop_scalevln_v1_3`	Five-frame history + route text → four-step action chunk (25 cm / 15° / stop)	Native motion translated to $(x, y, \Delta\theta)$ or STOP
OmniVLA-edge (Hirose et al., 2026)	`NHirose/omnivla-edge`	Five-frame history + route text → eight local waypoints (x_i, y_i, h_i^x, h_i^y)	Native motion translated to $(x, y, \Delta\theta)$ or STOP
LeLaN (Hirose et al., 2024)	`wo_col_loss_wo_temp.pth`	Five-frame history + route text → 24 velocity commands (v_i, ω_i)	Native motion translated to $(x, y, \Delta\theta)$ or STOP

Decoding and invalid outputs. Each instance is run once with temperature zero. The evaluator extracts the first valid action object, clips translations to 5 m, and forces $\Delta\theta = 0$ for the reported translation policy. An invalid output becomes a zero-displacement MOVE, never an oracle STOP. NaVid and NaVILA text actions are parsed as forward/left/right/stop; StreamVLN's native 25 cm and 15° actions are integrated before the same conversion. OmniVLA-edge's local trajectory is reduced to its first bounded displacement, while LeLaN's velocity command is integrated over one high-level step. Because these conversions leave each model with a different mean per-step displacement, the episode budget $\lceil d_{\text{route}}/1.2\,\text{m} \rceil$ of Table S2 is adjusted per model from its measured mean displacement, so that the destination stays reachable within the budget and no model is failed by step count alone.

D BENCHMARK COMPOSITION, PERTURBATIONS, AND METRICS

For the evaluation pools summarized in the main paper, this section reports the fine-grained composition and matched perturbation protocol.

D.1 COMPOSITION AND SPLITS

The separate 500-request destination-resolution set contains 241 exact-place, 140 nearest-category, and 119 intent-based requests. We report *full-sequence correctness*. An exact-place request must directly call `get_route` for the named destination without first searching. Nearest-category and intent-based requests must issue a valid `search_place` call, expose the oracle POI among the

Table S6: Route evaluation pool statistics.

Statistic	Value
Source walking-route files	2,679
Street View regions	136
Evaluation-route distance	379.43 km
Minimum / maximum length	100.17 / 1,094.68 m
Median / mean length	273.69 / 379.43 m
Routes at least 500 m	275 (27.5%)
Exact-place requests	482
Nearest-category requests	280
Intent-based requests	238

Table S7: Open-loop perturbation magnitudes.

Perturbation	Low	Medium	High
Turn position (m)	4–8	12–20	16–24
Heading (°)	0–15	15–45	30–45
Lateral position (m)	3–8	8–20	14–24

Table S8: Closed-loop perturbation magnitudes.

Perturbation	Low	Medium	High
Turn position (m)	3–8	8–15	15–25
Heading (°)	0–15	15–45	45–60
Lateral position (m)	3–8	8–20	20–30

returned candidates, and then call `get_route` for an acceptable POI ID. A correct final place reached through the wrong tool sequence is scored incorrect.

The open-loop evaluation contains 1,000 base decision states and their matched none, low, medium, and high variants, for 4,000 instances. The base states contain 700 turn-position, 200 heading-drift, and 100 lateral-position cases. The available 2DGS pool contains 102 routes ranging from 100.69 to 707.83 m, with a mean of 307.47 m. A fixed 50-route subset is used for closed-loop evaluation. It contains 29 exact-place, 16 nearest-category, and five intent-based routes. The subset is stratified by task type and sampled at approximately even distance intervals; the same IDs are then used for every model and noise level. Training/evaluation exclusion is applied at the source-route identifier rather than individual frames.

D.2 Matched Perturbations

For the open-loop benchmark, the clean and perturbed variants preserve the same physical observation, oracle action, and task identifier.

Closed-loop perturbations are sampled as the route evolves. On ordinary steps, their application probabilities are 10%, 20%, and 30% from low to high. Within a turn window, turn-position noise is sampled first with probabilities 50%, 60%, and 70%.

A stable seed derived from task, step, noise level, and draw name reproduces the same perturbation plan across models.

D.3 Metrics

For N decision states, a STOP target is correct only when the model stops. A MOVE target is correct when the output is a nonzero translation of at most 5 m whose direction differs from the oracle edge by less than 25°. Keypoint Decision Accuracy averages this indicator over the N states. For M episodes under the common episode budget and the arrival threshold $\tau_{\text{arrive}} = 5$ m, 2DGS Closed-

Table S9: Preliminary physical-robot deployment statistics. A trial succeeds only when the policy predicts STOP within 5 m of the destination.

Metric	Value
Robot platform	Unitree G1
Routes / trials	5 / 20
Route length	1.0 km average; 0.4–1.6 km
Task type	Exact-place + nearest-category + intent-based
Success rate	6 / 20 (30%) within 5 m
Final distance	3.7 m
Intervention rate	12% safety / regulatory
Decision latency	3.9 s / high-level step

Table S10: Keypoint Decision Accuracy (%) by ground-truth direction under each perturbation level. Each level contains 1,000 states: 290 forward (F), 366 left (L), 322 right (R), and 22 backward (B).

	None				Low				Medium				High			
Model	F	L	R	B	F	L	R	B	F	L	R	B	F	L	R	B
GPT-5.6-sol	48.3	45.9	50.0	31.8	31.7	39.9	37.6	18.2	13.8	15.0	18.3	0.0	20.3	19.9	13.7	31.8
Claude-Sonnet-5	56.9	0.0	0.0	0.0	65.9	0.0	0.9	0.0	34.1	0.0	1.2	0.0	59.3	0.0	0.0	0.0
Gemini-3.6-Flash	44.1	16.1	17.1	31.8	27.6	12.0	17.1	50.0	24.1	6.0	4.3	0.0	35.5	6.0	7.8	0.0
Qwen3-VL-4B	53.1	0.0	0.0	0.0	59.3	0.0	0.0	0.0	44.1	0.0	0.0	0.0	53.1	0.0	0.0	0.0
Qwen3-VL-8B	78.3	0.0	1.2	0.0	87.2	0.0	0.9	0.0	80.7	0.0	0.0	0.0	82.1	0.0	0.0	0.0
NaVid	81.0	4.9	0.0	0.0	83.4	6.0	0.9	0.0	78.3	4.9	1.2	0.0	81.0	6.8	1.2	0.0
NaVILA	82.1	3.0	1.2	0.0	83.4	3.0	0.0	0.0	82.1	3.0	0.0	0.0	84.5	3.0	0.0	0.0
StreamVLN	81.0	4.9	0.0	0.0	76.9	3.0	0.0	0.0	79.7	3.0	0.0	0.0	79.7	7.9	0.0	0.0
OmniVLA-edge	89.7	0.0	0.0	0.0	89.7	0.0	0.0	0.0	89.7	0.0	0.0	0.0	89.7	0.0	0.0	0.0
LeLaN	65.9	1.9	2.2	0.0	63.1	1.9	3.4	0.0	65.9	1.9	2.2	0.0	65.9	1.9	2.2	0.0
RouteGround (Image)	100.0	74.0	62.4	100.0	82.1	50.0	51.2	68.2	93.8	41.0	45.3	100.0	73.4	36.1	27.3	31.8
RouteGround (Text)	98.6	63.1	61.5	81.8	79.7	3.0	14.6	0.0	100.0	29.0	29.5	0.0	58.3	18.0	12.4	0.0

Loop Success is

$$\frac{1}{M}\sum_{j=1}^{M} \mathbb{1}[d(p_T^j, g^j) \le \tau_{\text{arrive}}]\mathbb{1}[a_T^j = \text{Stop}]. \tag{S1}$$

E Robot Deployment Statistics

To complement the simulated rollouts, we conduct 20 physical trials over five outdoor routes using a Unitree G1. RouteGround inference runs onboard, while GPS, heading, minimap, and route updates are transmitted through a 5G hotspot; the predicted $(x, y, \Delta\theta)$ or STOP command is executed through the Unitree SDK2 high-level locomotion interface. Table S9 summarizes navigation success, terminal accuracy, safety interventions, and end-to-end decision latency.

F Per-Direction Keypoint Accuracy

Table S10 splits Keypoint Decision Accuracy by the ground-truth motion direction. Most baselines answer forward almost exclusively: the Qwen3-VL models, Claude-Sonnet-5, and all five navigation models reach up to 89.7% on forward states but at most 7.9% on left or right turns, which keeps their overall accuracy close to or below the 29.0% share of forward states. GPT-5.6-sol is the only baseline with balanced turn accuracy under clean guidance (45.9% left, 50.0% right), but it falls to 15.0% and 18.3% under medium perturbations. RouteGround retains substantial turn accuracy at every level, although its left and right accuracy still declines as the perturbation grows. Backward states are rare (22 per level), so their percentages vary strongly.