One Scan is Enough: Demonstration-Free Adaptation for Language-Guided Navigation

Zhuoyuan Yu* **Yuxing Long*** **Du Wang***
Junyang Wang **Hongwei Fan** **Hao Dong†**
CFCS, School of Computer Science, Peking University
PrimeBot

Abstract

Deploying a language-guided navigation policy in a new building usually means watching it fail. Public-benchmark VLAs degrade scene by scene as layout, appearance, and language drift, and the standard fix of teleoperating fresh trajectories on site does not scale. We replace human demonstration with a single brief iPhone scan. **Scan2Instruction** compiles that scan into a complete training environment, automatically synthesizing executable navigation episodes with grounded instructions. **PRPT-KS** then adapts the policy through a closed-form Pareto utility read directly off the reconstructed scene, with no learned critic and no human labels. Across five real environments, the two designs together drive our RealNav to a **85% real-robot success rate**, compared with **49%** for a RealNav baseline post-trained on human demonstrations, and still achieve **82% in unscanned regions of the same buildings**, compared with **41%** for the same baseline. Our work turns demonstration-free scene adaptation for language-guided navigation from an aspiration into a working reality.

1 Introduction

Mobile robots become most useful when users can simply talk to them, with requests like *go to the printer along the corridor*, *find the white mug in the kitchen*, or *look for somewhere to sit*. Modern vision-and-language navigation policies perform reasonably on benchmark trajectories, but performance falls off sharply once they leave those benchmarks: every new building changes both what the robot sees and what users tend to ask. Adapting to each target space by collecting real-robot demonstrations is slow, expensive, and does not scale to the long tail of homes, offices, and institutions. This paper asks: ***Can a robot adapt to a new scene from a single passive scan, without any demonstrations?***

Recent advances in 3D Gaussian Splatting [Kerbl et al., 2023] reconstruct real environments from a casual scan and re-render photorealistic first-person observations inside them. A line of real-to-sim-to-real navigation systems builds on this capability to adapt a navigation policy directly in the reconstructed target scene [Chhablani et al., 2025, Liu et al., 2026, Zhu et al., 2025, Xie et al., 2025]. These systems demonstrate the value of reconstructed environments for navigation policy adaptation. Building on this foundation, we study how a single passive scan can automatically supply both grounded language-navigation episodes and geometry-derived post-training signals, without human-provided task demonstrations in the target scene. Figure 1 shows the overall pipeline.

The first component, **Scan2Instruction**, reframes a scanned scene from a passive renderer into an active source of training tasks. From the scan we extract both a Gaussian rendering of the environment and a queryable geometric map of free space, and then sample feasible navigation episodes spanning

*Equal Contribution.

†Corresponding author: hao.dong@pku.edu.cn.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



Figure 1: **One iPhone scan, a deployable language-guided navigation policy. Left: Scan2Instruction** compiles a brief handheld scan into scene-specific training episodes, and **PRPT-KS** adapts **RealNav** on them with no demonstrations and no human labels. **Right:** real-robot rollouts following natural-language instructions in the scanned scenes.

three complementary instruction types: Route following, Object Goal, and Human Demand. For each episode, the compiler plans a collision-aware reference trajectory, renders first-person observations along it, and synthesizes natural-language instructions that are visually grounded in the rendered viewpoints and spatially aligned with the planned route. After a passive scan and one-time manual scene calibration, task generation and adaptation require no human demonstrations.

The second component, **PRPT-KS**, is a scene-grounded reinforcement post-training algorithm for our continuous-trajectory navigation VLA **RealNav**. Once the scene is known, standard RL's expensive components become unnecessary: the learned critic, a learned reward model, large-scale exploration, and human preference labels. PRPT-KS reads a three-dimensional Pareto utility vector of path progress, obstacle safety, and correct stopping directly off the known scene, and instantiates the same vector three ways: as a scalar projection that drives the policy gradient, as a per-axis admissibility test that lets a planner intervene on unsafe or low-progress actions, and as a vector dominance check that re-weights preference pairs recovered from those interventions. Adaptation thus reaches strong performance using only scene-grounded signals—no task demonstrations, learned critic, or human labels in the target environment.

We evaluate on five real scenes (office, cottage, teaching building, two apartments) in both Gaussian simulation and on the robot. The scene-adapted RealNav reaches a **85% real-robot success rate on scanned regions, compared with 49% for the same model post-trained on human demonstrations, and generalizes to never-scanned regions at 82% versus 41% for the same baseline**.

Our contributions are threefold:

- We propose **Scan2Instruction**, the first training-data compiler that turns a single passive iPhone scan into scene-specific episodes with grounded language instructions, removing the per-scene teleoperation-and-annotation bottleneck of language-guided navigation.
- We design **PRPT-KS**, a scene-grounded RL post-training algorithm that reads a three-axis Pareto utility off the known scene and reuses it as policy gradient, planner admissibility test, and preference re-weighting, replacing the learned critic, learned reward model, exploration, and human labels.
- **Experiments** on five real scenes show that they drive **RealNav** to **85%** real-robot success on scanned regions and **82%** on unscanned ones, compared with 49% / 41% for the baseline post-trained on human demonstrations.

2 Related Work

2.1 3D Gaussian Splatting for Embodied Navigation

3D Gaussian Splatting (3DGS) reconstructs real scenes and renders robot-centric observations, reducing the sim-to-real visual gap for embodied navigation [Kerbl et al., 2023, Chhablani et al., 2025, Liu et al., 2026, Zhu et al., 2025, Xie et al., 2025]. We share the reconstruct-and-render setting with this line of work. Prior systems use reconstructed scenes for simulation and navigation policy training. Our focus is to automatically compile a single passive scan into grounded language-action episodes and geometry-derived post-training signals. Scan2Instruction samples executable tasks, plans collision-aware trajectories, renders first-person observations, and generates diverse instructions grounded in those trajectories. Thus, the scene becomes both a simulator and a source of language-action episodes and geometry-derived post-training signals for our continuous-trajectory VLA RealNav.

2.2 Reinforcement Post-Training for Navigation Policies

Reinforcement learning is widely used for embodied navigation, with training signals typically obtained through massive simulator interaction [Wijmans et al., 2020], map-based priors [Chaplot et al., 2020b,a], target-driven objectives [Zhu et al., 2017], or cross-modal reward shaping for VLN [Wang et al., 2019, Ma et al., 2019]. A complementary line uses real-world experience and expert interventions for policy improvement [Ross et al., 2011, Kahn et al., 2021a,b], often combined with general-purpose optimization and preference-learning tools [Schulman et al., 2017, Ouyang et al., 2022, Rafailov et al., 2023]. These methods explore from scratch, depend on discrete graphs with expert paths, or require real-robot trial-and-error. In contrast, since the target scene is already scanned, we query free space, reference paths, obstacle distances, and goal regions directly from geometry. PRPT-KS converts this geometry into a closed-form Pareto utility along three axes (progress, safety, stopping), used as a scalar advantage replacing the learned critic, an admissibility test gating planner intervention, and a dominance weight on the resulting preference pairs. Together they adapt RealNav without human takeover, reward models, or learned per-scene critics.

3 Method

3.1 Problem formulation

Task Setting. A mobile robot is deployed in a target environment for which the only prior input is a single passive scan $\mathcal{S}$ of the scene. At test time the robot receives natural-language instructions $i \in \mathcal{I}$ from the user. Adaptation is *demonstration-free*: the scan $\mathcal{S}$ may be used freely, but no human-provided task trajectory is available in the target scene.

Optimization Objective. Let $\tau = (o_{1:T}, a_{1:T})$ denote a rollout produced by a navigation policy $\pi_\theta(a_t \mid o_{1:t}, i)$ in the target environment, and let $R(\tau, i) \in \{0, 1\}$ indicate whether the trial succeeds for instruction i. We seek an *adaptation algorithm* $\mathcal{A} : (\theta_0, \mathcal{S}) \mapsto \theta$ such that

$$\pi_\theta^\star \;=\; \arg\max_{\theta}\; \mathbb{E}_{i\sim\mathcal{I}}\, \mathbb{E}_{\tau\sim\pi_\theta(\cdot|\mathcal{S},\, i)}[\, R(\tau,\, i)\,] \quad \text{s.t.} \quad \theta = \mathcal{A}(\theta_0,\, \mathcal{S})\,, \tag{1}$$

where θ_0 are the parameters of a base navigation VLA pretrained without target-scene access.

3.2 RealNav: A Continuous-Trajectory Navigation VLA

Model architecture. RealNav is a continuous-trajectory navigation VLA built on Qwen3-VL [Bai et al., 2025]. At each decision step t, the input is a natural-language instruction i together with a sampled history of RGB observations $o_{1:t}$, and we write $s_t = (o_{1:t}, i)$ for the full conditioning. The instruction and observation tokens are fed through the Qwen3-VL chat backbone, and after the user turn we append a small set of dedicated *navigation tokens*. These navigation tokens absorb the cross-modal context through self-attention, and their final hidden states are read out by a lightweight MLP *action head*. Concretely, RealNav implements the deterministic mapping

$$\mu_\theta(s_t) \;=\; f_{\text{head}}\Big(\, h_{\text{nav}}(s_t;\, \theta_{\text{vlm}})\, \Big) \;\in\; \mathbb{R}^{3K}, \tag{2}$$

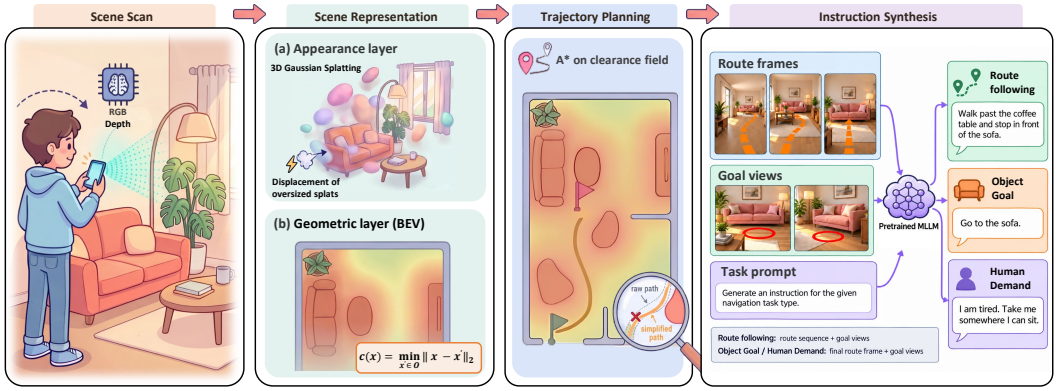

Figure 2: **Scan2Instruction pipeline.** A pretrained MLLM takes rendered images, goal views, and a type-specific prompt to generate Route following, Object Goal, or Human Demand instructions.

where $h_{\text{nav}}(\cdot;\theta_{\text{vlm}})$ collects the final hidden states of the appended navigation tokens after passing s_t through the Qwen3-VL backbone, and f_{head} is the MLP action head. The 24-dimensional output is reshaped into $K = 8$ future local waypoints $\{(x_k, y_k, \theta_k)\}_{k=1}^{K}$ in the current robot frame, with (x_k, y_k) a planar translation and θ_k a heading change. Stop is inferred when consecutive executed waypoint differences fall below preset position and heading thresholds, and is correct only when the robot is within the target radius.

Navigation pretraining. To establish a navigation-capable initial policy before any target-scene adaptation, we train μ_θ on R2R-CE and RxR-CE [Anderson et al., 2018, Ku et al., 2020, Krantz et al., 2020] by converting their reference trajectories into local waypoint labels $a_t^\star$ and fitting an MSE behavior-cloning loss directly on the deterministic output.

Gaussian policy parameterization for post-training. Both the policy-gradient ratio and the preference log-likelihood introduced in Section 3.4 treat the waypoint vector as a Gaussian $\pi_\theta(a_t \mid s_t) = \mathcal{N}(\mu_\theta(s_t), \Sigma)$ around the deterministic head, with isotropic covariance Σ in normalized waypoint coordinates.

3.3 Scan2Instruction: Compiling a scan into language-navigation episodes

Scan2Instruction turns a single scan into a dataset $\mathcal{D}_{\text{scene}}$ of executable language-navigation episodes for the target environment. Figure 2 summarizes this stage.

Scene representation. The scan $\mathcal{S}$ is captured as an RGB-D video on a handheld iPhone with a built-in LiDAR sensor, with per-frame camera poses provided by ARKit or estimated offline using ORB-SLAM3 [Campos et al., 2021]. From the posed RGB-D scan we recover a joint scene representation with two coupled layers. The **appearance layer** is a planar-regularized 3D Gaussian splat [Kerbl et al., 2023, Chen et al., 2024] reconstructed with LiDAR depth supervision. Gaussians with oversized projected footprints are moved far along the viewing ray to prevent them from occluding the scene and blurring the rendered view. The **geometric layer** is built by slicing an obstacle-height band from Gaussian occupancy projections, followed by thresholding, morphological cleanup, and connectivity filtering to obtain a reachable free-space mask. After one-time manual map alignment and traversability confirmation, we compute a continuous *clearance field*

$$c(\mathbf{x}) \;=\; \min_{\mathbf{x}' \in \mathcal{O}} \|\mathbf{x} - \mathbf{x}'\|_2, \tag{3}$$

where $\mathcal{O}$ denotes the blocked cells of the eroded planning map $\mathcal{M}$ defined below, so that $c(\mathbf{x})$ is the distance to the boundary of $\mathcal{M}$. The clearance field $c(\cdot)$ supplies a continuous geometric quantity for trajectory planning below. The eroded free-space planning map $\mathcal{M}$, defined below, supplies the binary membership check inside U_{safe} (Eq. (10)) and in the admissibility test of Section 3.4.

Trajectory planning. Each episode begins with an executable reference trajectory planned purely from geometry, providing a concrete path for the language synthesis stage to describe.

We sample starts from $\mathcal{M}$ and goals from the same connected component of $\mathcal{M}$; a goal with $c(\mathbf{x}) > c_{\text{goal}}$ is moved to the nearest cell in that component with $c(\mathbf{x}) \le c_{\text{goal}}$, so that it lies next to an object that an instruction can name. Between each retained pair we run $A^\star$ with a clearance-penalized edge cost,

$$\text{cost}(\mathbf{x} \to \mathbf{x}') \;=\; s(\mathbf{x}, \mathbf{x}') + \frac{w}{\max(d, \varepsilon)}\, m(d), \quad m(d) = \begin{cases} d_{\min}/(d+\varepsilon), & d < d_{\min}, \\ 1, & \text{otherwise}, \end{cases} \tag{4}$$

where s is the grid-step length, d is obstacle distance in grid units, w weights the clearance penalty, $d_{\min}$ sets the preferred clearance, and $\varepsilon > 0$ stabilizes the denominator. The reachable free-space mask is eroded to form the planning map $\mathcal{M}$ with a safety margin; endpoints and traversed cells must remain inside $\mathcal{M}$ and satisfy clearance constraints. Safety-constrained RDP [Douglas and Peucker, 1973] retains only chords that maintain minimum clearance. Accepted pairs are sampled in both directions.

Instruction synthesis. Scan2Instruction uses a pretrained multimodal large language model (MLLM) to synthesize three instruction types:

- **Route following** — actions and landmarks describing the route;
- **Object Goal** — a named target object, without route directions;
- **Human Demand** — a functional request, without naming the target object or describing the route.

The MLLM performs visual grounding and language generation: it interprets the rendered scene, identifies the destination, and expresses the task according to a type-specific text prompt. Route frames carry an orange future-path trail, while goal views mark the destination with a red ground circle. For instruction type γ, generation uses a single MLLM call:

$$i = \text{MLLM}\big(\{\tilde{o}_t\}_{t \in \mathcal{T}_\gamma},\, \mathcal{G};\, P_\gamma\big)\,, \tag{5}$$

where $\tilde{o}_t$ are guided route images, $\mathcal{G}$ contains the goal views, and P_γ specifies the instruction type and wording constraints. Route following uses the route sequence and goal views; Object Goal and Human Demand use goal views and the last route frame.

For each planned route, we pair the MLLM-generated instruction with the planner's reference path, poses, and local waypoint labels. We render a separate sequence of policy observations with camera intrinsics and heights different from those used for instruction synthesis, omitting the visual guides. Each compiled episode therefore combines the language instruction, policy observations, and planner supervision as

$$e \;=\; \big(\, i,\, \{o_t\}_{t=1}^{T},\, \{p_t\}_{t=1}^{T},\, \{a_t^\star\}_{t=1}^{T},\, \tau^\star,\, \gamma \,\big), \tag{6}$$

with i the instruction, o_t a rendered RGB observation, p_t a reference pose, $a_t^\star$ the local waypoint label, $\tau^\star$ the planner's reference path, and γ the family identifier. The union $\mathcal{D}_{\text{scene}} = \{e\}$ is the dataset Scan2Instruction hands to PRPT-KS.

For evaluation episodes, the synthesized instructions are additionally checked manually.

3.4 PRPT-KS: Pareto-utility post-training for known-scene navigation

PRPT-KS, short for **P**areto-utility **R**einforcement **P**ost-**T**raining in **K**nown **S**cenes, replaces the parts of standard RL that usually require expensive supervision with a single *Pareto utility vector* read directly off the known scene. Concretely, every candidate action a at state s_t is scored by

$$\mathbf{U}(s_t, a) \;=\; \big(\, U_{\text{prog}}(s_t, a),\; U_{\text{safe}}(s_t, a),\; U_{\text{stop}}(s_t, a) \,\big), \tag{7}$$

whose three components quantify path progress, obstacle safety, and correct stopping respectively. Writing $\succ_P$ for the standard Pareto dominance on $\mathbb{R}^3$ (weak inequality on every axis and strict inequality on at least one), the remainder of this section instantiates PRPT-KS as three different read-outs of the *same* $\mathbf{U}$: a scalar projection that drives the policy gradient (Eq. (12)), a per-axis admissibility test that gates planner intervention (Eq. (14)), and a vector dominance check that re-weights pairwise preferences (Eq. (15)). Because $\mathbf{U}$ is closed-form in known-scene geometry, none of these read-outs require a learned critic or reward model. Figure 3 illustrates the resulting training loop.

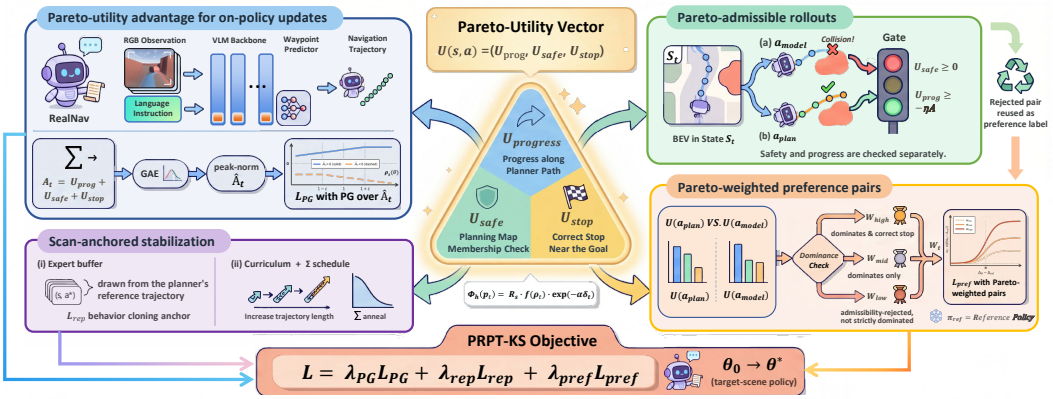

Figure 3: **PRPT-KS pipeline.** A Pareto utility vector read from known-scene geometry drives rollout updates, gates rollouts, and weights preference pairs.

Three Pareto utilities from known-scene geometry. Let $\tau^\star$ be the planner's reference path for the current episode and p_t the robot pose. The nearest reference-pose index j_t among N reference poses gives completion $\rho_t = j_t/(N-1)$; δ_t is distance to that pose. We use

$$\Phi_h(p_t) = R_s f(\rho_t) \exp(-\alpha \bar{\delta}_t), \quad \bar{\delta}_t = \begin{cases} 0, & \delta_t < \delta_{\text{dead}}, \\ \delta_t, & \text{otherwise.} \end{cases} \tag{8}$$

Here $R_s > 0$ scales the potential, f is an increasing affine function of completion, $\alpha > 0$ controls deviation decay, and δ_{dead} defines a deviation dead zone. The utility components are

$$U_{\text{prog}}(s_t, a) = \gamma \Phi_h(p^a_{t+1}) - \Phi_h(p_t), \tag{9}$$

$$U_{\text{safe}}(s_t, a) = -\lambda_{\text{obs}} \mathbf{1}[\exists w \in \mathcal{E}(a) : w \notin \mathcal{M}], \tag{10}$$

$$U_{\text{stop}}(s_t, a) = R_{\text{succ}} \mathbf{1}[a \text{ is a stop and } \|p^a_{t+1} - g\|_2 < r_{\text{succ}}]. \tag{11}$$

Here g is the goal position, and $\mathcal{E}(a)$ contains the interleaved execution subset of predicted waypoints, truncated at stopping. The safety penalty $\lambda_{\text{obs}} > 0$ applies when an executed waypoint leaves the eroded planning map $\mathcal{M}$, inside which every reference trajectory is planned. Training uses success reward $R_{\text{succ}} > 0$ and success radius r_{succ}.

Pareto-utility advantage for rollout updates. The single-step training signal is a one-dimensional projection of $\mathbf{U}$ obtained by summing its components,

$$A_t = U_{\text{prog}}(s_t, a_t) + U_{\text{safe}}(s_t, a_t) + U_{\text{stop}}(s_t, a_t), \tag{12}$$

We aggregate per-step signals with generalized advantage estimation [Schulman et al., 2016] and peak-normalize the resulting advantages $\hat{A}_t$. Rollouts execute deterministic mean actions with planner substitution; all executed actions, including substitutions, enter a clipped importance-ratio update [Schulman et al., 2017]

$$\mathcal{L}_{\text{PG}} = -\mathbb{E}_t\Big[\min\big(\rho_t(\theta)\,\hat{A}_t,\ \text{clip}\big(\rho_t(\theta),\,1-\epsilon,\,1+\epsilon\big)\,\hat{A}_t\big)\Big], \quad \rho_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\text{old}}(a_t \mid s_t)}, \tag{13}$$

Pareto-admissible rollouts. We wrap every step in a *per-axis admissibility test*:

$$\text{accept } a_t^{\text{model}} \iff U_{\text{safe}}(s_t, a_t^{\text{model}}) \geq 0 \ \wedge\ U_{\text{prog}}(s_t, a_t^{\text{model}}) \geq -\eta_A. \tag{14}$$

Here $\eta_A > 0$ is the tolerance for progress regression. The test requires the checked execution waypoints to remain inside the eroded planning map $\mathcal{M}$ and prevents progress gains from compensating for a safety violation. Rejected actions are replaced by planner actions at the same state and retained as preference examples. Once the robot enters the training success radius, the planner controls the remaining steps to the endpoint and supplies further preference pairs.

Table 1: **Target real-world scenes and synthesized data.** Evaluation instructions are manually checked.

Scene	Area (m^2)	Scan (min)	Train eps.	Sim test	Real trials
Office	318	20	300	100	60
Cottage	155	13	300	100	40
Teaching	130	11	300	100	40
Apartment 1	90	9	300	100	30
Apartment 2	75	9	300	100	30
Total	768	62	1500	500	200

Pareto-weighted preference pairs. Each rejected a_t^{model} is reused, paired with the planner's a_t^{plan} at the same state, as a state-conditioned preference label in continuous trajectory space. We consume these pairs with a pairwise preference objective [Rafailov et al., 2023] using the rollout-time policy as reference π_{ref}. The per-pair weight w_t is determined by *vector* Pareto dominance of a_t^{plan} over a_t^{model}:

$$w_t = \begin{cases} w_{\text{high}}, & \mathbf{U}(s_t, a_t^{\text{plan}}) \succ_P \mathbf{U}(s_t, a_t^{\text{model}}) \text{ and } U_{\text{stop}}(s_t, a_t^{\text{plan}}) > 0, \\ w_{\text{mid}}, & \mathbf{U}(s_t, a_t^{\text{plan}}) \succ_P \mathbf{U}(s_t, a_t^{\text{model}}), \\ w_{\text{low}}, & \text{otherwise (admissibility-rejected, not strictly dominated)}, \end{cases} \tag{15}$$

with $w_{\text{high}} > w_{\text{mid}} > w_{\text{low}} > 0$. Combining w_t with the standard sigmoid log-ratio formulation,

$$\mathcal{L}_{\text{pref}} = -\mathbb{E}\Big[w_t \, \log \sigma\big(\beta(\Delta_\theta - \Delta_{\text{ref}})\big) \Big], \tag{16}$$

with β the temperature and Δ_θ, Δ_{ref} the policy and reference log-ratio gaps between a_t^{plan} and a_t^{model}, computed under the same Gaussian parameterization of Section 3.2.

Scan-anchored stabilization and unified objective. Three standard ingredients keep the post-training loop stable. A step-wise expert buffer of $(s, a^\star)$ transitions sampled along the planner's reference trajectories supplies a behavior-cloning anchor $\mathcal{L}_{\text{rep}}$. A short-to-long trajectory-length curriculum decouples horizon ramp-up from policy improvement. A non-learned cosine schedule on Σ controls action dispersion during post-training. The three Pareto read-outs jointly drive the unified post-training objective

$$\mathcal{L} = \lambda_{\text{PG}} \mathcal{L}_{\text{PG}} + \lambda_{\text{rep}} \mathcal{L}_{\text{rep}} + \lambda_{\text{pref}} \mathcal{L}_{\text{pref}}, \tag{17}$$

where $\lambda_{\text{PG}}, \lambda_{\text{rep}}, \lambda_{\text{pref}} > 0$ balance the three terms, $\mathcal{L}_{\text{PG}}$ is driven by the scalar projection of $\mathbf{U}$, $\mathcal{L}_{\text{rep}}$ anchors the policy to planner reference actions, and $\mathcal{L}_{\text{pref}}$ injects scene-grounded correction signals weighted by Pareto dominance.

4 Experiments

4.1 Setup

Scenes and data. We evaluate on five real-world scenes: an office, a cottage, a teaching building, and two apartments. Each is scanned once and reconstructed into a renderable Gaussian scene and a geometric BEV map. Scan2Instruction then synthesizes the three instruction types of Section 3.3 per scan, with separately sampled training and evaluation episodes; disjoint routes or regions within a scanned scene are not guaranteed. Real-robot trial counts vary by scene, as summarized in Table 1.

Baselines. We compare against one mapping baseline, **DualMap** Jiang et al. [2025], which consumes the scan plus depth and pose streams and plans through a queryable open-vocabulary semantic map, and five recent VLN/VLA models: **NaVILA** Cheng et al. [2024], **Uni-NaVid** Zhang et al. [2024], **StreamVLN** Wei et al. [2025], **CorrectNav** Yu et al. [2025], and **OmniNav** Xue et al. [2025]. All learning-based methods share the same RGB stream and low-level controller. We additionally report **RealNav**, our VLA pretrained on R2R-CE/RxR-CE, and demonstration-adapted variants **CorrectNav (human-demo-sft)** and **RealNav (human-demo-sft)**, fine-tuned on 20 teleoperated trajectories per scene.

Table 2: **Main evaluation on five scanned scenes.** Each cell reports success rate (SR) as *sim / real* on reconstructed Gaussian scenes and the physical robot under the same task split. Average SR is computed from pooled successes and trials across scenes.

Method	Office	Cottage	Teaching	Apartment 1	Apartment 2	Average
3D scene-graph mapping:						
DualMap Jiang et al. [2025]	0.32 / 0.17	0.22 / 0.13	0.17 / 0.10	0.35 / 0.17	0.33 / 0.17	0.28 / 0.15
Navigation VLA:						
NaVILA Cheng et al. [2024]	0.25 / 0.20	0.20 / 0.23	0.12 / 0.15	0.19 / 0.20	0.22 / 0.23	0.20 / 0.20
Uni-NaVid Zhang et al. [2024]	0.17 / 0.22	0.22 / 0.23	0.29 / 0.15	0.19 / 0.20	0.21 / 0.20	0.22 / 0.20
StreamVLN Wei et al. [2025]	0.26 / 0.17	0.26 / 0.13	0.10 / 0.13	0.23 / 0.30	0.30 / 0.23	0.23 / 0.18
OmniNav Xue et al. [2025]	0.26 / 0.22	0.19 / 0.20	0.19 / 0.23	0.27 / 0.27	0.25 / 0.20	0.23 / 0.22
CorrectNav Yu et al. [2025]	0.13 / 0.18	0.14 / 0.23	0.18 / 0.25	0.36 / 0.27	0.29 / 0.20	0.22 / 0.22
CorrectNav (human-demo-sft) Yu et al. [2025]	0.35 / 0.47	0.36 / 0.43	0.40 / 0.48	0.58 / 0.57	0.51 / 0.50	0.44 / 0.48
Ours:						
RealNav	0.22 / 0.25	0.24 / 0.28	0.17 / 0.23	0.22 / 0.23	0.30 / 0.23	0.23 / 0.25
RealNav (human-demo-sft)	0.37 / 0.48	0.34 / 0.45	0.42 / 0.45	0.56 / 0.57	0.49 / 0.53	0.44 / 0.49
RealNav + PRPT-KS	**0.79 / 0.78**	**0.93 / 0.95**	**0.81 / 0.83**	**0.85 / 0.87**	**0.87 / 0.87**	**0.85 / 0.85**

Metrics. We report success rate (SR) in both simulation and on the physical robot, using a $1.5\,\mathrm{m}$ evaluation radius. SPL, navigation error (NE), and oracle success (OS) additionally require ground-truth path length and pose, which are not reliably measurable under our real-robot interface; we therefore report them only in the ablation.

4.2 Simulation and real-world evaluation

Table 2 reports success rate on the reconstructed Gaussian scenes and the physical robot under the same instruction types and task split. Representative real-robot rollouts appear in Figure 1 (right). RealNav is competitive with the external baselines in simulation but shows no clear advantage, indicating that generic navigation pre-training alone is insufficient for a newly scanned environment. After PRPT-KS post-training, success rate increases sharply across all five scenes despite the scene-specific experience coming from a single passive scan rather than demonstrations or teleoperation. The gain transfers to physical deployment: even with the rendering-camera gap and execution drift, the post-trained policy preserves strong performance, suggesting that PRPT-KS teaches the policy to follow language instructions while favoring safe, target-approaching actions under diverse viewpoints. The two human-demo-sft models, **CorrectNav (human-demo-sft)** and **RealNav (human-demo-sft)**, fine-tune their respective base models on the 20 teleoperated trajectories per scene; both improve over their untuned counterparts but remain far below RealNav + PRPT-KS.

4.3 Generalization to unscanned regions of the same scenes

To test whether the policy memorizes the scanned region, we keep the same five environments but evaluate exclusively in regions that were *never scanned*: not reconstructed into the Gaussian scene, never in the synthesized task set, and never visited by RL rollouts. We run 20 real-robot trials per scene (100 in total) under the same instruction types. Table 3 reports real-robot SR in this setting. RealNav + PRPT-KS retains strong performance in these regions. These results suggest that the adapted policy generalizes beyond the scanned regions.

4.4 Ablations

Table 4 compares the full RealNav + PRPT-KS pipeline with the base RealNav policy before scene-specific post-training and with three loss-term ablations. *w/o behavior-cloning anchor* sets $\lambda_{\mathrm{rep}} = 0$, removing the planner-reference replay loss while retaining policy-gradient and preference updates. *w/o Pareto-utility advantage* sets $\lambda_{\mathrm{PG}} = 0$, disabling the policy-gradient term while retaining reference replay, preference supervision, and Pareto-admissible rollouts. *w/o Pareto-weighted preference* sets $\lambda_{\mathrm{pref}} = 0$, removing the entire weighted preference loss while retaining policy-gradient updates, reference replay, and rollout correction.

Table 3: Real-robot evaluation in *unscanned regions* of the same five scenes as Table 2: starts, goals, and intermediate waypoints all lie outside the scanned region used for reconstruction, task synthesis, and RL rollout. Each cell reports real-world success rate (SR).

Method	Office	Cottage	Teaching	Apartment 1	Apartment 2	Average
3D scene-graph mapping:						
DualMap Jiang et al. [2025]	0.10	0.15	0.20	0.10	0.15	0.14
Navigation VLA:						
NaVILA Cheng et al. [2024]	0.15	0.20	0.20	0.10	0.25	0.18
Uni-NaVid Zhang et al. [2024]	0.20	0.30	0.15	0.20	0.25	0.22
StreamVLN Wei et al. [2025]	0.25	0.20	0.30	0.20	0.15	0.22
OmniNav Xue et al. [2025]	0.30	0.25	0.20	0.35	0.20	0.26
CorrectNav Yu et al. [2025]	0.30	0.35	0.25	0.30	0.25	0.29
CorrectNav (human-demo-sft) Yu et al. [2025]	0.45	0.40	0.35	0.45	0.35	0.40
Ours:						
RealNav	0.25	0.30	0.20	0.20	0.15	0.22
RealNav (human-demo-sft)	0.45	0.40	0.35	0.45	0.40	0.41
RealNav + PRPT-KS	**0.80**	**0.85**	**0.80**	**0.85**	**0.80**	**0.82**

Table 4: **Ablation analysis in simulation.** We compare the full RealNav + PRPT-KS pipeline with RealNav before scene-specific post-training and with variants that individually disable one loss term in the unified post-training objective. Metrics are measured on reconstructed Gaussian scenes.

Variant	SR ↑	SPL ↑	NE (m) ↓	OS ↑
RealNav + PRPT-KS (Full)	**0.85**	**0.834**	1.42	**0.88**
RealNav (w/o PRPT-KS post-training)	0.23	0.210	3.50	0.27
w/o behavior-cloning anchor	0.30	0.293	3.49	0.41
w/o Pareto-utility advantage	0.81	0.785	**1.31**	0.85
w/o Pareto-weighted preference	0.80	0.781	1.32	0.83

Scene-specific post-training increases SR from 0.23 to 0.85. Removing the behavior-cloning anchor causes the largest degradation, reducing SR to 0.30 and SPL from 0.834 to 0.293. Removing Pareto-utility advantage reduces SR to 0.81, SPL to 0.785, and OS from 0.88 to 0.85; removing Pareto-weighted preference reduces these metrics to 0.80, 0.781, and 0.83, respectively. These results support the contribution of all three loss terms to navigation success and path efficiency under the evaluated configuration. However, the variants without the policy-gradient term or preference supervision achieve lower NE (1.31 m and 1.32 m) than the full model (1.42 m), so the improvements do not extend uniformly to terminal navigation error.

5 Limitations and Broader Impacts

Limitations and future work. Scanning still requires a human operator, and the geometry becomes stale when layouts change substantially, requiring a rescan. A natural extension is active, incremental mapping, where the robot detects divergence between reconstructed geometry and onboard observations and refreshes the representation autonomously.

Broader impacts. Replacing per-scene teleoperation with a single passive scan lowers the barrier to deploying language-guided robots in homes, offices, and small institutions. Two concerns warrant note: scans of private spaces may contain identifiable people or documents, so we release assets only after anonymization and require scanner consent from space owners; and stale geometry is a safety consideration in highly dynamic spaces, where rescans are recommended. The full pipeline runs locally without cloud upload, supporting privacy-sensitive deployments.

Acknowledgments and Disclosure of Funding

This work was supported by the National Natural Science Foundation of China (Grant No. 62136001) and the Beijing Natural Science Foundation (Grant No. L2608141). The authors declare no competing interests.

References

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sunderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, José M. M. Montiel, and Juan D. Tardós. ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021.

Devendra Singh Chaplot, Dhiraj Gandhi, Abhinav Gupta, and Ruslan Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. In *Advances in Neural Information Processing Systems*, 2020a.

Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural SLAM. In *International Conference on Learning Representations*, 2020b.

Danpeng Chen, Hai Li, Weicai Ye, Yifan Wang, Weijian Xie, Shangjin Zhai, Nan Wang, Haomin Liu, Hujun Bao, and Guofeng Zhang. PGSR: Planar-based gaussian splatting for efficient and high-fidelity surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 2024.

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.

Gunjan Chhablani, Xiaomeng Ye, Muhammad Zubair Irshad, and Zsolt Kira. Embodiedsplat: Personalized real-to-sim-to-real navigation with gaussian splats from a mobile device. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

David H. Douglas and Thomas K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 10(2):112–122, 1973.

Jiajun Jiang, Yiming Zhu, Zirui Wu, and Jie Song. Dualmap: Online open-vocabulary semantic mapping for natural language navigation in dynamic changing scenes. *IEEE Robotics and Automation Letters*, 10(12):12612–12619, 2025. doi: 10.1109/LRA.2025.3621942.

Gregory Kahn, Pieter Abbeel, and Sergey Levine. BADGR: An autonomous self-supervised learning-based navigation system. *IEEE Robotics and Automation Letters*, 2021a.

Gregory Kahn, Pieter Abbeel, and Sergey Levine. LaND: Learning to navigate from disengagements. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2021b.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023.

Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *Proceedings of the European Conference on Computer Vision*, 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 Conference on Empirical Methods in Natural Language Processing*, 2020.

Jiahang Liu, Yuanxing Duan, Jiazhao Zhang, Minghan Li, Shaoan Wang, Zhizheng Zhang, and He Wang. Navgsim: High-fidelity gaussian splatting simulator for large-scale navigation. *arXiv preprint arXiv:2603.15186*, 2026.

Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. Self-monitoring navigation agent via auxiliary progress estimation. In *International Conference on Learning Representations*, 2019.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, 2023.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2011.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

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. *arXiv preprint arXiv:2507.05240*, 2025.

Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. DD-PPO: Learning near-perfect PointGoal navigators from 2.5 billion frames. In *International Conference on Learning Representations*, 2020.

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*, 2025.

Xinda Xue, Junjun Hu, Minghua Luo, Shichao Xie, Jintao Chen, Zixun Xie, Kuichen Quan, Wei Guo, Mu Xu, and Zedong Chu. Omninav: A unified framework for prospective exploration and visual-language navigation. *arXiv preprint arXiv:2509.25687*, 2025.

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. *arXiv preprint arXiv:2508.10416*, 2025.

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*, 2024.

Shaoting Zhu, Linzhan Mou, Derun Li, Baijun Ye, Runhan Huang, and Hang Zhao. Vr-robo: A real-to-sim-to-real framework for visual robot navigation and locomotion. *IEEE Robotics and Automation Letters*, 2025.

Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J. Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2017.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes].

   Justification: The abstract and introduction state the single-scan, demonstration-free adaptation setting, the three contributions (Scan2Instruction, RealNav, PRPT-KS), and the evaluation scope on five real scenes.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes].

   Justification: The Limitations and Broader Impacts section discusses the human-operated scan, the assumption that the scene remains stable after scanning, and the need to rescan when the layout changes substantially.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A].

   Justification: The paper proposes an empirical system without formal theoretical results.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes].

   Justification: The method section specifies the policy parameterization, compiler outputs, post-training losses, and evaluation protocol. Implementation details are provided in the publicly released project.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes].

   Justification: The project has been publicly released. The paper uses public datasets (R2R-CE/RxR-CE) and describes the Scan2Instruction pipeline.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes].

   Justification: Section 4 describes the scenes, baselines, metrics, splits, and real-robot setup. Training configurations are provided in the publicly released project.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes].

   Justification: Each trial is an independent Bernoulli outcome with known n per cell (Table 1), with variability over independent trial outcomes under the fixed task split.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes].

Justification: The experiments used NVIDIA A100 GPUs, approximately 264 GPU-hours for navigation pretraining and approximately 6 GPU-hours per scene for PRPT-KS. Scans were captured using an iPhone 17 Pro with Stray Scanner.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes].

   Justification: The work conforms to the NeurIPS Code of Ethics.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes].

    Justification: The Limitations and Broader Impacts section discusses positive impacts (lower per-scene deployment cost, reduced on-site labor, applicability in homes and small institutions) and negative impacts (privacy in scanned spaces, stale-geometry safety risk), with concrete mitigations (anonymization of released scene assets, consent requirement, fully local pipeline, recommended rescans).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes].

    Justification: For the planned post-acceptance release, scene assets will be anonymized to remove identifiable people, documents, and personal items, and the usage guidelines will require scanners to obtain consent from space owners. The release does not include high-risk pretrained models or scraped image datasets requiring further safeguards.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes].

    Justification: All external models, datasets, simulators, and baselines are cited. Asset usage information is provided in the publicly released project.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes].

    Justification: New assets include reconstructed scene assets, generated Scan2Instruction episodes, post-training code, and evaluation splits. Scans were captured by the authors with consent from space owners. The publicly released project provides supporting documentation, and identifiable content is subject to the anonymization safeguards described in the paper.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A].

    Justification: No crowdsourcing or external subjects. The teleoperated trajectories and human-annotated instructions used in the experiments were produced internally by the authors.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB)

approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A].

Justification: No human-subject experiments.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes].

Justification: A multimodal LLM/VLM is used to generate grounded navigation instructions in Scan2Instruction (Section 3.3).