

Affordance-Conditioned Decision Making: Bridging the Semantic-Spatial Gap in Zero-Shot Cross-Floor Vision-and-Language Navigation

Xuekang Yang¹, Lu Chen¹, Shuang Luo¹, Jialing Zhu²

Qi Zhang^{3,*}, Yue Gao^{4,*}, Xiang Zhang^{3,*}

¹Department of Computer Science and Engineering, Shanghai Jiao Tong University

²Department of Automation, Shanghai Jiao Tong University

³Defense Innovation Institute, Academy of Military Sciences

⁴MoE Key Laboratory of Artificial Intelligence and AI Institute, Shanghai Jiao Tong University

*Corresponding authors

Abstract

Vision-and-language navigation increasingly relies on general-purpose semantic planners, yet translating correct high-level intent into reliable physical execution remains difficult in spatially constrained transitions. Reaching a staircase, doorway, or narrow passage does not ensure traversal; the agent must identify an executable affordance pose and recover from accumulated action errors. We propose PACE (Preference-refined **A**ffordance-**C**onditioned **E**xecution), a supervised local execution module that augments frozen zero-shot semantic planners for reliable cross-floor navigation. PACE grounds transition-related semantics into a long-horizon, agent-centric traversable affordance pose and conditions short-horizon action generation on this spatial target, thereby aligning semantic goals with physical execution. We further post-train PACE through failure-aware preference refinement using rollout-derived pairs that contrast normal or recovery behaviors with deviation-amplifying behaviors, thereby improving closed-loop correction. We integrate PACE into six open-source zero-shot VLN navigators and demonstrate consistent improvements on the cross-floor subsets of R2R-CE and RxR-CE, increasing the average success rate from 16.35% to 27.65% and from 4.76% to 12.06%, respectively. Real-world experiments further demonstrate PACE’s applicability in unseen environments, highlighting the potential of traversable affordances to bridge semantic intent and reliable embodied behavior.

1 Introduction

Vision-and-language navigation in continuous environments requires embodied agents to translate natural-language instructions into low-level motion (Krantz et al., 2020). Recent zero-shot and learning-based methods, increasingly driven by general-purpose semantic planners, have made substantial progress (Zheng et al., 2026c; Chen et al., 2025b; He et al., 2026; Wei et al., 2025; Chu et al., 2026; Zhang et al., 2026). Nevertheless, multi-floor environments expose a persistent challenge: correct semantic intent does not ensure reliable execution at constrained transitions.

Existing cross-floor navigation methods primarily rely on explicit geometric representations, including hierarchical scene graphs (Werby et al., 2024), floor-aware maps (Zhang et al., 2025b; Gong et al., 2026; Huang et al., 2026), and reconstructed traversable surfaces (Li et al., 2026; Zheng et al., 2026a), to model inter-floor connectivity and support geometric planning. However, stair traversal remains a major source of failure: nearly 21% of navigation failures have been associated with unsuccessful stair traversal (Ramrakhya et al., 2023), while MANSION reports substantial degradation on

multi-floor long-horizon tasks (Che et al., 2026). These findings indicate that geometry can describe multi-floor structure and localize transition regions, but reaching a staircase does not guarantee successful traversal.

Recent work has characterized a related challenge as the *semantic-geometric gap*, namely grounding semantic concepts in environment-centric 3D structures (Li et al., 2026). Despite strong semantic perception, VLMs remain limited in reasoning over 3D structures and action-induced spatial transitions (Yang et al., 2025; Chen et al., 2025c; Windecker et al., 2025; Kong et al., 2025). The semantic-geometric gap specifies *where* a relevant structure is, whereas reliable navigation additionally requires an agent-centric executable affordance pose specifying *where* and *how* the agent should approach and traverse it. We term this remaining disconnect the *semantic-spatial gap*. These results show that geometric localization of a staircase does not ensure its successful traversal.

Existing methods address this gap through two main paradigms. Hierarchical methods insert waypoint-based spatial subgoals between semantic reasoning and motion execution (An et al., 2024; Qiao et al., 2025; Shi et al., 2025). However, these waypoints encode

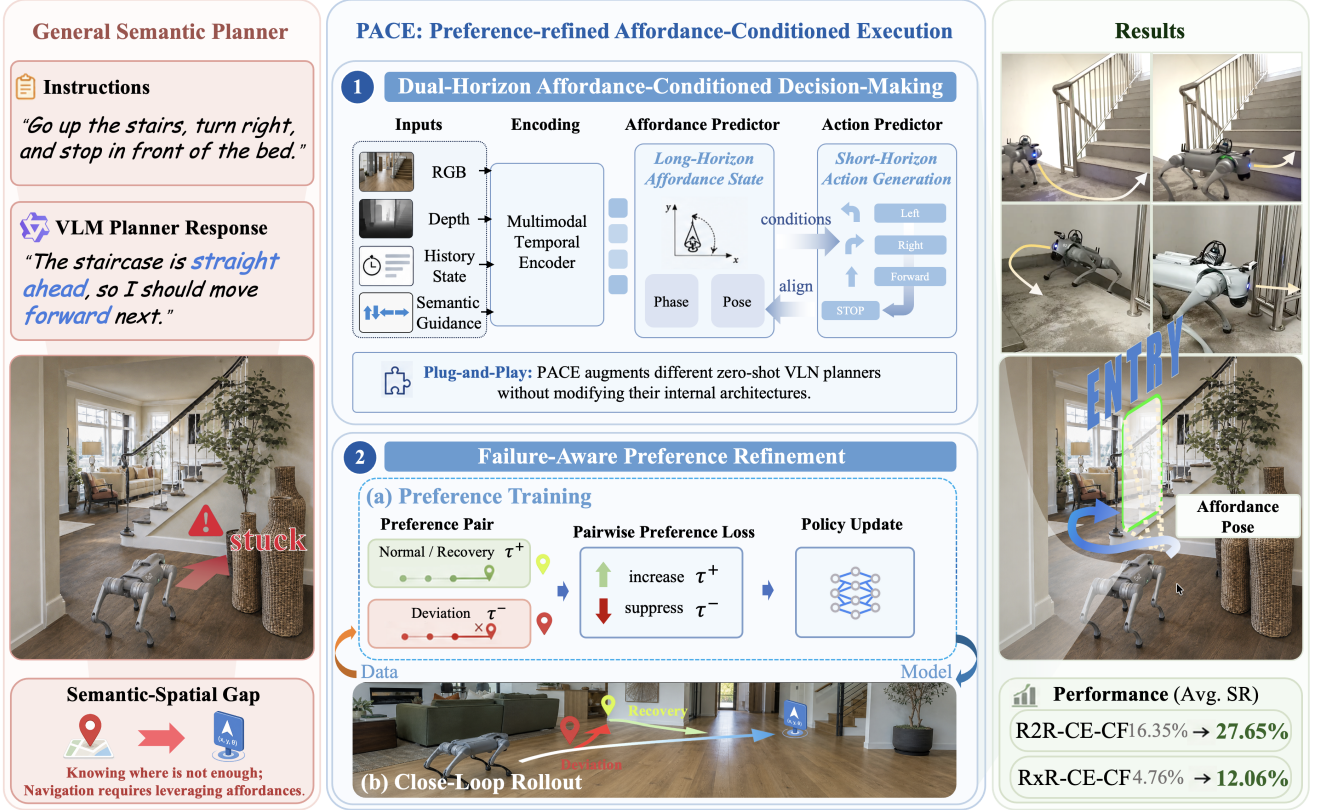


Figure 1: **Overview of PACE.** PACE bridges the semantic-spatial gap through traversable affordance estimation, affordance-conditioned action generation, and rollout-derived preference refinement.

where the agent *can* move rather than the instruction-conditioned pose from which a transition *should* be executed, while external planners provide limited semantic conditioning during execution (Wang et al., 2026a; Li et al., 2026). End-to-end imitation-learning policies instead map visual-language observations directly to actions (Wei et al., 2025; Chu et al., 2026; Zhang et al., 2024; Krantz et al., 2020), but expert-state-dominated supervision leaves them vulnerable to compounding errors and limits recovery from policy-induced deviations (Ross et al., 2011; Shao et al., 2024). Thus, neither paradigm jointly provides task-conditioned executable spatial grounding and robust closed-loop execution beyond the expert-state distribution.

To address these limitations, we propose PACE (Preference-refined Affordance-Conditioned Execution) in Fig. 1, a plug-and-play supervised local execution framework trained on the corresponding benchmark training split to augment frozen zero-shot semantic planners for cross-floor VLN; both components remain frozen in unseen evaluation environments. Our contributions are:

- We introduce PACE, a dual-horizon affordance-conditioned framework that grounds stair-transition semantics into long-horizon traversable affordance poses for conditioning short-horizon action generation.
- We propose failure-aware preference refinement using rollout-derived pairs, favoring normal and

recovery behaviors over deviation-amplifying alternatives to improve closed-loop correction beyond expert demonstrations.

- Across six open-source zero-shot VLN navigators, PACE improves average SR by 11.30 and 7.30 percentage points on R2R-CE-CF and RxR-CE-CF, respectively, and is successfully deployed in unseen real-world environments.

2 Related Work

2.1 Semantic-Spatial Grounding

Despite strong semantic understanding, multimodal large language models (MLLMs) remain unreliable in spatial reasoning (Yang et al., 2025; Chen et al., 2025c; Windecker et al., 2025; Kong et al., 2025). Embodied navigation methods therefore combine vision-language and segmentation models with depth and SLAM to associate semantics with 3D structures represented by BEV or voxel maps, 3D Gaussians, and layered memories (Yu et al., 2026; Jang and Kim, 2026; Zhou et al., 2026; Yokoyama et al., 2024; Long et al., 2025; Zhang et al., 2025a; Li et al., 2026; Wang et al., 2026b; Zheng et al., 2026b).

However, these environment-centric representations primarily localize semantic entities rather than model agent-centric executability. Affordance-aware methods capture functional agent-environment interactions but

often depend on predefined skills or external intermediate representations (ichter et al., 2023; Yuan et al., 2024; Chen et al., 2025a). In contrast, PACE directly predicts agent-centric traversable affordance poses from RGB-D observations and semantic guidance without explicit 3D scene construction.

2.2 Semantic-Aware Navigation

Semantic-aware VLN methods mainly follow end-to-end and hierarchical paradigms. End-to-end methods map visual-language observations to decisions through task-specific policy learning (Wei et al., 2025; Chu et al., 2026; Zhang et al., 2024; Krantz et al., 2020) or zero-shot LLM/VLM reasoning (Zhou et al., 2024; Bhatt et al., 2025; Long et al., 2024), whereas hierarchical methods insert waypoints or topological nodes between semantic reasoning and motion execution (An et al., 2024; Qiao et al., 2025; Shi et al., 2025).

However, end-to-end policies learn semantic-to-action mappings implicitly, zero-shot methods rely on unreliable long-horizon spatial reasoning, and hierarchical subgoals typically encode geometric accessibility rather than instruction-conditioned affordances (Li et al., 2026; Wang et al., 2026a). PACE instead conditions local action generation on an inferred affordance pose, explicitly grounding task semantics into executable spatial decisions.

3 Methods

Problem Formulation Given an instruction I and an unseen continuous environment $\mathcal{E}$, the agent receives an egocentric RGB-D observation $o_t = (r_t, d_t)$ at step t and selects an action from

$$\mathcal{A} = \{\text{FORWARD}, \text{LEFT}, \text{RIGHT}\}, \quad (1)$$

where r_t and d_t denote the RGB and depth observations, respectively. **FORWARD** advances the agent by 0.25 m, while **LEFT** and **RIGHT** rotate it by 30°. We focus on cross-floor episodes requiring stair traversal.

The high-level VLM provides traversal guidance $s_t \in \{\text{UP}, \text{DOWN}, \text{UNKNOWN}\}$. PACE operates on a K -step context $c_t = (o_{t-K+1:t}, \xi_{t-K+1:t}, s_t)$, where $\xi_i = (a_{i-1}, \hat{p}_{i-1}^{\text{aff}}, \hat{\phi}_{i-1})$ contains the preceding action, affordance pose, and traversal phase. PACE predicts

$$\begin{aligned} h_t^{\text{aff}} &= f_{\text{aff}}(c_t), & \hat{z}_t &= (\hat{p}_t^{\text{aff}}, \hat{\phi}_t), \\ \hat{p}_t^{\text{aff}} &= (\hat{x}_t, \hat{y}_t, \hat{\theta}_t), \end{aligned} \quad (2)$$

where $(\hat{x}_t, \hat{y}_t, \hat{\theta}_t)$ denotes the agent-centric traversable affordance pose—the desired entry pose before traversal and exit pose once traversal begins—and $\hat{\phi}_t \in \{\text{APPROACH}, \text{ENTRY}, \text{TRAVERSE}, \text{EXIT}\}$ denotes the traversal phase.

PACE then generates a short-horizon action proposal

$$\hat{\tau}_t = f_{\text{act}}(c_t, h_t^{\text{aff}}) = (\hat{a}_{t,1}, \dots, \hat{a}_{t,M_t}, \text{STOP}), \quad (3)$$

where $\hat{a}_{t,j} \in \mathcal{A}$ for $j \leq M_t$, $\hat{a}_{t,M_t+1} \equiv \text{STOP}$, and M_t counts the motion actions preceding the terminal token; hence $|\hat{\tau}_t| = M_t + 1$. The **STOP** token terminates only the current proposal and triggers the next closed-loop decision rather than ending the navigation episode. Throughout this paper, zero-shot refers to the high-level navigator and test-time protocol: PACE is supervised on the corresponding training split, while both PACE and the navigator remain frozen without supervision or adaptation in the unseen evaluation environments.

3.1 Framework Overview

As shown in Fig. 1, PACE combines dual-horizon affordance-conditioned decision making with failure-aware preference refinement. Imitation learning couples long-horizon affordance estimation with short-horizon action generation, while preference post-training on rollout-derived deviation and recovery trajectories improves closed-loop correction. At inference, the frozen module augments existing VLM planners for local stair traversal.

3.2 Dual-Horizon Affordance-Conditioned Execution

Modeling. For compact notation, let $z_t = (p_t^{\text{aff}}, \phi_t)$ denote the long-horizon affordance state, where $p_t^{\text{aff}} = (x_t, y_t, \theta_t)$ is an agent-centric traversable affordance pose and ϕ_t is the current traversal phase. PACE first extracts an affordance-aware latent representation:

$$h_t^{\text{aff}} = f_{\text{aff}}(c_t; \omega), \quad (4)$$

where ω denotes the parameters of the multimodal context encoder and long-horizon affordance estimator. PACE parameterizes the dual-horizon policy as

$$p_{\Theta}(z_t, \tau_t | c_t) = p_{\omega}(z_t | h_t^{\text{aff}}) p_{\psi}(\tau_t | c_t, h_t^{\text{aff}}), \quad (5)$$

where ψ denotes the parameters of the action generator and $\tau_t = (a_{t,1}, \dots, a_{t,M_t}, \text{STOP})$ denotes the short-horizon action sequence, with $a_{t,j} \in \mathcal{A}$ for $j \leq M_t$ and $a_{t,M_t+1} \equiv \text{STOP}$.

The long-horizon affordance state is decoded through parallel pose and phase heads:

$$\begin{aligned} \hat{p}_t^{\text{aff}} &= f_{\text{pose}}(h_t^{\text{aff}}; \omega), \\ p_{\omega}(\phi | h_t^{\text{aff}}) &= f_{\text{phase}}(h_t^{\text{aff}}; \omega), \\ \hat{\phi}_t &= \arg \max_{\phi} p_{\omega}(\phi | h_t^{\text{aff}}). \end{aligned} \quad (6)$$

yielding $\hat{z}_t = (\hat{p}_t^{\text{aff}}, \hat{\phi}_t)$. Conditioned on h_t^{aff} , the action generator autoregressively models

$$p_{\psi}(\tau_t | c_t, h_t^{\text{aff}}) = \prod_{j=1}^{|\tau_t|} p_{\psi}(a_{t,j} | a_{t,<j}, c_t, h_t^{\text{aff}}). \quad (7)$$

Here $a_{t,<j} = (a_{t,1}, \dots, a_{t,j-1})$ denotes the preceding action prefix. Expert-path height trends delimit the four traversal phases: $p_t^{\text{aff},*}$ targets the entry pose during **APPROACH** and the exit pose during **ENTRY/TRAVERSE**,

while EXIT is terminal. The selected pose is projected into the current agent frame using its horizontal displacement and tangent heading.

Given an expert dataset $\mathcal{D} = \{(c_t, p_t^{\text{aff}*}, \phi_t^*, \tau_t^*)\}$ constructed from expert stair-traversal trajectories, where $p_t^{\text{aff}*}$ is the phase-dependent entry or exit pose, ϕ_t^* is the traversal-phase label, and τ_t^* is the expert short-horizon action sequence, we define

$$\mathcal{L}_{\text{dual}}(\Theta; \mathcal{D}) = \mathbb{E}_{\mathcal{D}} \left[\begin{aligned} & \underbrace{\lambda_p \mathcal{L}_{\text{pose}}(\hat{p}_t^{\text{aff}}, p_t^{\text{aff}*})}_{(1) \text{ Affordance pose regression}} \\ & - \underbrace{\lambda_\phi \log p_\omega(\phi_t^* | h_t^{\text{aff}})}_{(2) \text{ Traversal phase classification}} \\ & - \underbrace{\lambda_a \sum_{j=1}^{|\tau_t^*|} \log p_\psi(a_{t,j}^* | a_{t,<j}^*, c_t, h_t^{\text{aff}})}_{(3) \text{ Action sequence classification}} \end{aligned} \right]. \quad (8)$$

The supervised initialization is obtained by

$$\Theta^{(0)} = \arg \min_{\Theta} \mathcal{L}_{\text{dual}}(\Theta; \mathcal{D}),$$

where $\Theta = \{\omega, \psi\}$ denotes all trainable parameters, and λ_p , λ_ϕ , and λ_a balance the three supervision terms. Terms (1)–(3) correspond to traversable affordance pose regression, traversal phase classification, and autoregressive action prediction, respectively. Conditioning action generation on h_t^{aff} couples long-horizon affordance grounding with short-horizon physical execution.

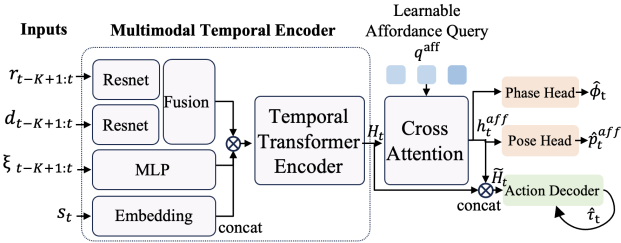


Figure 2: Model Architecture.

Network Architecture. As illustrated in Fig. 2, PACE comprises a multimodal temporal encoder, a query-based affordance estimator, and a Transformer-based causal action decoder. RGB observations $r_{t-K+1:t}$ and depth observations $d_{t-K+1:t}$ are encoded by separate ImageNet-pretrained ResNet backbones and projected into a shared d -dimensional feature space, where a gated fusion module integrates the two visual streams. The historical context $\xi_{t-K+1:t}$ is encoded by an MLP, while the semantic guidance s_t is represented by a learnable embedding. The resulting visual, historical, and semantic features are concatenated into one multimodal token per context step. After adding temporal positional embeddings, a Transformer encoder models dependencies across the K -step context and produces $H_t \in \mathbb{R}^{K \times d}$, where d denotes the hidden dimension.

A learnable affordance query $q^{\text{aff}} \in \mathbb{R}^d$ performs multi-head cross-attention over H_t to extract

$$h_t^{\text{aff}} = \text{MHA}(q^{\text{aff}}, H_t, H_t) \in \mathbb{R}^d. \quad (9)$$

Parallel phase and pose heads decode h_t^{aff} into the traversal-phase prediction $\hat{\phi}_t$ and the traversable affordance pose $\hat{p}_t^{\text{aff}}$, respectively.

For short-horizon action generation, the affordance representation is appended to the temporal features as an additional memory token:

$$\tilde{H}_t = [H_t; h_t^{\text{aff}}] \in \mathbb{R}^{(K+1) \times d}. \quad (10)$$

A causal Transformer decoder parameterized by ψ attends to $\tilde{H}_t$ and autoregressively predicts each action conditioned on the preceding action prefix $a_{t,<j}$. The decoder is trained with teacher forcing, while beam search generates candidate sequences at inference until **STOP** or the maximum proposal length is reached. The predicted affordance pose is further used for geometric reranking of the candidate sequences.

3.3 Failure-Aware Preference Refinement

Imitation learning exposes PACE primarily to expert states, limiting its ability to recover from policy-induced deviations during closed-loop execution. We therefore refine the policy by favoring normal or recovery behaviors over deviation-amplifying alternatives.

Preference Pair Construction. We collect model-induced states through autoregressive closed-loop rollouts of the current PACE policy. A deviation is identified when the positional or heading error from the expert route exceeds 0.5 m or 30° for two consecutive steps, whereas failure is triggered immediately when either error exceeds 1.5 m or 60°. From each failure point, A* plans a trajectory back to the expert route, which is converted into discrete expert actions and replayed under teacher forcing. Recovery ends when the positional and heading errors remain within 0.25 m and 30° for two consecutive steps, after which subsequent states are treated as normal behavior. We form fixed-length contiguous same-role snippets within each rollout and match each deviation-amplifying snippet ζ^- with shuffled normal and recovery snippets ζ^+ from the global pools.

Preference Optimization. To distinguish a temporal preference snippet from the short-horizon proposal τ_t , we denote an ordered snippet as

$$\zeta = ((c_i, \tau_i))_{i=1}^L, \quad \tau_i = (a_{i,1}, \dots, a_{i,M_i}, \text{STOP}), \quad (11)$$

where L is the snippet length, M_i counts the motion actions preceding **STOP**, and $|\tau_i| = M_i + 1$ includes the terminal token. Given the affordance-aware representation $h_i^{\text{aff}} = f_{\text{aff}}(c_i; \omega)$, each proposal is scored by its length-normalized conditional log-likelihood:

$$\ell_{\Theta}(c_i, \tau_i) = \frac{1}{|\tau_i|} \sum_{j=1}^{|\tau_i|} \log p_{\psi}(a_{i,j} | a_{i,<j}, c_i, h_i^{\text{aff}}), \quad (12)$$

where $\Theta = \{\omega, \psi\}$. All proposals are rescored under teacher forcing, with $a_{i, < j}$ denoting the recorded action prefix. Recovery snippets contain A*-derived expert actions, whereas deviation snippets contain model-generated actions collected during autoregressive rollouts. Importantly, the scores are computed using model-inferred rather than ground-truth affordance representations, matching closed-loop inference. Length normalization prevents a bias toward shorter proposals.

We aggregate the proposal scores over the temporal snippet:

$$S_{\Theta}(\zeta) = \frac{1}{L} \sum_{i=1}^L \ell_{\Theta}(c_i, \tau_i). \quad (13)$$

Temporal aggregation captures whether consecutive decisions amplify or reduce navigation errors, which cannot be reliably characterized by isolated actions. Given a cross-rollout preference pair (ζ^+, ζ^-) , we define

$$\begin{aligned} \Delta_{\Theta} &= S_{\Theta}(\zeta^+) - S_{\Theta}(\zeta^-), \\ P_{\Theta}(\zeta^+ \succ \zeta^-) &= \sigma(\beta \Delta_{\Theta}). \end{aligned} \quad (14)$$

where ζ^+ denotes a normal or recovery snippet, ζ^- denotes a deviation snippet, and β controls the preference sharpness. The pairwise preference loss is

$$\mathcal{L}_{\text{pref}}(\Theta; \mathcal{D}_{\text{pref}}) = -\mathbb{E}_{(\zeta^+, \zeta^-) \sim \mathcal{D}_{\text{pref}}} [\log \sigma(\beta \Delta_{\Theta})]. \quad (15)$$

The post-training objective is

$$\begin{aligned} \mathcal{L}_{\text{post}} &= \frac{1}{2} \mathcal{L}_{\text{dual}}(\Theta; \mathcal{D}_{\text{exp}}) \\ &\quad + \frac{1}{2} \mathcal{L}_{\text{dual}}(\Theta; \mathcal{D}_{\text{on}}) \\ &\quad + \lambda_{\text{pref}} \mathcal{L}_{\text{pref}}(\Theta; \mathcal{D}_{\text{pref}}). \end{aligned}$$

Equal-sized expert and on-policy batches preserve the original affordance-estimation and action-generation capabilities while adapting PACE to model-induced states, whereas the preference term favors normal and recovery behaviors over deviation-amplifying alternatives.

3.4 Training, Inference, and Plug-and-Play Integration

Algorithm 1 summarizes PACE training and receding-horizon inference. Supervised initialization is followed by iterative refinement using labeled on-policy states and rollout-derived preference pairs.

At inference, beam search retains up to B candidate short-horizon action sequences:

$$\mathcal{B}_t = \text{BeamSearch}_B [p_{\psi}(\tau_t | c_t, h_t^{\text{aff}})], \quad |\mathcal{B}_t| \leq B. \quad (16)$$

The candidates are rolled out with known motion primitives and reranked by their likelihood and geometric consistency with the predicted affordance pose:

$$\begin{aligned} J_t(\tau) &= \log p_{\psi}(\tau | c_t, h_t^{\text{aff}}) - \lambda_g G(\tau, \hat{p}_t^{\text{aff}}), \\ \hat{\tau}_t &= \arg \max_{\tau \in \mathcal{B}_t} J_t(\tau), \end{aligned} \quad (17)$$

Here $(r_{xy}, r_{\theta}) = \text{Transform}(\hat{p}_t^{\text{aff}}; \tau)$ is the terminal residual after executing τ , $G = \|r_{xy}\|_2 + \lambda_{\theta} |\text{wrap}(r_{\theta})| +$

Algorithm 1 PACE Training and Inference

Require: Expert data $\mathcal{D}_{\text{exp}}$, refinement rounds R , beam width B

- 1: $\Theta^{(0)} \leftarrow \arg \min_{\Theta} \mathcal{L}_{\text{dual}}(\Theta; \mathcal{D}_{\text{exp}})$ ▷ Imitation learning
 - 2: **for** $r = 1, \dots, R$ **do**
 - 3: $\mathcal{T}^{(r)} \leftarrow \text{Rollout}(\pi_{\Theta^{(r-1)}})$
 - 4: $\mathcal{D}_{\text{on}}^{(r)} \leftarrow \text{LabelOnPolicy}(\mathcal{T}^{(r)})$
 - 5: $(\zeta^+, \zeta^-)^{(r)} \leftarrow \text{BuildPreferencePairs}(\mathcal{T}^{(r)})$
 - 6: $\mathcal{D}_{\text{pref}}^{(r)} \leftarrow \{(\zeta^+, \zeta^-)^{(r)}\}$
 - 7: $\Theta^{(r)} \leftarrow \arg \min_{\Theta} \mathcal{L}_{\text{post}}(\Theta; \mathcal{D}_{\text{exp}}, \mathcal{D}_{\text{on}}^{(r)}, \mathcal{D}_{\text{pref}}^{(r)})$
 - 8: **end for**
 - Inference at step t :**
 - 9: $\hat{z}_t \leftarrow (\hat{p}_t^{\text{aff}}, \hat{\phi}_t)$ ▷ Affordance estimation
 - 10: $\mathcal{B}_t \leftarrow \text{BeamSearch}_B(p_{\psi}(\tau_t | c_t, h_t^{\text{aff}}))$
 - 11: $\hat{\tau}_t \leftarrow \arg \max_{\tau \in \mathcal{B}_t} J_t(\tau)$ ▷ Geometry-aware selection
 - 12: Execute $\hat{a}_{t,1}$ and replan
-

$\lambda_s \mathbf{1}[\text{STOP} \notin \tau]$, and $\log p_{\psi}(\tau | c_t, h_t^{\text{aff}})$ is the cumulative sequence log-likelihood.

PACE interfaces with heterogeneous VLN agents through unified RGB-D, semantic-guidance, and action interfaces, without accessing internal representations or requiring retraining. GroundingDINO detects visible staircases, while the VLM assesses their relevance and provides directional guidance. PACE retains control until the phase head predicts EXIT or a safety/budget limit is reached, then returns control to the navigator and may be reactivated if traversal remains incomplete.

4 Experiments

We evaluate PACE in terms of: (1) long-horizon affordance and short-horizon action prediction; (2) improvements to zero-shot VLN baselines; (3) real-world cross-floor navigation; and (4) standalone stair traversal and ablations of affordance conditioning and preference refinement.

4.1 Experimental Setup

Datasets. We evaluate on the cross-floor subsets of R2R-CE (Krantz et al., 2020) and RxR-CE(en-US) (Gao et al., 2025; Ku et al., 2020), denoted R2R-CE-CF and RxR-CE-CF, which retain episodes with start-to-goal vertical displacement $|h_s - h_g| > 1$ m. R2R-CE-CF contains 925 training and 261 val-unseen episodes, while the en-US subset of RxR-CE-CF contains 932 and 217, respectively. PACE is trained on the corresponding training splits, and all baselines are evaluated on the full val-unseen splits.

Baselines and Metrics. We integrate PACE with six recent open-source zero-shot VLN systems. VLN-Zero uses cache-enabled neurosymbolic planning (Bhatt et al., 2025), whereas CA-Nav performs constraint-aware value mapping (Chen et al., 2025b). 3-step-Nav adopts global-local planning with trajectory auditing (Zheng et al., 2026c), while Open-Nav employs open-source LLM reasoning (Qiao et al., 2025). STRIDER

Method	R2R-CE-CF				RxR-CE-CF			
	SR (%) $\uparrow$	OSR (%) $\uparrow$	SPL (%) $\uparrow$	NDTW $\uparrow$	SR (%) $\uparrow$	OSR (%) $\uparrow$	SPL (%) $\uparrow$	NDTW $\uparrow$
	Alone / +PACE	Alone / +PACE	Alone / +PACE	Alone / +PACE	Alone / +PACE	Alone / +PACE	Alone / +PACE	Alone / +PACE
VLN-Zero	6.90 / 23.75$\blacktriangle$	10.34 / 30.27$\blacktriangle$	6.00 / 18.45$\blacktriangle$	0.24 / 0.46$\blacktriangle$	3.23 / 7.37$\blacktriangle$	4.61 / 8.76$\blacktriangle$	2.14 / 4.87$\blacktriangle$	0.18 / 0.22$\blacktriangle$
3-step-Nav	6.90 / 19.54$\blacktriangle$	11.49 / 36.02$\blacktriangle$	5.95 / 12.53$\blacktriangle$	0.38 / 0.45$\blacktriangle$	5.99 / 10.14$\blacktriangle$	5.99 / 20.28$\blacktriangle$	4.81 / 3.65 Δ	0.37 / 0.32 $\blacktriangledown$
CA-Nav	14.94 / 25.67$\blacktriangle$	27.97 / 56.70$\blacktriangle$	7.15 / 11.84$\blacktriangle$	0.17 / 0.23$\blacktriangle$	4.15 / 11.98$\blacktriangle$	11.98 / 25.35$\blacktriangle$	1.54 / 4.95$\blacktriangle$	0.17 / 0.20$\blacktriangle$
STRIDER	17.24 / 21.84Δ	21.84 / 29.50$\blacktriangle$	15.85 / 17.53Δ	0.47 / 0.51Δ	2.30 / 8.29$\blacktriangle$	4.15 / 11.98$\blacktriangle$	1.80 / 6.64$\blacktriangle$	0.35 / 0.38$\blacktriangle$
Open-Nav	20.31 / 24.90Δ	24.52 / 37.55$\blacktriangle$	18.43 / 19.99Δ	0.52 / 0.57$\blacktriangle$	3.23 / 6.91Δ	3.69 / 9.68$\blacktriangle$	2.84 / 5.76Δ	0.34 / 0.40$\blacktriangle$
HSGM	31.80 / 50.19$\blacktriangle$	42.91 / 63.22$\blacktriangle$	20.87 / 35.93$\blacktriangle$	0.29 / 0.34$\blacktriangle$	9.68 / 27.65$\blacktriangle$	13.82 / 37.79$\blacktriangle$	6.66 / 19.19$\blacktriangle$	0.38 / 0.46$\blacktriangle$

Table 1: Cross-floor navigation results on the val-unseen splits of R2R-CE-CF and RxR-CE-CF. Each pair reports Alone/+PACE, with the higher value in bold. For the 95% paired-bootstrap CI of $\Delta = +PACE - \text{Alone}$, $\blacktriangle$, Δ , and $\blacktriangledown$ denote robust improvement (CI > 0), inconclusive difference (CI contains zero), and robust decrease (CI < 0), respectively.

optimizes structured waypoint decisions (He et al., 2026), whereas HSGM plans over hierarchical semantic-geometric maps (Li et al., 2026). Since prior works do not separately report cross-floor results, we rerun all released baselines under a unified protocol using the same high-level VLM, both alone and with PACE.

For overall cross-floor VLN, we report Success Rate (SR, within 3 m of the goal), Oracle Success Rate (OSR), Success weighted by Path Length (SPL), and Normalized Dynamic Time Warping (NDTW). For local stair traversal, we report OSR, SPL, NDTW, and Collision Rate (CR).

Implementation Details. We locally deploy the open-source Qwen-3.6-27B-FP8 (Qwen Team, 2026) as the shared high-level VLM for cost-efficient evaluation. PACE uses five-step contexts and ImageNet-pretrained dual-stream ResNet-18 backbones. Supervised training runs for 15 epochs with AdamW, batch size 32, an initial learning rate of 10^{-4} , and cosine decay. Post-training performs 20 on-policy rounds at 10^{-5} , collecting 2,048 rollouts per round and mixing equal-sized expert and on-policy batches. Rollout starts combine recorded expert states and GroundingDINO randomized approach states at a ratio of 1:8. Preference learning uses five-step snippets, $\lambda_{\text{pref}} = 0.1$, and $\beta = 1.0$. Inference uses beam size 5 and a maximum proposal length of 48. Training takes approximately 19 hours on four NVIDIA RTX 5090 GPUs. Real-world experiments use a Unitree Go2 Edu equipped with a RealSense D435i and a Livox Mid-360, with PACE inference running on an RTX 3070 Ti laptop.

4.2 Main Results

PACE Prediction Performance. Figure 3 evaluates the standalone predictive performance of PACE over both horizons through policy rollouts on the R2R-CE-CF val-unseen split. For the long-horizon affordance state, pose errors are concentrated near the origin, with a mean positional error of 0.329 m and a yaw MAE of 9.9° , indicating that PACE provides a reasonably accurate spatial target for local execution. Phase prediction achieves an accuracy of 96.94%, although 27.7% of ENTRY states are misclassified as APPROACH, revealing ambiguity at the boundary between approaching and

Method	R2R-CE-CF		RxR-CE-CF	
	Takeover Rate (%)	SR (%) $\uparrow$	Takeover Rate (%)	SR (%) $\uparrow$
	Alone / +PACE	Alone / +PACE	Alone / +PACE	Alone / +PACE
VLN-Zero	39.08	7.84 / 51.96$\blacktriangle$	11.06	8.33 / 41.67$\blacktriangle$
3-step-Nav	51.72	10.37 / 37.04$\blacktriangle$	42.86	10.75 / 20.43$\blacktriangle$
CA-Nav	65.52	19.30 / 36.26$\blacktriangle$	35.48	6.49 / 28.57$\blacktriangle$
STRIDER	27.59	19.44 / 47.22$\blacktriangle$	17.05	5.41 / 24.32$\blacktriangle$
Open-Nav	38.31	27.00 / 41.00$\blacktriangle$	28.11	8.20 / 24.59$\blacktriangle$
HSGM	76.63	34.50 / 63.50$\blacktriangle$	51.15	11.71 / 52.25$\blacktriangle$

Table 2: Takeover-subset SR, comparing Alone/+PACE on aligned episode IDs. Takeover Rate is the fraction of episodes with actual PACE takeover; $\blacktriangle$ denotes a paired-bootstrap 95% CI above zero.

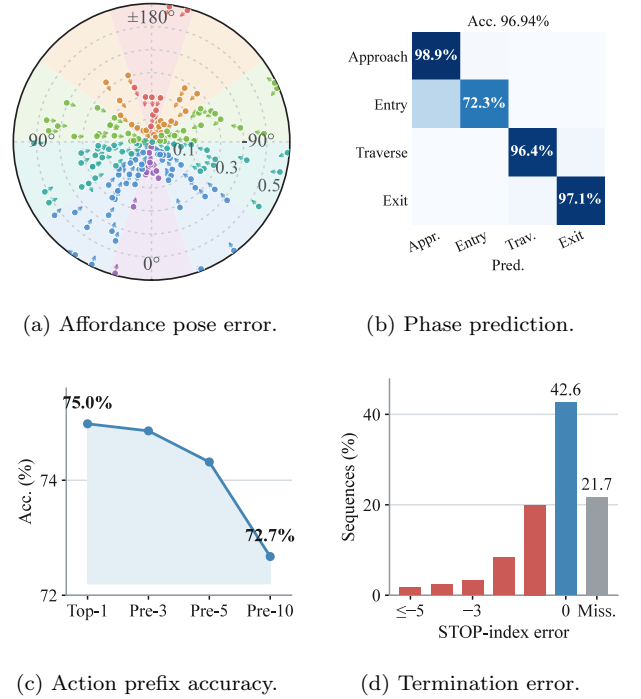


Figure 3: Standalone PACE predictions on R2R-CE-CF val-unseen. In (a), the center denotes zero positional error; radius and angle encode its magnitude and direction, while arrow deviation from the inward direction represents heading error.

entering the stairs. For short-horizon action generation, prefix accuracy decreases only modestly from 74.98% for the first action to 72.67% over the first ten actions, suggesting limited autoregressive error accumulation under rollout evaluation. Moreover, 42.6% of STOP predictions are exact, and 71.0% occur no more than two steps early. Overall, these rollout results demonstrate reliable affordance grounding and stable action generation under policy-induced states, consistent with the robustness objective of failure-aware preference refinement.

Quantitative Evaluation on VLN Benchmarks.

Table 1 shows that standalone zero-shot VLN methods remain limited in cross-floor scenarios, with the best SR reaching only 31.80% on R2R-CE-CF and 9.68% on RxR-CE-CF. With PACE, all six methods improve across every metric on R2R-CE-CF, with average gains of 11.30, 19.03, and 7.01 percentage points in SR, OSR, and SPL, respectively, and 0.082 in NDTW. On RxR-CE-CF, PACE improves SR and OSR for all methods and improves SPL and NDTW for five of six. HSGM+PACE achieves the highest SR, OSR, and SPL on both benchmarks.

On episodes with actual PACE takeover, Table 2 shows SR gains of 9.68–44.12 percentage points across all navigator–benchmark pairs, with all paired-bootstrap 95% confidence intervals above zero. However, non-random takeover and stochastic VLM sampling make this analysis associative rather than causal. Across 10,000 paired-bootstrap resamples of the full aligned results, PACE yields 40/48 robust improvements, seven inconclusive differences, and one robust decrease. Gain variations likely reflect takeover frequency and execution quality. NDTW improves less because PACE targets critical stair transitions, whereas NDTW evaluates the full trajectory. Overall, PACE is broadly compatible with diverse zero-shot VLN navigators.

Qualitative Real-World Case Study. We deploy VLN-Zero (Bhatt et al., 2025) with PACE without an explicit navigation map or environment cache. Planning takes 0.2–0.5 s for the VLM or approximately 0.2 s for PACE; with 2 s MPC execution, each control cycle takes 2.2–2.5 s. As shown in Fig. 4, we evaluate the system in an unseen two-floor stairwell with an elevation gain of approximately 7 m under the instruction “Enter the stairwell and proceed upstairs until reaching the top floor.” SLAM is used only for trajectory visualization.

PACE successfully negotiates narrow and directionally ambiguous stair-transition regions, as illustrated by Regions (1) and (2). At stair corners, it predicts an executable affordance pose aligned with the next flight and translates it into local actions, allowing the robot to complete the turn and continue ascending. In Region (2), PACE occasionally emits STOP before reaching the stair exit; the high-level VLM detects the incomplete instruction and reactivates PACE, enabling recovery and completion of the two-floor ascent. The

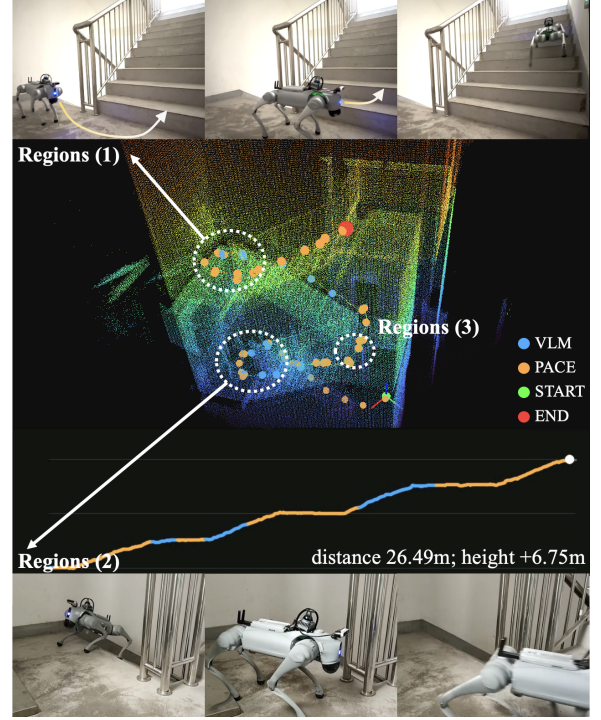


Figure 4: Real-world two-floor stair traversal with VLN-Zero and PACE. Blue and orange points denote VLM- and PACE-controlled trajectory segments, respectively.

alternating control is reflected by the interleaved VLM- and PACE-controlled trajectory segments.

PACE nevertheless exhibits local control instability during stair traversal, including left–right oscillations and reduced clearance from the railing. In Region (3), the clustered orange trajectory points indicate repeated local corrections and reduced execution efficiency, likely caused by sim-to-real viewpoint discrepancies and contact-induced perturbations during ascent.

4.3 Ablation Studies

# Method	OSR (%) ↑	CR (%) ↓	SPL (%) ↓	NDTW (%) ↑
1 PACE	67.4 ± 1.1	2.0 ± 0.2	64.3 ± 2.9	84.8 ± 1.1
2 Waypoint Cond.	63.1 ± 0.5	11.5 ± 4.5	59.9 ± 2.1	70.6 ± 1.0
3 w/o Pref. Loss	56.5 ± 1.4	4.0 ± 0.8	54.0 ± 4.7	64.3 ± 1.9
4 w/o Aff. Cond.	53.9 ± 2.6	4.7 ± 0.4	50.6 ± 3.6	59.8 ± 1.4
5 Aux. Aff. Only	49.3 ± 3.1	2.9 ± 0.2	47.6 ± 2.8	57.3 ± 3.5
6 w/o F.A. Ref.	37.3 ± 0.9	8.3 ± 2.2	34.5 ± 4.0	57.4 ± 1.3
7 Action-Only BC	17.5 ± 0.5	9.2 ± 3.7	15.6 ± 6.9	51.7 ± 2.6

Table 3: Standalone stair-traversal ablation of PACE. “Aff.”, “Cond.”, “Pref.”, and “F.A. Ref.” denote affordance, conditioning, preference, and failure-aware refinement, respectively. Results are reported as mean±standard deviation over three runs.

We conduct ablations on standalone stair-traversal segments of R2R-CE-CF to isolate local execution from high-level VLM uncertainty (Table 3). Compared with waypoint conditioning, PACE improves OSR by 4.3 percentage points while reducing CR by 9.5 points and

improving NDTW by 14.2 points, indicating that its affordance state provides more effective guidance than a generic goal pose. Removing affordance conditioning or using affordance prediction only as auxiliary supervision decreases OSR by 13.5 and 18.1 points, respectively, supporting its direct use by the action decoder. The action-only BC baseline exhibits a 49.9-point OSR gap, confirming the importance of the intermediate spatial representation. Removing the preference loss reduces OSR by 10.9 points, whereas removing the entire failure-aware refinement stage causes a 30.1-point drop, demonstrating the complementary contributions of preference optimization and policy-induced recovery supervision.

5 Conclusion

This work presents PACE, a plug-and-play affordance-conditioned supervised local execution framework that connects zero-shot semantic planners with reliable execution for cross-floor VLN. PACE grounds stair-transition semantics into traversable affordance poses, conditions short-horizon action generation on the learned affordance representation, and improves closed-loop correction through failure-aware preference refinement. Experiments across six zero-shot VLN navigators and real-world deployment demonstrate its effectiveness and applicability. Future work will extend PACE to other spatially constrained transitions, such as doorways and narrow passages.

References

- Dong An, Hanqing Wang, Wenguan Wang, Zun Wang, Yan Huang, Keji He, and Liang Wang. Etpnav: Evolving topological planning for vision-language navigation in continuous environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.
- Neel P Bhatt, Yunhao Yang, Rohan Siva, Pranay Samineni, Daniel Milan, Zhangyang Wang, and Ufuk Topcu. Vln-zero: Rapid exploration and cache-enabled neurosymbolic vision-language planning for zero-shot transfer in robot navigation. *arXiv preprint arXiv:2509.18592*, 2025.
- Lirong Che, Shuo Wen, Shan Huang, Chuang Wang, Yuzhe Yang, Gregory Dudek, Xueqian Wang, and Jian Su. Mansion: Multi-floor language-to-3d scene generation for long-horizon tasks, 2026. URL <https://arxiv.org/abs/2603.11554>.
- Jiaqi Chen, Bingqian Lin, Xinmin Liu, Lin Ma, Xiaodan Liang, and Kwan-Yee K Wong. Affordances-oriented planning using foundation models for continuous vision-language navigation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 23568–23576, 2025a.
- Kehan Chen, Dong An, Yan Huang, Rongtao Xu, Yifei Su, Yonggen Ling, Ian Reid, and Liang Wang. Constraint-aware zero-shot vision-language navigation in continuous environments, 2025b. URL <https://arxiv.org/abs/2412.10137>.
- Shiqi Chen, Tongyao Zhu, Ruochen Zhou, Jinghan Zhang, Siyang Gao, Juan Carlos Niebles, Mor Geva, Junxian He, Jiajun Wu, and Manling Li. Why is spatial reasoning hard for VLMs? An attention mechanism perspective on focus areas. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 9910–9932. PMLR, 13–19 Jul 2025c. URL <https://proceedings.mlr.press/v267/chen25cr.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.
- Fang Gao, Lei Shi, Jingfeng Tang, Jiabao Wang, Shaodong Li, Shengheng Ma, and Jun Yu. Visual and textual commonsense-enhanced layout learning for vision-and-language navigation. *IEEE Transactions on Automation Science and Engineering*, 22:21311–21324, 2025. doi: 10.1109/TASE.2025.3604018.
- Zeyong Gong, Rong Li, Tianshuai Hu, Ronghe Qiu, Lingdong Kong, Lingfeng Zhang, Guoyang Zhao, Yiyi Ding, and Junwei Liang. Stairway to success: An online floor-aware zero-shot object-goal navigation framework via llm-driven coarse-to-fine exploration. *IEEE Robotics and Automation Letters*, 11(3):2943–2950, 2026.
- Diqi He, Xuehao Gao, Hao Li, Junwei Han, and Dingwen Zhang. Strider: Navigation via instruction-aligned structural decision space optimization. *Advances in Neural Information Processing Systems*, 38: 26700–26720, 2026.
- Jingzhi Huang, Junkai Huang, Haoyang Yang, Haoang Li, and Yi Wang. Aerr-nav: Adaptive exploration-recovery-reminiscing strategy for zero-shot object navigation, 2026. URL <https://arxiv.org/abs/2603.17712>.
- brian ichter, Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, Dmitry Kalashnikov, Sergey Levine, Yao Lu, Carolina Parada, Kanishka Rao, Pierre Sermanet, Alexander T Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Mengyuan Yan, Noah Brown, Michael Ahn, Omar Cortes, Nicolas Sievers, Clayton Tan, Sichun Xu, Diego Reyes, Jarek Rettinghouse, Jornell Quiambao, Peter Pastor, Linda Luu,

- Kuang-Huei Lee, Yuheng Kuang, Sally Jesmonth, Nikhil J. Joshi, Kyle Jeffrey, Rosario Jauregui Ruano, Jasmine Hsu, Keerthana Gopalakrishnan, Byron David, Andy Zeng, and Chuyuan Kelly Fu. Do as i can, not as i say: Grounding language in robotic affordances. In Karen Liu, Dana Kulic, and Jeff Ichnowski, editors, *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 287–318. PMLR, 14–18 Dec 2023. URL <https://proceedings.mlr.press/v205/ichter23a.html>.
- Won Shik Jang and Ue-Hwan Kim. Context-nav: Context-driven exploration and viewpoint-aware 3d spatial reasoning for instance navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9626–9636, 2026.
- Yangzhe Kong, Daeun Song, Jing Liang, Dinesh Manocha, Ziyu Yao, and Xuesu Xiao. Autospatial: Visual-language reasoning for social robot navigation through efficient spatial reasoning learning. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11298–11304. IEEE, 2025.
- Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 104–120, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58604-1.
- Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4392–4412, 2020.
- Kailing Li, Tianwen Qian, Lijin Yang, Yuqian Fu, Jingyu Gong, Xiaoling Wang, and Liang He. Bridging the 2d-3d gap: A hierarchical semantic-geometric map for vision language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15243–15252, 2026.
- Yuxing Long, Xiaoqi Li, Wenzhe Cai, and Hao Dong. Discuss before moving: Visual language navigation via multi-expert discussions. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 17380–17387, 2024. doi: 10.1109/ICRA57147.2024.10611565.
- Yuxing Long, Wenzhe Cai, Hongcheng Wang, Guanqi Zhan, and Hao Dong. Instructnav: Zero-shot system for generic instruction navigation in unexplored environment. In *Conference on Robot Learning*, pages 2049–2060. PMLR, 2025.
- Yanyuan Qiao, Wenqi Lyu, Hui Wang, Zixu Wang, Zerui Li, Yuan Zhang, Minghui Tan, and Qi Wu. Open-nav: Exploring zero-shot vision-and-language navigation in continuous environment with open-source llms. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6710–6717. IEEE, 2025.
- Qwen Team. Qwen3.6-27B: Flagship-level coding in a 27B dense model, April 2026. URL <https://qwen.ai/blog?id=qwen3.6-27b>.
- Ram Ramrakhyia, Dhruv Batra, Erik Wijmans, and Abhishek Das. Pirlnav: Pretraining with imitation and rl finetuning for objectnav. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17896–17906, 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 fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.
- Jie-Jing Shao, Hao-Sen Shi, Lan-Zhe Guo, and Yufeng Li. Offline imitation learning with model-based reverse augmentation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2608–2617, 2024.
- Xiangyu Shi, Zerui Li, Wenqi Lyu, Jiatong Xia, Feras Dayoub, Yanyuan Qiao, and Qi Wu. Smartway: Enhanced waypoint prediction and backtracking for zero-shot vision-and-language navigation. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 16923–16930. IEEE, 2025.
- Ronghu Wang, Ziyang Liu, Weidong Chen, Banghai He, and Ziyang Wang. Semantic-aware and spatially-grounded multi-expert network for continuous vision-and-language navigation. *Knowledge-Based Systems*, page 116308, 2026a.
- Zihan Wang, Seungjun Lee, and Gim Hee Lee. Dynam3d: Dynamic layered 3d tokens empower vlm for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 38:153522–153544, 2026b.
- Meng Wei, Chenyang Wan, Jiaqi Peng, Xiqian Yu, Yuqiang Yang, Delin Feng, Wenzhe Cai, Chenming Zhu, Tai Wang, Jiangmiao Pang, et al. Ground slow, move fast: A dual-system foundation model for generalizable vision-and-language navigation. *arXiv preprint arXiv:2512.08186*, 2025.
- Abdelrhman Werby, Chenguang Huang, Martin Büchner, Abhinav Valada, and Wolfram Burgard. Hierarchical open-vocabulary 3d scene graphs for language-grounded robot navigation. In *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024*, 2024.

- 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.
- Jihan Yang, Shusheng Yang, Anjali W Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10632–10643, 2025.
- Naoki Yokoyama, Sehoon Ha, Dhruv Batra, Jiuguang Wang, and Bernadette Bucher. Vlmf: Vision-language frontier maps for zero-shot semantic navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 42–48. IEEE, 2024.
- Mingming Yu, Fei Zhu, Wenzhuo Liu, Yirong Yang, Qunbo Wang, Jing Liu, et al. C-nav: Towards self-evolving continual object navigation in open world. *Advances in Neural Information Processing Systems*, 38:126181–126207, 2026.
- Shuaihang Yuan, Hao Huang, Yu Hao, Congcong Wen, Anthony Tzes, and Yi Fang. Gamap: Zero-shot object goal navigation with multi-scale geometric-affordance guidance. *Advances in Neural Information Processing Systems*, 37:39386–39408, 2024.
- Jiazhao Zhang, Kunyu Wang, Rongtao Xu, Gengze Zhou, Yicong Hong, Xiaomeng Fang, Qi Wu, Zhizheng Zhang, and He Wang. Navid: Video-based vlm plans the next step for vision-and-language navigation. *arXiv preprint arXiv:2402.15852*, 2024.
- Jie Zhang, Xiaoyue Chen, Anzhe Chen, Dayiheng Liu, Deqing Li, Gengze Zhou, Hale Yin, Haoqi Yuan, Haoyang Li, Jiahao Li, Jiazhao Zhang, Jingen Zhou, Kaiyuan Gao, Kun Yan, Lihan Jiang, Ningyuan Tang, Pei Lin, Qihang Peng, Shengming Yin, Tianhe Wu, Tianyi Yan, Xiao Xu, Yan Shu, Yanran Zhang, Ye Wang, Yi Wang, Yilei Chen, Yixian Xu, Yiyang Huang, Yuxiang Chen, Zekai Zhang, Zhendong Wang, Zixing Lei, Zhixuan Liang, Zihao Liu, Zikai Zhou, Chenxu Lv, Xiong-Hui Chen, and Chenfei Wu. Qwen-robotworld technical report: Unifying embodied world modeling through language-conditioned video generation, 2026. URL <https://arxiv.org/abs/2606.17030>.
- Lingfeng Zhang, Xiaoshuai Hao, Qinwen Xu, Qiang Zhang, Xinyao Zhang, Pengwei Wang, Jing Zhang, Zhongyuan Wang, Shanghang Zhang, and Renjing Xu. Mapnav: A novel memory representation via annotated semantic maps for vlm-based vision-and-language navigation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13032–13056, 2025a.
- Lingfeng Zhang, Hao Wang, Erjia Xiao, Xinyao Zhang, Qiang Zhang, Zixuan Jiang, and Renjing Xu. Multi-floor zero-shot object navigation policy. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6416–6422, 2025b. doi: 10.1109/ICRA55743.2025.11128607.
- Han Zheng, Zhe Chen, Yudong Huang, Haoran Liu, Jinghao Wang, Ming Yang, and Tong Qin. Trav-explorer: Cross-floor embodied exploration via traversability-aware 3-d planning. *arXiv e-prints*, pages arXiv–2605, 2026a.
- Wancai Zheng, Hao Chen, Xianlong Lu, Linlin Ou, and Xinyi Yu. 3dgsnav: Enhancing vision-language model reasoning for object navigation via active 3d gaussian splatting, 2026b. URL <https://arxiv.org/abs/2602.12159>.
- Wanrong Zheng, Yunhao Ge, and Laurent Itti. Three-step nav: A hierarchical global-local planner for zero-shot vision-and-language navigation, 2026c. URL <https://arxiv.org/abs/2604.26946>.
- Gengze Zhou, Yicong Hong, and Qi Wu. Navgpt: Explicit reasoning in vision-and-language navigation with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 7641–7649, 2024.
- Zibo Zhou, Yue Hu, Ling kai Zhang, Zonglin Li, and Siheng Chen. Beliefmapnav: 3d voxel-based belief map for zero-shot object navigation. *Advances in Neural Information Processing Systems*, 38:83008–83036, 2026.

This document supplements the main submission. All notation, metrics, baselines, and external models referred to below follow the main paper, where the corresponding references are given.

A Failure-Aware Preference Refinement Details

Rollout and recovery labels. Each refinement round collects 2,048 closed-loop rollouts from recorded expert states and GroundingDINO-randomized approach states at a 1:8 ratio. The current policy is executed autoregressively, while the privileged expert route is used only during training to compute nearest-route positional and heading errors and to derive recovery targets. A deviation begins when either error exceeds 0.5 m or 30° for two consecutive steps. Failure is triggered immediately when either error exceeds 1.5 m or 60° . From a failure state, A* plans toward a future expert-route anchor, and the resulting path is converted into the same 0.25 m and 30° motion primitives. Recovery ends when both errors remain within 0.25 m and 30° for two consecutive steps. Algorithm 2 summarizes this training-time labeling process.

Algorithm 2 Rollout Failure and Recovery Labeling

Require: Current policy π_Θ and privileged expert route

- 1: $m \leftarrow \text{NORMAL}$; $n_{\text{drift}} \leftarrow 0$; $n_{\text{return}} \leftarrow 0$
- 2: **while** the rollout is active **do**
- 3: Compute nearest-route errors (d_t, δ_t)
- 4: **if** $m \neq \text{RECOVERY}$ **then**
- 5: **if** $d_t > 0.5 \text{ m}$ or $\delta_t > 30^\circ$ **then**
- 6: $n_{\text{drift}} \leftarrow n_{\text{drift}} + 1$
- 7: **else**
- 8: $n_{\text{drift}} \leftarrow 0$
- 9: **end if**
- 10: **if** $n_{\text{drift}} \geq 2$ **then**
- 11: $m \leftarrow \text{DEVIATION}$
- 12: **end if**
- 13: **if** $d_t > 1.5 \text{ m}$ or $\delta_t > 60^\circ$ **then**
- 14: Plan an A* path to a future expert-route anchor
- 15: Discretize the path into recovery primitives
- 16: $m \leftarrow \text{RECOVERY}$
- 17: **end if**
- 18: **else**
- 19: Execute the next A*-derived recovery primitive
- 20: **if** $d_t \leq 0.25 \text{ m}$ and $\delta_t \leq 30^\circ$ **then**
- 21: $n_{\text{return}} \leftarrow n_{\text{return}} + 1$
- 22: **else**
- 23: $n_{\text{return}} \leftarrow 0$
- 24: **end if**
- 25: **if** $n_{\text{return}} \geq 2$ **then**
- 26: $m \leftarrow \text{NORMAL}$
- 27: **end if**
- 28: **end if**
- 29: Record the current state and proposal with role m
- 30: **end while**

The expert route and A*-derived actions are privileged training signals and are never available during evaluation.

Snippet pools and matching. Within each rollout, maximal contiguous runs with the same role (NORMAL,

DEVIATION, or RECOVERY) are divided into all sliding windows of length five. The resulting windows are pooled globally across rollouts within the current refinement round. These pools are rebuilt, rather than accumulated, across rounds. After the three pools are independently shuffled, each deviation window is paired once with a normal window and once with a recovery window when the corresponding positive pool is non-empty. Positive pools wrap around when exhausted, producing at most two pairs per deviation window without imposing an additional class ratio.

Algorithm 3 Preference-Pair Construction in Round r

Require: Policy $\pi_{\Theta^{r-1}}$ and expert data

- 1: $\mathcal{T}^r \leftarrow \text{Rollout}(\pi_{\Theta^{r-1}})$
- 2: Label rollout states as normal, deviation, or recovery
- 3: Build length-5 pools $\mathcal{N}^r$, $\mathcal{D}^r$, and $\mathcal{R}^r$
- 4: Independently shuffle $\mathcal{N}^r$, $\mathcal{D}^r$, and $\mathcal{R}^r$
- 5: $\mathcal{P}^r \leftarrow \emptyset$
- 6: **for** $k = 1, \dots, |\mathcal{D}^r|$ **do**
- 7: **if** $|\mathcal{N}^r| > 0$ **then**
- 8: $u \leftarrow 1 + ((k-1) \bmod |\mathcal{N}^r|)$
- 9: $\mathcal{P}^r \leftarrow \mathcal{P}^r \cup \{(\mathcal{N}_u^r, \mathcal{D}_k^r)\}$
- 10: **end if**
- 11: **if** $|\mathcal{R}^r| > 0$ **then**
- 12: $v \leftarrow 1 + ((k-1) \bmod |\mathcal{R}^r|)$
- 13: $\mathcal{P}^r \leftarrow \mathcal{P}^r \cup \{(\mathcal{R}_v^r, \mathcal{D}_k^r)\}$
- 14: **end if**
- 15: **end for**
- 16: Update Θ^r using equal expert/on-policy batches and $\mathcal{P}^r$

Every recorded proposal, including its terminal STOP, is rescored under teacher forcing using model-predicted affordance features. For a snippet $\zeta = ((c_i, \tau_i))_{i=1}^L$ with $L = 5$, its score is

$$S_\Theta(\zeta) = \frac{1}{L} \sum_{i=1}^L \frac{1}{|\tau_i|} \sum_{j=1}^{|\tau_i|} \log p_\psi(a_{i,j} \mid a_{i,<j}, c_i, h_i^{\text{aff}}). \quad (18)$$

Token averaging removes a systematic preference for shorter proposals, while temporal averaging evaluates whether consecutive decisions sustain correction rather than relying on an isolated action.

B Implementation and Annotation Details

Architecture and optimization. PACE uses separate ImageNet-pretrained ResNet-18 backbones for RGB and depth. Their 512-dimensional outputs are each projected to $d/2 = 128$ dimensions, which is the shared visual feature space referred to in the main paper, and are combined by a learned gate. Historical actions and phases use 16-dimensional embeddings, historical affordance poses use a 64-dimensional MLP encoding, and directional guidance uses a 16-dimensional embedding. The concatenated visual, historical, and semantic features form one context token per step, which is projected to the $d = 256$ Transformer hidden space. The temporal encoder has two four-head layers, the query-based affordance estimator has one four-head decoder layer, and the causal action decoder has two four-head layers. All Transformer feed-forward layers

have dimension 1,024. The complete model has 27.4M trainable parameters, including 22.4M in the two visual backbones.

Setting	Value
Visual backbones	dual ImageNet-pretrained ResNet-18
Context	five steps ($K = 5$)
Hidden dimension	$d = 256$
Attention / FFN	4 heads / 1,024
Temporal/query/action	2/1/2 layers
Dropout	feature/Transformer = 0.35/0.20
Objective weights	$\lambda_p/\lambda_\phi/\lambda_a = 1/0.5/1$
Loss details	Smooth-L1 $\beta = 1$; label smoothing 0.05
Imitation learning	15 epochs; batch 32
IL optimizer	AdamW; LR 10^{-4} ; WD 5×10^{-4}
Failure-aware refinement	20 on-policy rounds; LR 10^{-5}
Refinement rollouts	2,048 per round
Rollout starts	expert/randomized = 1:8
Refinement batches	expert/on-policy = 16/16
Preference	five-step snippets; $\lambda_{\text{pref}} = 0.1$; $\beta = 1$
Proposal	maximum length 48, including STOP
Inference	beam size 5
Geom. weights	$(\lambda_g, \lambda_\theta, \lambda_s) = (0.35, 0.25, 0.5)$
IL checkpoint	lowest validation objective
Refinement checkpoint	final refinement round
Seeds	42 (main); 42/43/44 (ablations)

Table 4: PACE implementation settings. Imitation learning uses cosine learning-rate decay; all test-time models are frozen.

Cross-floor splits and scene isolation. We preserve the official R2R-CE and RxR-CE train/val-unseen assignments and retain episodes with $|h_s - h_g| > 1$ m. This yields 925/261 R2R-CE-CF and 932/217 RxR-CE-CF train/val-unseen episodes, matching the main paper. The retained train/val-unseen episodes span 61/11 R2R scenes and 59/11 RxR scenes, with no scene overlap between the two splits.

Pose and phase annotation. Algorithm 4 gives the deterministic label construction. A non-flat height change starts a stair run, and the preceding expert pose is treated as its entry. A direction reversal closes the run immediately; alternatively, F consecutive flat states confirm a plateau, with the first flat state selected as the exit. We use $F = 3$ for R2R and $F = 9$ for RxR and discard runs whose total height change is below 0.5 m. For runs with an observed pre-stair pose, APPROACH denotes the retained prefix, ENTRY marks the pre-stair key pose, TRAVERSE covers the interval toward the exit pose, and EXIT is terminal. When a sequence starts on the stairs, it enters TRAVERSE directly without an ENTRY label. Pose yaw follows the local expert-path tangent.

Because EXIT is terminal and defines no remaining traversal target, no affordance pose label is assigned to it. The pose regression term of $\mathcal{L}_{\text{dual}}$ is therefore masked on EXIT states, and the expectation in that term is taken over the non-terminal states only; the phase and action terms remain active on all states. Equivalently, λ_p is set to zero whenever $\phi_t^* = \text{EXIT}$.

Algorithm 4 Affordance Pose and Phase Labels

Require: Expert poses $\{q_i = (X_i, Y_i, Z_i, \theta_i)\}_{i=1}^T$

- 1: Quantize ΔY_i as up/down/flat using 0.05 m
- 2: Start a run at the first non-flat change; set entry $e = i - 1$
- 3: Close at a sign reversal, or after F flats using the first flat as exit x
- 4: Let $b \leftarrow (e = 1)$ indicate that the sequence starts on stairs
- 5: **if** $|Y_x - Y_e| \geq 0.5$ m **then**
- 6: **for** each index j associated with this run **do**
- 7: **if** b and $j < x$ **then**
- 8: $\phi_j^* = \text{TRAVERSE}$; $p_j^* = q_x$
- 9: **else if** $j < e$ **then**
- 10: $\phi_j^* = \text{APPROACH}$; $p_j^* = q_e$
- 11: **else if** $j = e$ **then**
- 12: $\phi_j^* = \text{ENTRY}$; $p_j^* = q_x$
- 13: **else if** $e < j < x$ **then**
- 14: $\phi_j^* = \text{TRAVERSE}$; $p_j^* = q_x$
- 15: **else**
- 16: $\phi_j^* = \text{EXIT}$; no pose target, mask $\mathcal{L}_{\text{pose}}$
- 17: **end if**
- 18: **end for**
- 19: **end if**

For the current horizontal pose (X_i, Z_i, θ_i) and target (X^*, Z^*, θ^*) , let $\Delta X = X^* - X_i$ and $\Delta Z = Z^* - Z_i$. The agent-centric label is

$$\begin{aligned} x^* &= \sin \theta_i \Delta X + \cos \theta_i \Delta Z, \\ y^* &= -\cos \theta_i \Delta X + \sin \theta_i \Delta Z, \\ \theta_{\text{rel}}^* &= \text{wrap}(\theta^* - \theta_i). \end{aligned} \quad (19)$$

Here, $+x$ points forward and $+y$ points left. Randomized approach trajectories terminate at the same entry pose and use the same agent-centric transformation.

C Integration Details

Unified baseline adapter. PACE interfaces with all six frozen navigators through unified RGB-D, semantic-guidance, and action interfaces. Each navigator retains its own maps, memory, hidden states, and planner; none are exposed to PACE or retrained. GroundingDINO first detects visible staircases. For each detection, the VLM assesses its relevance to the current instruction and returns a directional prediction $d \in \{\text{UP}, \text{DOWN}, \text{UNKNOWN}\}$ together with a relevance confidence score q . PACE is activated when the VLM either predicts an explicit traversal direction or confidently confirms that the detected staircase is relevant to the instruction, i.e., $d \in \{\text{UP}, \text{DOWN}\}$ or $q \geq \delta_{\text{act}}$. This confidence-based condition allows PACE to take over when the VLM reliably recognizes the required staircase even if its directional prediction remains UNKNOWN. Otherwise, the navigator retains control. Algorithm 5 summarizes this closed-loop integration.

Handoff and reactivation. PACE executes only the first primitive $\hat{a}_{t,1}$ of each short-horizon proposal and replans from the subsequent RGB-D observation. The terminal STOP ends only the current proposal and triggers the next closed-loop decision; it neither terminates the navigation episode nor immediately returns control to the navigator. PACE terminates upon episode

Algorithm 5 PACE Takeover and Handoff

Require: Frozen navigator $\mathcal{N}$, instruction I , observation o_t

```
1: mode  $\leftarrow$  NAVIGATOR
2:  $d \leftarrow$  UNKNOWN
3: while episode is active do
4:   if mode = NAVIGATOR then
5:      $d \leftarrow$  UNKNOWN;  $q \leftarrow$  0
6:     if GroundingDINO detects a visible staircase then
7:        $(d, q) \leftarrow$  VLM( $I, o_t$ )
8:     end if
9:     if ( $d \in \{\text{UP}, \text{DOWN}\} \vee q \geq \delta_{\text{act}}$ ) and budget remains
       then
10:       initialize PACE context
11:       mode  $\leftarrow$  PACE
12:     else
13:       execute  $\mathcal{N}(I, o_t)$  and acquire  $o_{t+1}$ 
14:     end if
15:   else
16:      $(\hat{\phi}_t, \hat{\tau}_t) \leftarrow$  PACE( $c_t, d$ )
17:     if  $\hat{\phi}_t = \text{EXIT}$  then
18:       mode  $\leftarrow$  NAVIGATOR
19:     else
20:       execute  $\hat{a}_{t,1}$  and acquire  $o_{t+1}$ 
21:       update context  $c_{t+1}$ 
22:       if episode completion or safety/budget event then
23:         mode  $\leftarrow$  NAVIGATOR
24:       end if
25:     end if
26:   end if
27: end while
```

completion and otherwise hands control back when the phase head predicts EXIT or when a safety or budget limit is reached. In simulation, each takeover is limited to 200 executed primitives and is interrupted after five consecutive forward collisions or six consecutive forward attempts producing less than 0.05 m displacement. After handoff, the navigator receives the latest observation and may reactivate PACE through the same staircase-detection and VLM-based activation gate, provided that traversal remains incomplete and sufficient takeover budget remains.

Consequently, a handoff occurring before the stair exit has actually been reached is caused by the phase head rather than by the proposal-level STOP. Specifically, an EXIT prediction emitted while the agent is still traversing returns control immediately, consistent with the phase-boundary ambiguity reported in the main paper. The reactivation gate makes such premature handoffs recoverable: if the staircase remains visible and the VLM again predicts a traversal direction or confidently confirms its relevance, PACE can be reactivated in both simulation and real-world deployment.

D Expanded Evaluation Results

Table 6 expands the compact significance markers in the main paper by reporting the exact effect sizes and confidence intervals for all 48 method-metric comparisons. PACE yields robust gains in 9/12 SR, 12/12 OSR, 8/12 SPL, and 11/12 NDTW comparisons. Of the remaining eight comparisons, seven are inconclusive; the only robust decrease is NDTW for 3-step-Nav on RxR-CE-CF ($\Delta = -0.051$, 95% CI $[-0.081, -0.022]$).

Table 5 complements the takeover analysis in the main paper with its exact SR intervals. All 12 intervals are strictly above zero, with gains ranging from 9.68 to 44.12 percentage points, providing consistent evidence across navigator-benchmark pairs.

Alone and +PACE are aligned by episode ID but use independently sampled VLM outputs. Thus, the analysis retains the realized end-to-end variation rather than relying on matched VLM decisions. The uniformly positive takeover-conditioned intervals indicate that the observed SR advantage remains stable under this variation, although repeated decoding seeds would be required to isolate VLM-induced variance.

Method	R2R-CE-CF		RxR-CE-CF	
	Δ SR (pp)	95% CI	Δ SR (pp)	95% CI
VLN-Zero	44.12	[34.31, 53.92]	33.33	[16.67, 54.17]
3-step-Nav	26.67	[16.30, 36.30]	9.68	[2.15, 18.28]
CA-Nav	16.96	[7.60, 25.73]	22.08	[10.39, 33.77]
STRIDER	27.78	[13.89, 40.28]	18.92	[8.11, 32.43]
Open-Nav	14.00	[3.00, 25.00]	16.39	[4.92, 27.87]
HSGM	29.00	[21.00, 37.00]	40.54	[30.63, 50.45]

Table 5: Takeover-conditioned SR differences and their percentile paired-bootstrap 95% confidence intervals.

E Real-World Robot Deployment Details

Table 7 summarizes the real-world hardware and computation assignment. The Unitree Go2 Edu carries a RealSense D435i and a Livox Mid-360, while its Jetson Orin NX runs sensing, motion control, and data-recording services. Learned-model inference is performed offboard: PACE runs on an RTX 3070 Ti laptop, which reaches the same self-hosted Qwen-3.6-27B-FP8 deployment used in simulation. As in simulation, the high-level VLM is served locally on our own workstation rather than by a third-party service; the laptop only acts as a client and reaches that self-hosted endpoint over HTTPS within a private network.

The offboard runtime requests fresh RGB-D observations and submits one discrete action at a time to the NX through authenticated HTTP. FORWARD is mapped to a 0.25 m relative target and tracked from sport-mode odometry using velocity-reference MPC at 10 Hz with a 15-step prediction horizon. LEFT and RIGHT use odometry-closed-loop 30° rotations. Velocity commands are sent at 50 Hz, and each primitive is allocated a 2 s execution budget. Unitree’s built-in obstacle avoidance is enabled on level ground and disabled during stair traversal to avoid altering the commanded stair motion. Together with 0.2–0.5 s VLM or approximately 0.2 s PACE inference, the control cycle is 2.2–2.5 s under the reported deployment protocol.

The Mid-360 measurements are processed offline for SLAM and trajectory visualization and are not used for planning or as input to PACE. In the unseen two-floor

Method	R2R-CE-CF				RxR-CE-CF			
	Δ SR (pp)	Δ OSR (pp)	Δ SPL (pp)	Δ NDTW	Δ SR (pp)	Δ OSR (pp)	Δ SPL (pp)	Δ NDTW
VLN-Zero	16.86 [11.88,22.22]	19.92 [14.56,25.67]	12.46 [8.13,16.88]	0.219 [0.184,0.255]	4.15 [0.92, 7.37]	4.15 [0.92, 7.37]	2.73 [0.66, 5.10]	0.035 [0.010, 0.062]
3-step-Nav	12.64 [7.28,18.39]	24.52 [18.01,31.03]	6.58 [2.40,10.76]	0.074 [0.041,0.108]	4.15 [0.46, 7.83]	14.29 [9.22,19.35]	-1.16 [-3.87, 1.38]	-0.051 [-0.081,-0.022]
CA-Nav	10.73 [4.21,17.24]	28.74 [21.84,36.02]	4.69 [1.20, 8.16]	0.060 [0.030,0.091]	7.83 [3.69,11.98]	13.36 [8.76,17.97]	3.41 [1.64, 5.36]	0.033 [0.014, 0.053]
STRIDER	4.60 [-0.38, 9.96]	7.66 [2.30,13.03]	1.69 [-2.64, 6.06]	0.034 [0.009,0.059]	5.99 [3.23, 9.22]	7.83 [4.61,11.52]	4.84 [2.39, 7.54]	0.031 [0.010, 0.053]
Open-Nav	4.60 [-0.77, 9.96]	13.03 [6.90,19.16]	1.56 [-3.22, 6.25]	0.053 [0.028,0.079]	3.69 [0.00, 7.37]	5.99 [3.23, 9.22]	2.92 [-0.34, 6.31]	0.065 [0.042, 0.089]
HSGM	18.39 [11.49,25.29]	20.31 [13.41,26.82]	15.06 [9.75,20.25]	0.053 [0.021,0.084]	17.97 [11.52,24.42]	23.96 [17.51,30.41]	12.53 [7.93,17.19]	0.081 [0.048, 0.114]

Table 6: Observed differences $\Delta = +\text{PACE} - \text{Alone}$ under the main-paper protocol. The first and second lines for each method report Δ and its percentile paired-bootstrap 95% CI, respectively, using 10,000 episode-aligned resamples.

Component	Configuration / role
Robot	Unitree Go2 Edu
Onboard compute	Jetson Orin NX (16 GB); sensing and motion
RGB-D sensor	RealSense D435i; 848×480 at 30 Hz
LiDAR	Livox Mid-360; offline mapping only
Offboard runtime	RTX 3070 Ti laptop; PACE and VLM client
High-level VLM	self-hosted Qwen-3.6-27B-FP8; HTTPS client
Robot link	Authenticated HTTP over a private network

Table 7: Hardware and computation assignment for real-world deployment.

stairwell, the main observed failure modes were left-right oscillation, reduced railing clearance, and a premature handoff before the stair exit was actually reached. In the last case, the navigator received the fresh observation, the VLM reassessed the instruction as incomplete, and PACE was reactivated through the same detection and direction gate, completing the traversal. These observations expose remaining sim-to-real sensitivity to viewpoint shifts and contact perturbations. Figure 5 complements the stairwell case with two additional unseen indoor navigation examples; their point-cloud maps are visualization outputs only.

F Failure Case Analysis

Protocol. We pool the 261 R2R-CE-CF and 217 RxR-CE-CF val-unseen episodes, yielding 478 episode-aligned evaluations per baseline under the main-paper protocol. Failed episodes are assigned to four mutually exclusive categories in priority order: *stair transition* when the aligned PACE run invokes stair takeover on that episode, *goal stopping* when OSR = 1 but SR = 0, *route deviation* when NDTW < 0.30, and *local execution* for the remaining failures. Each segment is normalized by all 478 episodes rather than only the failed ones, so the segments of one bar sum to that baseline’s overall failure rate.

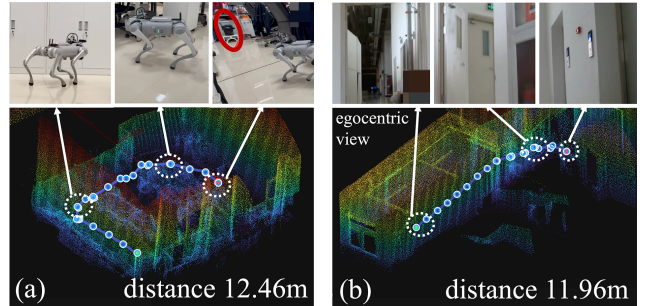


Figure 5: Additional real-world navigation examples. (a) “Walk forward to the end, turn right, and continue straight until you reach the trash bin.” (b) “Walk forward through the first doorway, turn right at the second doorway, and stop in front of the elevator.” The point-cloud maps visualize the executed trajectories and are not used for planning or as input to PACE.

Two properties of this decomposition should be noted. First, the stair-transition category is defined by the takeover decision of the aligned PACE run, so it identifies episodes whose failure coincides with a stair transition rather than establishing that the stairs caused the failure. Second, the priority order assigns an episode that both fails at a stair transition and stops away from the goal to the stair-transition category; the categories are therefore mutually exclusive by construction but not independent.

Results. As shown in Fig. 6, PACE reduces the average overall failure rate from 88.9% to 79.4% (−9.5 percentage points) across all six baselines. The average stair-transition failure rate decreases more substantially, from 34.2% to 23.8% (−10.4 points, or 30.4% relatively). HSGM exhibits the largest improvement, with stair-related failures decreasing from 47.9% to 26.4%. Since the stair-transition reduction of 10.4 points exceeds the overall reduction of 9.5 points, the aggregate gain is attributable to the stair-transition category, with the remaining categories contributing no net reduction. This

is consistent with PACE acting locally at constrained transitions rather than improving global instruction following, and with the smaller NDTW gains reported in the main paper.

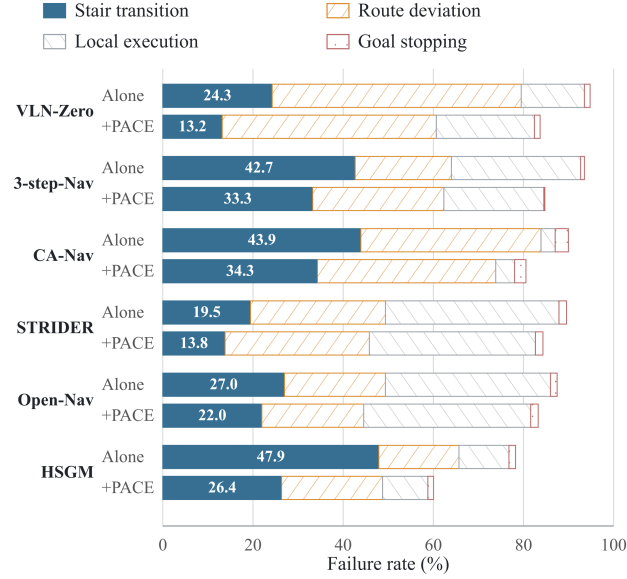


Figure 6: Failure-rate decomposition across six navigation baselines on the 478 episode-aligned val-unseen episodes. Segment lengths are normalized by all episodes, such that each bar represents the overall failure rate. Blue highlights stair-transition failures.

Qualitative success cases. Figure 7 illustrates representative successful trajectories across all six navigators. In the same scene, PACE enables 3-step-Nav, Open-Nav, and STRIDER to complete the cross-floor route despite their different perception, memory, and planning mechanisms. The remaining examples further demonstrate robustness to takeover timing: PACE can guide the agent through the transition when activated either after it has entered the staircase, as in VLN-Zero, or near the stair entrance, as in HSGM.

Failure cases. Figure 8 illustrates several remaining failure modes. For CA-Nav, PACE fails to exit and return control, exhausts its takeover budget, and drives the agent away from the target, constituting a harmful takeover. With 3-step-Nav, repeated collisions lead to persistent turning without effective recovery. For Open-Nav and STRIDER, the underlying navigators fail to identify a reasonable global direction, which cannot be corrected by a local execution module. In the HSGM case, PACE is never activated, so the failure originates from the baseline navigation process rather than PACE execution. These cases highlight limitations in exit prediction, collision recovery, upstream direction selection, and takeover triggering.

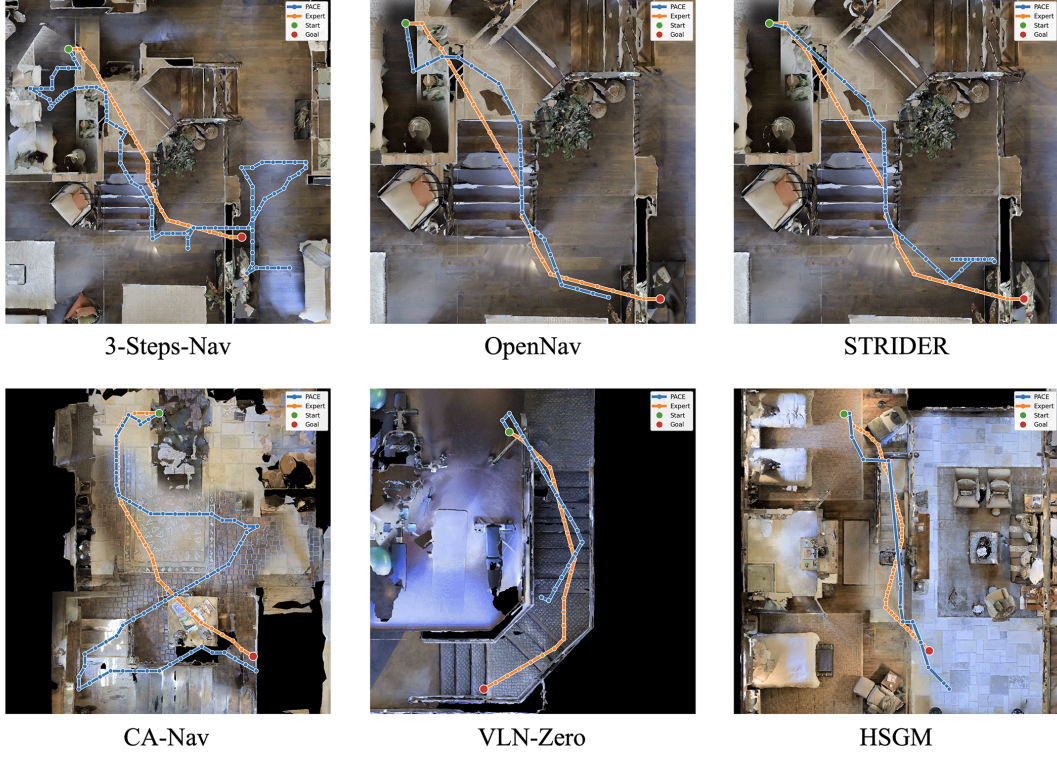


Figure 7: Representative successful navigation cases with PACE. The top row compares 3-step-Nav, Open-Nav, and STRIDER in the same scene, showing that PACE supports navigators with different internal mechanisms. The bottom row presents additional cases across CA-Nav, VLN-Zero, and HSGM. In particular, PACE completes the traversal when activated either on the staircase (VLN-Zero) or near the stair entrance (HSGM). Blue and orange denote the PACE-executed and expert trajectories, respectively.

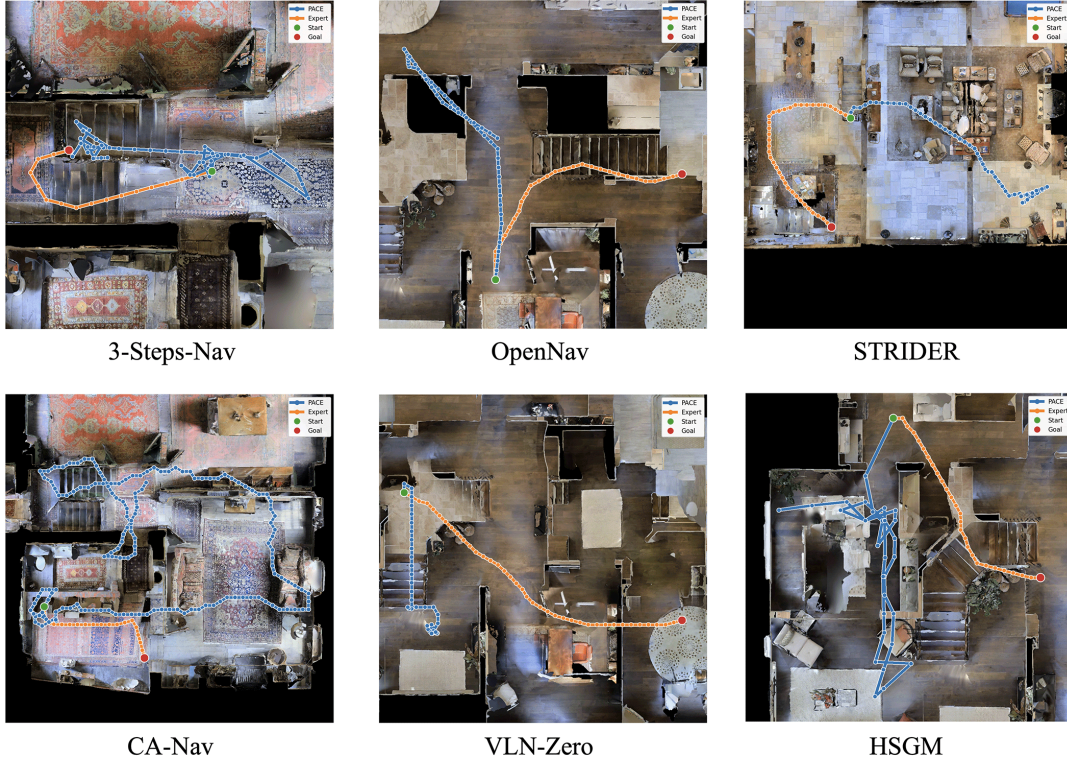


Figure 8: Representative failure cases across the six navigators. Blue and orange denote the executed and expert trajectories, respectively. Failures arise from both PACE-specific execution errors and limitations of the underlying navigators.