View Invariant Learning for Vision-Language Navigation in Continuous Environments

Josh Qixuan Sun, Huaiyuan Weng, *Graduate Student Member, IEEE*, Xiaoying Xing, Chul Min Yeum, and Mark Crowley, *Member, IEEE*

***Abstract*—Vision-Language Navigation in Continuous Environments (VLNCE), where an agent follows instructions and moves freely to reach a destination, is a key research problem in embodied AI. However, most existing approaches are sensitive to viewpoint changes, i.e. variations in camera height and viewing angle. Here we introduce a more general scenario, V^2-VLNCE (VLNCE with Varied Viewpoints) and propose a view-invariant post-training framework, called VIL (View Invariant Learning), that makes existing navigation policies more robust to changes in camera viewpoint. VIL employs a contrastive learning framework to learn sparse and view-invariant features. We also introduce a teacher-student framework for the Waypoint Predictor Module, a standard part of VLNCE baselines, where a view-dependent teacher model distills knowledge into a view-invariant student model. We employ an end-to-end training paradigm to jointly optimize these components. Empirical results show that our method outperforms state-of-the-art approaches on V^2-VLNCE by 8-15% measured on Success Rate for two standard benchmark datasets R2R-CE and RxR-CE. Evaluation of VIL in standard VLNCE settings shows that despite being trained for varied viewpoints, VIL often still improves performance. On the harder RxR-CE dataset, our method also achieved state-of-the-art performance across all metrics. This suggests that adding VIL does not diminish the standard viewpoint performance and can serve as a plug-and-play post-training method. We further evaluate VIL for simulated camera placements derived from real robot configurations (e.g. Stretch RE-1, LoCoBot), showing consistent improvements of performance. Finally, we present a proof-of-concept real-robot evaluation in two physical environments using a panoramic RGB sensor combined with LiDAR. These results show that VIL improves robustness not only in simulation but also in real-world navigation scenarios, making it a practical strategy for embodied agents. The code is available at https://github.com/realjoshqsun/V2-VLNCE.**

***Index Terms*—Vision-Based Navigation, Representation learning, Robot learning, Deep Learning Methods.**

Received 2 October 2025; accepted 16 February 2026. Date of publication 3 March 2026; date of current version 1 April 2026. This article was recommended for publication by Associate Editor F. Rameau and Editor P. Vasseur upon evaluation of the reviewers' comments. This work was supported by NSERC Discovery Grant "Leveraging Inductive Biases and Self-Supervised Learning for Efficient Learning in Complex Domains" Prof. Mark Crowley, University of Waterloo 2025-2030 under Grant RGPIN-2025-07221. *(Corresponding author: Josh Qixuan Sun.)*

Josh Qixuan Sun and Mark Crowley are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: q84sun@uwaterloo.ca; mcrowley@uwaterloo.ca).

Huaiyuan Weng and Chul Min Yeum are with the Department of Civil and Environmental Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: h22weng@uwaterloo.ca; chulminy@gmail.com).

Xiaoying Xing is with the Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL 60208 USA (e-mail: xiaoyingxing2026@u.northwestern.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/LRA.2026.3669785, provided by the authors.

Digital Object Identifier 10.1109/LRA.2026.3669785

I. Introduction

VISION-LANGUAGE *Navigation (VLN)* [1] requires an agent to follow human instructions by executing a sequence of actions. Traditional VLN assumes a predefined topological graph, where the agent moves along predefined graph edges. The broader *VLN in Continuous Environments (VLNCE)* [2] problem removes this constraint, enabling agents to move freely in continuous space. Prior works in VLNCE mainly improve navigation by designing better neural architectures [3], [4], [5], [6] or incorporating richer spatial and semantic features [7], [8], [9], [10]. However, these learned policies often struggle when the camera viewpoint (i.e. height and viewing angle) changes during deployment where even small shifts in viewpoint can lead to large performance drops.

This *Varied Viewpoint* challenge, occurs when agents need to generalize across environments with different egocentric camera placements and is especially relevant in real-world robotics, where robots have different camera mounting positions. To systematically study this problem, we introduce *V^2-VLNCE (VLNCE with Varied Viewpoints)*, a generalized setting designed to evaluate policy performance under diverse camera viewpoints. We focus on two variables: camera height and angle. In each episode, we sample a viewpoint from a 2D distribution over a range of heights and angles which better reflect the variation in real-world scenarios. While prior work [11], [12] considered this varied viewpoint challenge, they were developed for robotic manipulation rather than navigation tasks. As for VLNCE, GV-Nav [13] focused on the impact of height shift and addressed by training with this specific configuration. However, that approach only considers a single, fixed camera height and does not account for variations in both heights and angles simultaneously as we do.

While prior work has touched on aspects of viewpoint variation, their solutions are either not applicable to navigation or suffer from significant inefficiencies. For robotic manipulation, methods like MV-MVM [11], RoboUniView [12], and ReViWo [14] address varied viewpoints, but only for manipulation tasks. They also often rely on extensive pre-training to learn robust representations, which are then fine-tuned for specific downstream tasks. Thus, this paradigm is computationally intensive and not easily transferable. For VLNCE, GVNav [13] focuses on the impact of viewpoint changes by adopting a single, fixed ground-level viewpoint and training their model from scratch with this specific configuration. Thus, they cannot account for more complex, continuous variations in both heights

and angles. As we will demonstrate in Section IV-C, this simple retraining strategy proves insufficient to handle the more demanding V^2-VLNCE setting. These limitations show a critical need for more computationally efficient and generalizable approaches. Instead of costly retraining for each new viewpoint, we develop a single policy that can be adapted to diverse viewpoints with minimal effort.

We propose *View Invariant Learning (VIL)*, a strategy that adapts existing policies to varied viewpoints without retraining from scratch. VIL consists of two components: a contrastive learning objective and a teacher-student framework for waypoint prediction. The contrastive framework encourages the policy to learn sparse, view-invariant features by aligning representations from different viewpoints of the same scene, while separating unrelated observations. Features used for contrastive learning are extracted through a projection head, and the learned representations are shared with the navigation policy. For waypoint prediction, a frozen teacher model, initialized from a pretrained policy, processes observations from a standard viewpoint. The student model shares the same architecture as the teacher but trains only a small adapter module inserted into the waypoint predictor, while freezing the rest of the weights. The student, receiving varied-viewpoint inputs, learns to match the teacher's outputs through a distillation loss. Both components are trained jointly and end-to-end to enable efficient viewpoint adaptation.

Our contributions are as follows:

1) We introduce V^2-VLNCE, a new evaluation setting that incorporates both camera height and viewing angle variations to simulate diverse camera viewpoints. This setting enables a more realistic and systematic analysis of viewpoint robustness.
2) We propose VIL, a strategy trained with diverse viewpoints using a contrastive learning objective and a teacher-student framework.
3) We conduct extensive experiments in simulation, showing that VIL outperforms existing baselines in the V^2-VLNCE setting.
4) We further evaluate VIL under simulated camera placements derived from real robot configurations, confirming robustness across practical embodiment settings.

II. Related Work

Vision-language navigation: Prior studies on VLNCE [2] have focused on enhancing input modalities through a variety of methods, including panoramic RGB-D images [3], [5], [6], [7], [8], semantic information [4], [9], [15], [16], occupancy maps [10], [17], and larger-scale training data [18], [19]. Other works focus on designing more efficient neural networks for vision-and-language fusion [20], [21], [22], [23], [24].

Waypoint predictors are crucial to recent VLNCE models [6], [7], [19], as they bridge VLN and VLNCE and enable pretraining on VLN. To adapt to ground-level views, GVNav [13] retrained the waypoint predictor separately with matching data. In contrast, we train the entire model end-to-end, removing the need for separate waypoint predictor training.

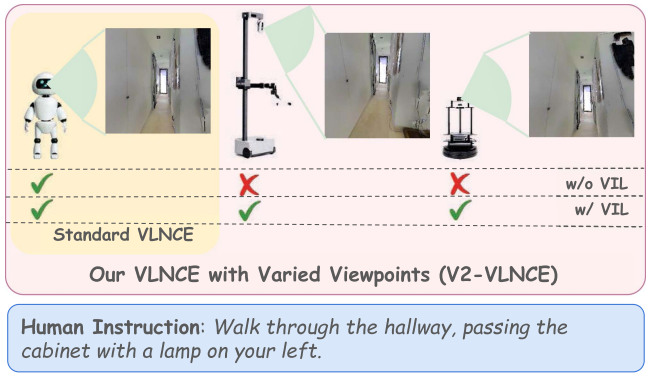

Fig. 1. Comparison between standard VLNCE and our proposed V^2-VLNCE. Under viewpoint changes, baseline navigation policies suffer from degraded performance. Applying View Invariant Learning (VIL) significantly improves robustness, enabling the agent to navigate under varied viewpoints.

Varied viewpoint challenge in robotics: In robotic manipulation, several works focus on learning view-invariant representations to address viewpoint variation [11], [12], [14]. However, these approaches usually adopt a two-stage training pipeline: first learning a view-invariant encoder, then training the policy on top of the frozen encoder. This strategy is less suitable for VLNCE. First, VLNCE policies are typically pretrained on VLN datasets, and applying a two-stage pipeline would discard the benefits of this pretraining. Second, the training cost would be high. Our goal is not to retrain new policies from scratch, but to adapt existing policies to varied viewpoints. Third, VLNCE architectures often include a waypoint predictor, which would also require separate training in a two-stage pipeline, further increasing the cost.

In robotic navigation, several recent works have explored viewpoint robustness under different task settings. GVNav [13] studies ground-level viewpoint variation for VLNCE by retraining both the navigation policy and the waypoint predictor on a fixed low-height camera configuration. In contrast, our work considers a more general viewpoint setting by modeling a joint distribution over camera height and pitch angle, which better reflects realistic camera mounting variations. Moreover, GVNav relies on a decoupled training scheme with separate optimization of the waypoint predictor and the policy, whereas our VIL framework enables efficient end-to-end adaptation through lightweight adapters without retraining the core pretrained policy. RING [25], a concurrent work, investigates viewpoint robustness for the ObjectNav benchmark by randomizing camera configurations during training. However, ObjectNav aims a simple semantic label (e.g., "find a mug"), while VLNCE requires the agent to faithfully follow long-form, multi-step linguistic instructions and reach specific subgoals along a trajectory and demands much finer alignment between language and visual perspective. Methodologically, RING follows a domain randomization strategy, whereas our approach introduces two architecture-compatible modules, contrastive learning and waypoint predictor distillation, to explicitly enforce viewpoint invariance while preserving pretrained VLNCE knowledge.

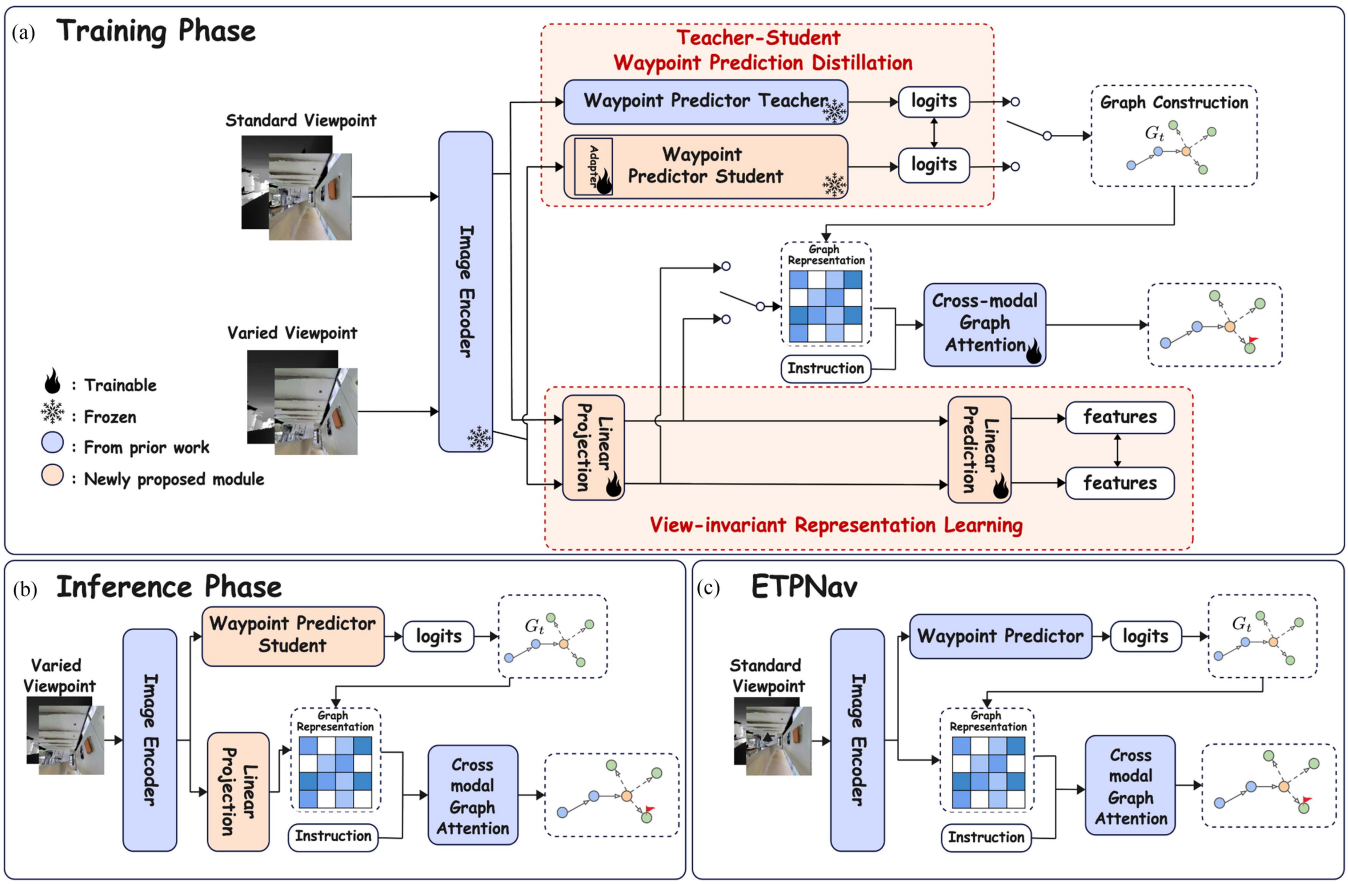

Fig. 2. Overview of our view-invariant learning framework. (a) Training Phase: Given standard and varied viewpoints, the image encoder extracts features for both. A contrastive learning objective is applied to align representations across viewpoints and encourage view-invariant features. Meanwhile, a teacher-student framework is used for waypoint prediction, where a frozen teacher processes standard views and a student model adapts to varied views by training only a lightweight adapter module. (b) Inference Phase: Only the student model is used to predict waypoints under varied viewpoints. (c) ETPNav baseline: A standard VLNCE architecture without contrastive learning or teacher-student training. .

III. Method

A. ETPNav Preliminary

We build on ETPNav [6], a strong panoramic VLNCE baseline. At each step t, the agent receives a natural language instruction and panoramic RGB-D observations $O_t = \{O_t^{\text{rgb}}, O_t^{\text{d}}\}$ consisting of 12 RGB and 12 depth views captured at equally spaced 30° intervals. As illustrated in Fig. 2(c), ETPNav predicts navigable waypoint candidates from the panoramic inputs, incrementally builds a local topological map, fuses instruction embeddings with graph representations through cross-modal graph attention, and finally selects the next navigation target. For details of the full architecture, we refer readers to [6].

B. View-Invariant Representation Learning

Policies trained under a fixed camera configuration often fail to generalize when the viewpoint changes. To address this, we introduce a contrastive learning objective that encourages learning of viewpoint-invariant features and integrates directly into the navigation model.

Given a panoramic RGB-D observation O_t at time step t, we generate two views of the scene: a standard viewpoint O_t^{std} and a varied viewpoint O_t^{var}, created by randomly shifting the camera height and angle. Both views are encoded by a shared visual encoder $f_{\text{enc}}(\cdot)$.

We apply a three-layer projection head $g(\cdot)$ after the encoder, following the standard design of SimCLRv2 [26]. We denote the output of the first linear layer as $g_1(\cdot)$, the second as $g_2(\cdot)$, and the third as $g_3(\cdot)$. The features used for downstream navigation and contrastive learning are:

$$f_{\text{task}} = g_1(f_{\text{enc}}(O_t)), \quad f_{\text{contrast}} = g_3(g_2(g_1(f_{\text{enc}}(O_t))))$$

We further distinguish the task features from different viewpoints. Let $f_{\text{task}}^{\text{std}}$ denote the task feature from the standard viewpoint and $f_{\text{task}}^{\text{var}}$ denote that from the varied viewpoint. We construct a graph representation by sampling either $f_{\text{task}}^{\text{std}}$ or $f_{\text{task}}^{\text{var}}$ with probability p_1, and combine the selected features with the topological graph G_t to represent the current scene structure. For contrastive learning, we also denote the features from the two viewpoints separately as $f_{\text{contrast}}^{\text{std}}$ and $f_{\text{contrast}}^{\text{var}}$ to compute the contrastive loss across viewpoints.

To ensure compatibility with the pretrained ETPNav model, we initialize the first linear layer g_1 as an identity matrix. This initialization keeps the original feature distribution at the start of training and allows gradual adaptation to varied viewpoints.

The contrastive learning objective enforces feature consistency between the standard and varied views of the same scene. For each instance in a training batch, indexed by (i, j) where i denotes the batch index and j denotes the panoramic view index, where $j \in [0, 1, \ldots, 11]$, corresponding to $[0°, 30°, \ldots, 330°]$ heading angles., we define positive pairs as the features corresponding to the same heading j under standard and varied viewpoints. Negative pairs are constructed from two sources: (1) random cross-scene negatives sampled from different scenes. For implementation efficiency, the latter is achieved by shifting indices within the mini-batch, i.e., $((i+1) \pmod{\text{batch\_size}}, j)$, and (2) intra-scene hard negatives by selecting features from the opposite heading $(i, (j+6) \pmod{12})$ The contrastive loss follows the InfoNCE [26]:

$$\mathcal{L}_{\text{cl}} = -\log \frac{\exp(\text{sim}(q, k^+)/\tau)}{\exp(\text{sim}(q, k^+)/\tau) + \sum_{k^-} \exp(\text{sim}(q, k^-)/\tau)},$$

where q is the feature of the standard view, k^+ is the feature of the varied view of the same scene, k^- are features from negative samples, and $\text{sim}(\cdot, \cdot)$ denotes cosine similarity. τ is the temperature parameter, set to 1.0 following standard practice. By jointly optimizing this contrastive objective with the navigation policy, the agent learns feature representations that are more robust to viewpoint variations without sacrificing performance on the original downstream task.

C. Teacher-Student Waypoint Prediction Distillation

In VLN tasks, the quality of waypoint prediction is critical for performances. Recent works such as GVNav [13] have observed that waypoint predictors trained under the standard viewpoint experience significant performance degradation when evaluated from a ground-level viewpoint. GVNav addresses this issue by retraining the waypoint predictor separately with ground-level viewpoint data, but this two-stage training strategy incurs a high training cost. In contrast, we propose an integrated teacher-student framework that enables robust waypoint prediction under varied viewpoints without additional training stages.

The teacher and student models share the same waypoint predictor architecture, introduced in Section III-A, where a transformer-based network predicts a dense heatmap of nearby navigable waypoints from panoramic RGB-D observations. Both the teacher and student models are initialized from the one used in ETPNav [6]. As shown in Fig. 2(a), the teacher model is frozen, and operates on standard viewpoint observations. The student model processes varied viewpoint observations and adapts via lightweight adapter layers, while the rest of the weights are frozen. Fig. 3 illustrates the detailed architecture of the waypoint predictor student in Figure 2(a). Importantly, the adapter is implemented as the original input linear layer of the predictor, which is made trainable during distillation, rather than being inserted as an additional module into the backbone. Formally, given an observation at time t, the teacher outputs waypoint logits S_t^{teacher}, and the student outputs S_t^{student}. To align the student with the teacher, we apply KL divergence as the distillation loss:

$$\mathcal{L}_{\text{wpd}} = \text{KL}\left(\text{softmax}(S_t^{\text{teacher}}) \parallel \text{softmax}(S_t^{\text{student}})\right).$$

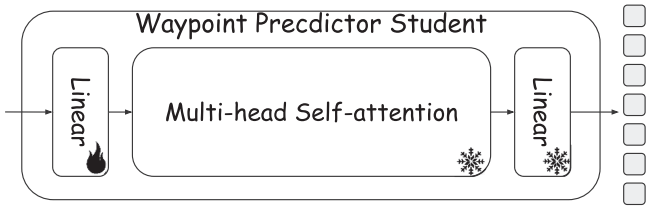

Fig. 3. Detailed architecture of the waypoint predictor student module used in teacher–student distillation.

where both logits are normalized by softmax before computing divergence. During graph construction, we sample either the teacher or student predictions, with probability p_2 or $1 - p_2$ respectively, to build the local topological map G_t.

D. Training Objective

Our full model is trained end-to-end by jointly optimizing three objectives: the standard navigation loss $\mathcal{L}_{\text{nav}}$, the contrastive learning loss $\mathcal{L}_{\text{cl}}$ introduced in Section III-B, and the waypoint predictor distillation loss $\mathcal{L}_{\text{wpd}}$ introduced in Section III-C. The overall training loss is formulated as:

$$\mathcal{L} = \mathcal{L}_{\text{nav}} + \lambda_1 \mathcal{L}_{\text{cl}} + \lambda_2 \mathcal{L}_{\text{wpd}}.$$

Here, λ_1 and λ_2 are hyperparameters that balance the contributions of different losses.

IV. Experiments

We aim to answer the following four research questions. **Q1**: How does our VIL strategy perform compared to existing baseline methods under varied viewpoints? (Section IV-A) **Q2**: Does VIL maintain performance on the original VLNCE setting? (Section IV-B) **Q3**: Is standard fine-tuning (i.e., retraining the model under varied viewpoint data) sufficient? What is the contribution of each component? (Section IV-C) **Q4**: Does VIL extrapolate to out-of-distribution viewpoints? (Section IV-E)

Baselines: We evaluate evaluate our VIL strategy by applying it to two strong VLNCE baselines: BEVBert [27] and ETPNav [6]. Both methods demonstrate strong performance on standard VLNCE benchmarks and provide trained checkpoints, making them widely used in recent studies. We apply VIL on top of each baseline to evaluate its compatibility and performance gain across different architectures.

Benchmarks: The R2R-CE dataset [2] comprises a total of 5,611 trajectories and the average path length is 9.89 m and each instruction is comprised of an average of 32 words. Compared to R2R-CE, RxR-CE [28] is larger and more challenging. RxR-CE presents substantively longer instructions, an average of 120 words per instruction, and annotated paths in RxR-CE are much longer than those in R2R-CE with an average length of 15.32 m. To evaluate generalization, the val-seen and val-unseen splits are commonly used. Both splits include navigation instructions not seen during training. The main difference lies in the environments: val-seen environments appear in the training set, while val-unseen does not.

Metrics: We adopt the following navigation metrics from previous works. Navigation Error (NE): average geometric distance

TABLE I
COMPARISON ON R2R-CE AND RXR-CE UNDER THE *VARIED VIEWPOINT* AND *GROUND-LEVEL VIEWPOINT* SETTINGS. THE *VARIED VIEWPOINT* SETTING CORRESPONDS TO OUR PROPOSED V^2-VLNCE SETUP. THE *GROUND-LEVEL VIEWPOINT* SETTING IS ADAPTED FROM GVNAV [13]. EVALUATION METRICS ARE CONSISTENT WITH GVNAV. **BOLD** INDICATES PERFORMANCE IMPROVEMENTS INTRODUCED BY VIL.

Method	val-seen					val-unseen				
	NE↓	nDTW↑	OSR↑	SR↑	SPL↑	NE↓	nDTW↑	OSR↑	SR↑	SPL↑
R2R-CE, *Ground-level Viewpoint* [13]										
HPN [31] [ICCV2021]	6.30	57	43	37	35	6.79	54	35	30	28
CMA [7] [CVPR2022]	5.99	55	58	44	38	6.68	49	50	37	30
VLN⟳BERT [7] [CVPR2022]	5.46	55	56	47	39	6.25	50	49	39	33
GVNav [13] [ICRA2025]	3.88	66	70	64	56	4.89	58	62	55	45
BEVBert [27] [ICCV2023]	3.26	70	76	70	62	4.63	61	67	59	49
BEVBert + VIL (Ours)	**3.16**	**71**	**77**	**71**	**63**	**4.61**	**62**	66	59	**50**
ETPNav [6] [TPAMI2024]	4.48	62	71	62	50	5.27	55	63	52	42
ETPNav + VIL (Ours)	**4.02**	**67**	**71**	**64**	**55**	**4.91**	**59**	**65**	**57**	**47**
R2R-CE, *Varied Viewpoint (Ours)*										
HPN [31] [ICCV2021]	6.32	57	43	35	33	6.76	54	35	29	27
CMA [7] [CVPR2022]	6.59	49	45	32	27	6.91	46	40	28	23
VLN⟳BERT [7] [CVPR2022]	5.93	52	50	39	34	6.39	48	44	32	27
VLN-3DFF [9] [CoRL2024]	5.59	49	54	42	32	6.12	45	54	41	31
g3D-LF [32] [CVPR2025]	5.06	58	57	51	41	5.26	56	57	50	40
BEVBert [27] [ICCV2023]	4.48	61	65	57	47	5.32	56	58	49	39
BEVBert + VIL (Ours)	**3.91**	**67**	**70**	**63**	**55**	**5.15**	**58**	**62**	**52**	**44**
ETPNav [6] [TPAMI2024]	5.16	59	58	49	42	5.58	55	55	47	38
ETPNav + VIL (Ours)	**4.02**	**66**	**69**	**64**	**55**	**4.90**	**59**	**61**	**55**	**45**
RxR-CE, *Varied Viewpoint (Ours)*										
ETPNav [6] [TPAMI2024]	8.07	50	49	40	31	7.82	49	48	39	31
ETPNav + VIL (Ours)	**5.99**	**63**	**62**	**55**	**46**	**6.42**	**59**	**57**	**50**	**41**

in meters between the final and target location; Success Rate (SR): the ratio of paths with NE less than 3 meters; Oracle SR (OSR) [1]: SR given an oracle stop policy; SR penalized by Path Length (SPL) [29]; Normalized Dynamic Time Wrapping (nDTW) [30]: a normalized DTW score between predicted and ground-truth paths; Success weighted normalized Dynamic Time Warping (SDTW) [30]: nDTW weighted by success.

Implementation details: All models are initialized from the official pretrained checkpoints of the corresponding backbones, and all hyperparameters of the backbone models follow their original implementations, while only a small set of VIL-specific hyperparameters is introduced. The hyperparameters specific to VIL are: sampling probabilities $p_1 = p_2 = 0.1$, contrastive learning loss weight $\lambda_1 = 0.2$, and waypoint predictor distillation loss weight $\lambda_2 = 10.0$.

A. Performance Under Varied Viewpoints

The *Varied Viewpoint* protocol corresponds to our proposed V^2-VLNCE setting. Concretely, each viewpoint is defined by a height-angle pair (h, θ) sampled from a uniform distribution $\mathcal{U}([-0.5\text{m}, 0.5\text{m}]) \times \mathcal{U}([-30°, 30°])$, relative to the standard VLNCE configuration. This generalized setup better reflects real-world differences and tests model robustness to viewpoint shifts. As a baseline, we include GVNav [13], which introduces a *Ground-level Viewpoint* setting by lowering the camera height to 0.8 meters. Although GVNav does not account for angles and uses a fixed-height viewpoint, it is the only prior work that explicitly investigates viewpoint shift in VLNCE. We therefore consider it a relevant setting.

Performance on R2R-CE: Table I shows that applying VIL substantially improves performance under the *Varied Viewpoint* setting. For example, ETPNav + VIL achieves significant gains over the base ETPNav model on both val-seen and val-unseen splits. Specifically, our model improves NE by 0.68-1.14, nDTW by 3%–7%, OSR by 6%–9%, SR by 8%–15%, and SPL by 7%–13%. Similarly, compared to BEVBert, our method shows consistent improvement across all metrics. These results demonstrate the effectiveness of VIL in promoting viewpoint-robust navigation behavior. Moreover, compared to GVNav, a method specifically designed for *Ground-level Viewpoint*, ETPNav + VIL still performs better on val-unseen (e.g., +3% OSR, +2% SR, +2% SPL). This suggests that our method can be generalized to *Ground-level Viewpoint* as well, even without training on matched *Ground-level Viewpoint* samples.

Performance on RxR-CE: The RxR-CE dataset is much larger than R2R-CE, providing a stronger test of model scalability. As shown in Table I, applying VIL under the *Varied Viewpoint* setting yields clear improvements. ETPNav + VIL outperforms the base ETPNav on both val-seen and val-unseen, improving nDTW by 10%-13%, OSR by 9%-13%, SR by 11%-15%, and SPL by 10%-15%.

B. Performance Under the Standard Viewpoint

Although VIL is trained with varied viewpoints, it does not degrade performance under the standard VLN-CE setting. As shown in Table II, VIL maintains or slightly improves navigation metrics compared to the base models, demonstrating robustness to viewpoint variations. On R2R-CE val-unseen, ETPNav + VIL increases SR by 1.5 and SPL by 0.8, while on RxR-CE val-unseen, ETPNav + VIL improves SPL by 2.3. These results show that VIL generalizes well to standard viewpoints, confirming that training with varied viewpoints does not compromise, and can

TABLE II
PERFORMANCE UNDER STANDARD VIEWPOINT. METRICS: NE↓, SR↑, SPL↑. BOLD INDICATES IMPROVEMENT FROM VIL.

Method	val-seen			val-unseen		
	NE↓	SR↑	SPL↑	NE↓	SR↑	SPL↑
R2R-CE						
VLN⟳BERT [7] [CVPR2022]	5.02	50	44	5.74	44	39
ENP [33] [NeurIPS2024]	3.90	68	59	4.69	58	50
NaVILA [34] [RSS2025]	-	-	-	5.22	54	49
BEVBert [27] [ICCV2023]	3.24	70.9	62.8	4.63	59.1	49.2
BEVBert + VIL (Ours)	**3.16**	**71.1**	**63.0**	**4.61**	58.6	**49.6**
ETPNav [6] [TPAMI2024]	3.97	65.8	59.2	4.78	56.8	48.9
ETPNav + VIL (Ours)	**3.71**	**67.6**	**60.4**	**4.69**	**58.3**	**49.7**
RxR-CE						
VLN⟳BERT [7] [CVPR2022]	-	-	-	8.98	27.1	22.7
ENP [33] [NeurIPS2024]	5.10	62.0	51.2	5.51	55.3	45.1
NaVILA [34] [RSS2025]	-	-	-	6.77	49.3	44.0
ETPNav [6] [TPAMI2024]	5.39	60.0	49.1	5.96	53.8	43.9
ETPNav + VIL (Ours)	**4.89**	**63.5**	**53.0**	**5.62**	**55.6**	**46.2**

even slightly enhance, performance in the original setting. Moreover, the table also includes state-of-the-art map-free methods, where map-free means that no pre-exploration of environments is used. On RxR-CE, our method not only improves over the base model but also outperforms these state-of-the-art map-free methods across all metrics.

C. Ablation Study

We conduct an ablation study in Table III to evaluate the contribution of three components: exposing the model to *Varied Viewpoint* data (retrain), contrastive learning (CL), and waypoint predictor distillation (WPD). **Is standard fine-tuning sufficient?** Retraining on *Varied Viewpoint* improves varied viewpoint SPL slightly (+0.7), but can slightly harm standard viewpoint SPL (-1.2). **Effect of WPD.** Removing WPD degrades performance substantially, e.g., val-unseen SPL drops by 8.3 (*Varied Viewpoint*) and 4.1 (*Standard Viewpoint*) **Effect of CL.** Contrastive learning improves val-unseen SPL by 2.8 (*Varied Viewpoint*) and enhances generalization under the standard viewpoint, increasing SR and SPL by 0.7 and 1.0 respectively. These results show that each component contributes to better navigation, particularly in unseen or varied viewpoint settings. We further confirm the same trends on the larger RxR-CE dataset, though we omit the detailed table due to space constraints.

D. Viewpoint Robustness Analysis

Variance across viewpoint changes: We evaluate robustness by sampling 81 fixed viewpoints covering height and angle variations. To quantify this consistency, we compute the standard deviation of each metric across the 81 configurations. As in Table IV, our method substantially reduces variance compared to the baseline (e.g., SPL std drops by 65%), demonstrating that VIL not only improves average performance, but also stabilizes behavior under spatial perturbation.

Evaluation with fixed real-robot camera placements: To further examine viewpoint robustness under realistic configurations, we evaluate navigation performance using camera placements corresponding to three robots: Stretch RE-1, Stretch RE-1 (Factory), and LoCoBot. Only the camera placement is adopted, not the robot hardware, as our inputs are 360° RGB-D images. Both ETPNav and ETPNav+VIL are evaluated on the full val-seen and val-unseen splits.

VIL consistently improves navigation performance across all robot placements. For example, SPL on val-unseen increases by over 20% for Stretch RE-1, demonstrating that VIL generalizes effectively to diverse camera configurations.

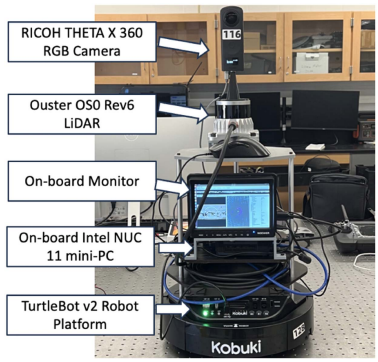

Fig. 4. The robot platform used in our experiments.

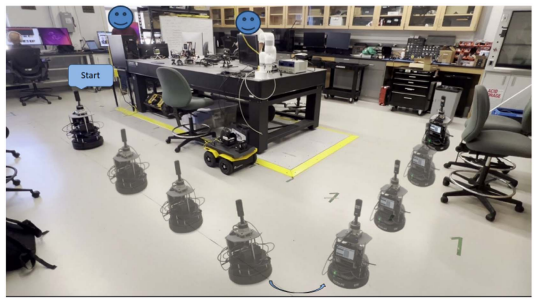

Fig. 5. Real world demo of our proposed VIL.

E. Out-of-Distribution Viewpoint Generalization

In addition to robustness within the training range, we study extrapolation to unseen viewpoints. We compare two training distributions: (i) a *large* range $\mathcal{U}([-0.5\,\mathrm{m}, 0.5\,\mathrm{m}]) \times \mathcal{U}([-30^\circ, 30^\circ])$, where the test viewpoints lie on the distribution boundary (thus still in-distribution), and (ii) a *small* range $\mathcal{U}([-0.4\,\mathrm{m}, 0.4\,\mathrm{m}]) \times \mathcal{U}([-20^\circ, 20^\circ])$, where the same test viewpoints fall outside the training support (true OOD). We evaluate on two extreme configurations, $(-0.5\,\mathrm{m}, 30^\circ)$ and $(0.5\,\mathrm{m}, -30^\circ)$.

Across both OOD configurations, VIL outperforms the baseline by a large margin. Notably, even when trained on the reduced viewpoint range, VIL improves SPL by $+13.3$ on $(-0.5\,\mathrm{m}, 30^\circ)$ and $+6.3$ on $(0.5\,\mathrm{m}, -30^\circ)$ (val-unseen). Compared with the large-range model, the small-range model shows only a slight drop in performance, indicating that VIL maintains robustness even with limited viewpoint diversity during training.

F. Real-Robot Evaluation

We further validate VIL in real-world settings using a TurtleBot v2 platform. The robot is equipped with a RICOH THETA X 360 RGB camera, a Ouster OS0 Rev6 LiDAR, and an onboard

TABLE III
ABLATION STUDY ON R2R-CE WITH *VARIED VIEWPOINT* AND *STANDARD VIEWPOINT* SETTING. THE BEST PERFORMANCE FOR EACH METRIC IS HIGHLIGHTED IN **BOLD**, AND THE SECOND-BEST IS UNDERLINED. ALL ABLATION SETTINGS USE THE SAME TRAINING CONFIGURATIONS, INCLUDING BATCH SIZE AND TOTAL TRAINING STEPS. CL: CONTRASTIVE LEARNING, WPD: WAYPOINT PREDICTOR DISTILLATION.

Method	retrain	CL	WPD	*Varied Viewpoint*						*Standard Viewpoint*					
				val-seen			val-unseen			val-seen			val-unseen		
				NE↓	SR↑	SPL↑	NE↓	SR↑	SPL↑	NE↓	SR↑	SPL↑	NE↓	SR↑	SPL↑
ETPNav	×	×	×	5.16	49.4	42.0	5.58	46.8	38.4	3.97	65.8	59.2	4.78	56.7	48.9
	✓	×	×	4.52	54.6	46.0	5.21	49.8	39.1	3.62	66.1	57.6	4.66	57.9	47.7
	✓	✓	×	4.55	56.4	45.3	5.09	49.7	37.2	3.72	67.0	56.8	4.64	58.2	45.6
	✓	×	✓	4.27	59.6	51.8	5.01	52.6	42.7	**3.51**	**68.5**	**60.5**	**4.63**	57.6	48.7
	✓	✓	✓	**4.02**	**63.6**	**54.9**	**4.90**	**54.6**	**45.5**	3.71	67.6	60.4	4.69	**58.3**	**49.7**

TABLE IV
STANDARD DEVIATION σ FOR ALL METRICS ACROSS 81 VIEWPOINTS ON R2R-CE VAL-UNSEEN

Model	σ_{NE}	σ_{nDTW}	σ_{OSR}	σ_{SR}	σ_{SPL}
ETPNav	0.82	7.83	9.12	10.54	10.79
+ VIL	**0.28**	**2.43**	**3.42**	**3.66**	**3.59**

TABLE V
NAVIGATION PERFORMANCE USING FIXED CAMERA PLACEMENTS FROM REAL ROBOTS. METRICS: NE↓, SR↑, SPL↑.

Robot	Method	val-seen			val-unseen		
		NE	SR	SPL	NE	SR	SPL
StrRE-1	ETPNav	7.50	21.9	14.5	7.08	23.1	15.5
	+VIL	5.37	47.8	40.1	5.83	43.7	35.4
StrRE-1 (Fac)	ETPNav	7.38	23.8	16.1	7.09	22.7	14.8
	+VIL	5.57	47.9	39.6	6.00	40.3	32.0
LoCoBot	ETPNav	4.38	61.3	51.6	5.17	54.2	42.8
	+VIL	3.80	66.7	56.4	4.91	56.0	45.7

TABLE VI
PERFORMANCE UNDER OOD VIEWPOINTS ON R2R-CE VAL-SEEN/UNSEEN

Config	Method	val-seen			val-unseen		
		NE	SR	SPL	NE	SR	SPL
$(-0.5\text{m}, 30°)$	ETPNav	6.51	36.2	26.2	6.30	36.3	25.9
	+VIL (large)	4.69	54.8	46.4	5.18	50.2	41.0
	+VIL (small)	4.78	54.0	45.6	5.43	47.5	39.2
$(0.5\text{m}, -30°)$	ETPNav	5.30	51.7	42.5	5.50	46.6	37.9
	+VIL (large)	4.13	61.2	52.7	5.14	52.3	43.7
	+VIL (small)	4.09	61.7	54.3	5.14	52.1	44.2

Intel NUC 11 mini-PC (i7-1165G7 CPU, 8 GB RAM). The sensors are extrinsically calibrated following [35] to align LiDAR and camera frames. This setup yields 12 aligned RGB-D views covering 360°. Unlike prior work such as GVNav [13], which relies on rotating a monocular RGB-D camera to synthesize panoramic inputs, our design directly produces 360° RGB-D observations by fusing a panoramic RGB sensor with a 360° LiDAR.

The client robot collects RGB-D observations and communicates with a remote server via ROS 2 over a VPN. On the server, our model processes incoming images in real time using an NVIDIA A5000 GPU and outputs navigation actions, which are transmitted back for execution. The real-world experiment is a zero-shot evaluation: the developed model is trained entirely in simulation, using the varied-viewpoint R2R-CE setup. Specifically, the simulation training distribution covered camera heights between 0.75 m and 1.75 m, while the real robot's camera was measured at 0.7 m, representing an out-of-distribution embodiment.

TABLE VII
REAL-ROBOT EVALUATION IN TWO ENVIRONMENTS. WE REPORT SUCCESS RATE (SR) BEFORE AND AFTER APPLYING VIL.

Environment	SR (ETPNav)	SR (+VIL)
Office	28	**44**
Lounge	20	**48**

We evaluate in two indoor environments: an office and a lounge. Each setting includes 5 instructions, repeated 5 times from different starting locations. Results in Table VII show that VIL improves navigation robustness across environments.

These results confirm that VIL consistently enhances the robustness of navigation in real-world deployments, supporting its practicality for embodied agents beyond simulation.

G. Computational Efficiency

Beyond performance improvements, we examine the training and inference cost of VIL compared to the baseline. While ETPNav requires extensive pre-training and fine-tuning stages (around 11.5 days in total), VIL post-training converges in only 48 hours. This corresponds to roughly 14% of the full training time. The full VIL model has 335.21 M total parameters with 143.21 M trainable parameters, while the corresponding baseline has 317.31 M total parameters with 142.93 M trainable parameters. The difference is marginal, confirming that the additional modules introduced by VIL are lightweight.

We also observe that the peak GPU memory usage increases only marginally (from $\sim$ 6000 MB to 6200–6300 MB with the same batch size). At inference, the overhead is negligible since VIL adds only a single linear projection, resulting in no distinguishable difference in per-step runtime. These results confirm that VIL is both training-efficient and deployment-friendly, making it practical for real-world navigation.

V. Conclusion

We introduced V^2-VLNCE, a varied-viewpoint scenario to evaluate robustness of VLNCE policies. To address viewpoint sensitivity, we proposed View Invariant Learning (VIL), which improves generalization in both V^2-VLNCE and standard VLNCE. Real-robot experiments further confirm its effectiveness, showing that VIL is a practical solution for VLNCE.

Acknowledgment

The authors thank Laura McCrackin (University of Waterloo), Yurun Chen (Shanghai Jiao Tong University and Eastern Institute of Technology, Ningbo), Xinzhu Fu (National University of Singapore), Rui Wang (Hohai University), Jiangran Lyu (Peking University), and Tianyi Hu for helpful discussions.

References

[1] P. Anderson et al., "Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3674–3683.

[2] J. Krantz, E. Wijmans, A. Majumdar, D. Batra, and S. Lee, "Beyond the nav-graph: Vision-and-language navigation in continuous environments," in *Proc. 16th IEEE Eur. Conf. Comput. Vis.*, 2020, pp. 104–120.

[3] Y. Hong, Q. Wu, Y. Qi, C. Rodriguez-Opazo, and S. Gould, "VLN BERT: A recurrent vision-and-language BERT for navigation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 1643–1653.

[4] G. Georgakis et al., "Cross-modal map learning for vision and language navigation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 15460–15 470.

[5] Z. Wang, X. Li, J. Yang, Y. Liu, and S. Jiang, "GridMM: Grid memory map for vision-and-language navigation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 15625–15636.

[6] D. An et al., "ETPNav: Evolving topological planning for vision-language navigation in continuous environments," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 47, no. 7, pp. 5130–5145, Jul. 2025.

[7] Y. Hong, Z. Wang, Q. Wu, and S. Gould, "Bridging the gap between learning in discrete and continuous environments for vision-and-language navigation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 15439–15 449.

[8] Z. Wang et al., "Lookahead exploration with neural radiance representation for continuous vision-language navigation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 13753–13 762.

[9] Z. Wang, X. Li, J. Yang, Y. Liu, and S. Jiang, "Sim-to-real transfer via 3D feature fields for vision-and-language navigation," in *Proc. 8th Annu. Conf. Robot Learn.*, 2025, pp. 2982–2995.

[10] Y. Zhang and P. Kordjamshidi, "Narrowing the gap between vision and action in navigation," in *Proc. 32nd ACM Int. Conf. Multimedia*, 2024, pp. 856–865.

[11] Y. Seo, J. Kim, S. James, K. Lee, J. Shin, and P. Abbeel, "Multi-view masked world models for visual robotic manipulation," in *Proc. 40th Int. Conf. Mach. Learn.*, 2023, pp. 30613–30632.

[12] F. Liu, F. Yan, L. Zheng, C. Feng, Y. Huang, and L. Ma, "Robouniview: Visual-language model with unified view representation for robotic manipulation," 2024, *arXiv:2406.18977*.

[13] Z. Li et al., "Ground-level viewpoint vision-and-language navigation in continuous environments," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2025, pp. 5266–5273.

[14] J.-C. Pang et al., "Learning view-invariant world models for visual robotic manipulation," in *Proc. 13th Int. Conf. Learn. Representations*, 2025.

[15] Y. Hong et al., "Learning navigational visual representations with semantic map supervision," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 3055–3067.

[16] M. Z. Irshad, N. C. Mithun, Z. Seymour, H.-P. Chiu, S. Samarasekera, and R. Kumar, "Semantically-aware spatio-temporal reasoning agent for vision-and-language navigation in continuous environments," in *Proc. 26th Int. Conf. Pattern Recognit.*, 2022, pp. 4065–4071.

[17] L. Yue, D. Zhou, L. Xie, F. Zhang, Y. Yan, and E. Yin, "Safe-VLN: Collision avoidance for vision-and-language navigation of autonomous robots operating in continuous environments," *IEEE Robot. Automat. Lett.*, vol. 9, no. 6, pp. 4918–4925, Jun. 2024.

[18] Z. Wang et al., "Scaling data generation in vision-and-language navigation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 12009–12020.

[19] Z. Wang et al., "Bootstrapping language-guided navigation learning with self-refining data flywheel," in *Proc. 13th Int. Conf. Learn. Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=OUuhwVsk9Z

[20] H. Wang, W. Liang, L. Van Gool, and W. Wang, "Dreamwalker: Mental planning for continuous vision-language navigation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 10873–10 883.

[21] S. Raychaudhuri, S. Wani, S. Patel, U. Jain, and A. Chang, "Language-aligned waypoint (LAW) supervision for vision-and-language navigation in continuous environments," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2021, pp. 4018–4028.

[22] J. Zhang et al., "NaVid: Video-based VLM plans the next step for vision-and-language navigation," in *Proc. Robot.: Sci. Syst. (RSS)*, 2024.

[23] J. Zhang et al., "Uni-NaVid: A video-based vision-language-action model for unifying embodied navigation tasks," in *Proc. Robot.: Sci. Syst. (RSS)*, 2025.

[24] A.-C. Cheng et al., "Navila: Legged robot vision-languageaction model for navigation," in *Proc. Robot.: Sci. Syst. (RSS)*, 2025.

[25] A. Eftekhar et al., "The one ring: A robotic indoor navigation generalist," 2024, *arXiv:2412.14401*.

[26] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton, "Big self-supervised models are strong semi-supervised learners," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, vol. 33, pp. 22243–22255 .

[27] D. An et al., "BEVBert: Multimodal map pre-training for language-guided navigation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023.

[28] A. Ku, P. Anderson, R. Patel, E. Ie, and J. Baldridge, "Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2020, pp. 4392–4412.

[29] P. Anderson et al., "On evaluation of embodied navigation agents," 2018, *arXiv:1807.06757*.

[30] G. I. Magalhaes, V. Jain, A. Ku, E. Ie, and J. Baldridge, "General evaluation for instruction conditioned navigation using dynamic time warping," in *Proc. NeurIPS Visually Grounded Interact. Lang. Workshop*, vol. 1, 2019.

[31] J. Krantz, A. Gokaslan, D. Batra, S. Lee, and O. Maksymets, "Waypoint models for instruction-guided navigation in continuous environments," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 15162–15 171.

[32] Z. Wang and G. H. Lee, "g3D-LF: Generalizable 3D-language feature fields for embodied tasks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2025, pp. 14191–14 202.

[33] R. Liu, W. Wang, and Y. Yang, "Vision-language navigation with energy-based policy," in *Proc. 38th Annu. Conf. Neural Inf. Process. Syst.*, 2024, pp. 108208–108230.

[34] A.-C. Cheng et al., "NaVILA: Legged robot vision-language-action model for navigation," 2025, *arXiv:2412.04453*.

[35] K. Koide, S. Oishi, M. Yokozuka, and A. Banno, "General, single-shot, target-less, and automatic LiDAR-camera extrinsic calibration toolbox," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2023, pp. 11301–11 307.