

monoVLN: Bridging the Observation Gap between Monocular and Panoramic Vision and Language Navigation

Renjie Lu^{1*}, Yu Zhou^{1*}, Hao Cheng², Jingke Meng^{1†}, Wei-Shi Zheng^{1,3,4}

¹School of Computer Science and Engineering, Sun Yat-sen University, ²Hunan University,

³Pengcheng Lab, ⁴Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education, China
{lurj3, zhouy635}@mail2.sysu.edu.cn, haocheng@hnu.edu.cn, mengjke@gmail.com, wszheng@ieee.org

Abstract

Vision and Language Navigation (VLN) requires agents to navigate 3D environments by following natural language instructions. While existing methods predominantly assume access to panoramic observations, many practical robotics are equipped with monocular RGBD cameras, creating a significant configuration disparity. In this work, we address this critical gap by developing a novel 3DGS-based framework for monocular VLN agents, focusing on the intrinsic information incompleteness challenge. Our approach incorporates two key innovations: (1) implicit partial completion module for inferring representations of missing regions in incompletely rendered panoramic feature maps, and (2) an uncertainty-aware active perception strategy that enables the agent to actively acquire visual observation when uncertain about its decision. Extensive experiments on R2R-CE and RxR-CE datasets demonstrate that our monoVLN outperforms all existing monocular methods, significantly improve 8% success rate on R2R-CE compared to previous monocular methods. We also validate our monoVLN in real-world environments, providing a practical solution for real-world VLN.

1. Introduction

Vision and Language Navigation (VLN) [5, 27] involves guiding an agent to navigate to a specified destination by interpreting natural language instructions. This process typically requires the agent to execute fine-grained actions (e.g., rotating 15 degrees left/right or advancing 0.25 meters) within navigable areas. VLN has attracted significant research interests [3, 10, 11, 13, 17, 20, 21, 34, 43, 46, 50, 52].

Current researches in VLN predominantly focus on navigation utilizing panoramic observations, which assumes that the 360-degree area surrounding the agent is fully ac-

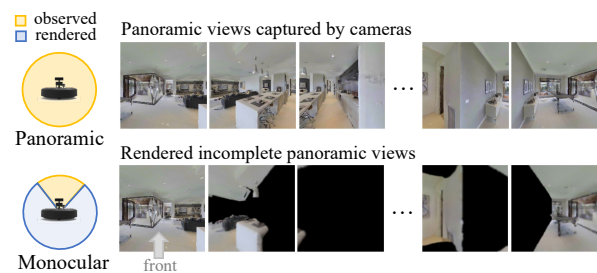


Figure 1. An example of panoramic views captured by cameras and rendered panoramic views. The rendered views are incomplete due to unobserved areas during navigation, resulting in certain regions being rendered with black.

cessible. These methods achieves notable performance but facing significant limitations in real-world robotic deployment. The primary constraint stems from the prevalent use of monocular RGBD cameras in practical robotic applications, which restrict the observable area to a limited field of view around the agent at a step. Consequently, the development of models compatible with monocular camera is imperative for enhancing practicality and compatibility in real-world scenarios. Recent methods [46, 50] address this challenge by reconstructing 3D feature fields that integrate previously observed information and rendering panoramic views, enabling the direct transfer of panoramic VLN models to monocular settings and achieving impressive results. However, they ignore a fundamental challenge inherent to monocular VLN: the intrinsic information incompleteness resulting from the limited field of view. As illustrated in Figure 1, the rendered views by works [46, 50] are often incomplete, which prevents visual encoders from extracting comprehensive features and leads to suboptimal performance. *In this work, we present the first attempt to solve this problem.*

An intuitive solution is to perform a 360-degree rotation at each step to mimic panoramic observations. While this method is conceptually simple, it introduces significant inefficiencies, leading to an increase of over 200% in the num-

* These authors contributed equally to this work.

† denotes the corresponding authors.

ber of navigation steps required. To alleviate this, drawing inspiration from human navigation, we first attempt to leverage the agent’s subjective perspective, presenting a simple yet effective active perception strategy, enabling it to actively acquire visual observations when uncertain about its decision. Specifically, we focus on two key challenges: determining *when to look*, which involves identifying the moments for the agent to gather observations, and *where to look*, which involves selecting a informative direction for visual data collection. This strategy can potentially reduce severe incompleteness shown in Figure 1, where nearly all regions appear black in a given view while avoiding many extra steps.

For the partial absence of the observation within a view, we further propose an implicit partial completion module. This module leverages contextual feature relationships to infer representations of missing regions rendered from feature field, enabling the extraction of comprehensive features for partially incomplete views. Specifically, to infer features for missing regions, we leverage the MAE framework, which operates on partial features. The view encoder takes rendered partial feature maps as input and generates corresponding latent representations. These latent features are then decoded to reconstruct complete feature maps, guided by the auxiliary loss inspired by MAE[18]. Through this reconstruction-based training paradigm, the decoder learns to predict full feature maps from partial inputs, enabling the view encoder to implicitly infer missing visual information.

Based on the above, we propose monoVLN, a new 3DGS-based VLN framework designed for practical robotic applications using monocular RGB-D cameras. Unlike previous works [46, 50], which rely on NeRF for constructing 3D feature fields, we employ 3D Gaussian Splatting(3DGS). This technique enables fast rendering and a training-free feature field, significantly improving efficiency. We conduct extensive experiments on the widely used R2R-CE[27] and RxR-CE[29] datasets. Our proposed model significantly improves 8% success rate compared to previous monocular methods on R2R-CE, while only including 9% extra steps. To validate practical applicability, we deploy monoVLN on real-world robotic platforms, where it achieves fast inference on RTX 4060 GPU, highlighting its potential for real-world application.

In summary, our contributions are as follows:

- We propose monoVLN, a 3DGS-based framework for monocular vision and language navigation, enabling efficiency and effectiveness.
- We present the first attempt to tackle the information incompleteness problem in monocular VLN by introducing implicit completion and active perception.
- Our model achieves a new state-of-the-art on R2R-CE and RxR-CE datasets, with an 8% higher success rate on R2R-CE compared to previous monocular VLN methods.

2. Related Works

Vision and Language Navigation. The field of natural language-guided navigation in unseen environments[5, 27, 29] has seen substantial progress in recent years. A primary focus has been on architectural innovations, particularly in developing advanced decision-making backbones[10, 20] and enhancing vision modeling techniques[1, 2, 19, 32]. Building upon these foundations, various planning strategies have been proposed, including self-correction mechanisms[25, 35], graph-level planning[11, 34, 44] and hierarchical frameworks[14]. Complementing these advances, visual data augmentation methods[30, 31, 43] have emerged for improving visual generalization ability. Other line of work focus on enhancing visual-language representation, which leverage back-translation-based data generation[13, 24, 45] and domain-specific pretraining techniques[16, 17, 36, 38, 39]. More recently, [47, 48] focus on scaling data generation, approaching near human performance on the well-recognized R2R benchmark.

VLN in Continuous Environments. Despite the remarkable progress in VLN, these agents are developed on discrete predefined topological graphs, making it impractical to real-world robot applications. To bridge the gap, VLN-CE[27] pioneers continuous environment navigation by constructing the R2R-CE dataset. Early methods[8, 15, 27, 28, 41] on R2R-CE demonstrated significantly inferior performance compared to their discrete VLN counterparts. Subsequent studies[21, 26] analyze the difference between discrete and continuous VLN and developed frameworks that transfers discrete VLN agents to continuous domains, substantially reducing the gap between these two paradigms. Recently, ETPNav[3] enhances the framework through improved waypoint predictor and topological mapping. HNR[49] introduces a feature field representation combined with an expanded frontier-based action space.

Real World VLN Models. Both discrete and continuous models are trained and evaluated in simulators. [6] first deploy VLN models in real world. VLMap [22] introduces a framework that constructs BEV maps and use LLMs to generate high-level navigation code. To enable agents to comprehend free-form instructions and avoid noisy depth in real-world scenarios, NaVid [52] utilizes a video-based LLM to process RGB frames and predict actions in text form. While VLN agents are predominantly developed with panoramic observations, recent work 3DFF[50] transfers panoramic VLN models to monocular observations via feature field, g3D-LF[46] further enhances their quality. However, these approaches overlook the inherent information incompleteness of monocular observations. Our work presents the first attempt to tackle this challenge through active perception and implicit inference of complete views, effectively mitigating the issue.

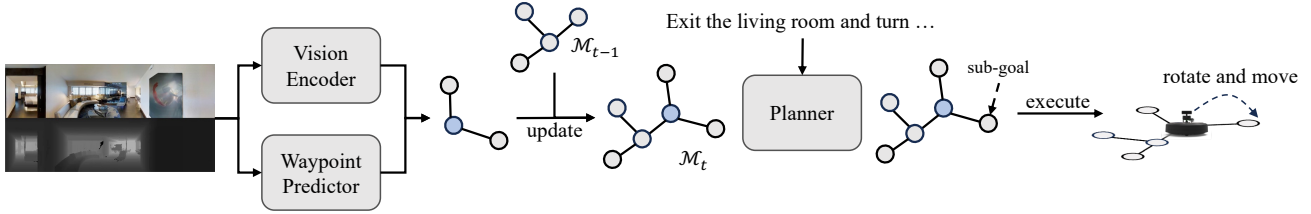


Figure 2. Typical panoramic VLN model pipeline. The blue node represents the current location, while the gray nodes represent other positions in the environment. The planner predicts a sub-goal on the graph through node classification, guiding the agent to move toward the predicted node. If the target node is a direct neighbor of the current node, the agent simply rotates and moves to it. Otherwise, the agent navigates to the target node by following the shortest path on the graph.

3. Method

3.1. VLN Background

Problem Formulation. In VLN, an agent is provided with a natural language instruction \mathcal{I} describing the navigation task. At time step t , the agent observes an RGB image r_t and a depth image d_t captured by a monocular RGB-D camera, and its current 4D pose $p_t = (x_t, y_t, z_t, \theta_t)$, where (x_t, y_t, z_t) represents the agent’s coordinates and θ_t denotes its heading direction. The agent selects an action a_t from a discrete action space $\mathcal{A} = \{\text{turn left } 15^\circ, \text{turn right } 15^\circ, \text{move forward } 0.25\text{m}, \text{stop}\}$. The navigation task is considered successful when the agent stops at the target location within 3 meters.

VLN Models with Panoramic Views. In this part, we outline the general pipeline of panoramic VLN models. As illustrated in Figure 2, typical panoramic VLN framework facilitates navigation by constructing a topological map derived from panoramic views and planning paths on this map using visual observations and natural language instructions. Specifically, at step t , the agent observes panoramic RGB images $\mathcal{R}_t = \{r_{t,i}\}_{i=1}^{12}$ and panoramic depths $\mathcal{D}_t = \{d_{t,i}\}_{i=1}^{12}$ towards 12 different orientations with 30 degrees between each. The RGB images \mathcal{R}_t are encoded into features $\{f_{t,i}^v\}_{i=1}^{12}$, which are stored in nodes[3] or edges[34] of the topological map \mathcal{M}_t . The depth images \mathcal{D}_t are used to predict the coordinates of nearby navigable waypoints[3, 21, 34], enabling the update of the topological map from \mathcal{M}_{t-1} to \mathcal{M}_t . A planner then encodes the topological map \mathcal{M}_t along with the instruction \mathcal{I} and selects the optimal waypoint as the sub-goal to move. With the predicted sub-goal, the agent navigates to it along the shortest path on \mathcal{M}_t . If it fails to reach the sub-goal due to obstacles, we stop it immediately.

3.2. Framework Overview

We present the overall framework of our monoVLN in Figure 3, designed for practical VLN using monocular RGB-D cameras. Our framework primarily addresses the challenge of intrinsic information incompleteness caused by the limited field of view in monocular observations. Specif-

ically, with the monocular RGB-D observation, we incrementally construct a 3D feature field by leveraging the feature 3DGS [53]. This feature field is used for two functionalities: first, rendering an agent-centric top-down view feature map (BEV feature) input to a waypoint predictor to predict the coordinates of nearby navigable waypoints. Second, rendering panoramic feature maps processed by an implicit partial completion module to enable comprehensive panoramic understanding by inferring missing regions in rendered views. With the rendered panoramic features and predicted waypoints, we construct a topological map and perform decision planning, following the general pipeline of VLN models[3, 11, 21, 34]. When the agent encounters uncertainty in its decision-making process, it selects an informative directions and actively acquires visual observations to refine its decisions. Details are presented below.

3.3. 3DGS-based Feature Field Construction

In this section, we describe how we construct 3DGS feature field to effectively convert monocular observation to panoramic observation and enable transferring powerful panoramic VLN model to the monocular setting. By reconstructing the feature field during navigation and rendering panoramic views using it, the rendered panoramic views provide richer information compared to monocular observations since the reconstructed feature field integrates historical observations. We opt for a 3DGS-based feature field as it permits us to directly convert a point cloud into Gaussians *without training*. This choice allows us to concentrate on the core navigation task without diverting attention to scene reconstruction.

Specifically, at time step t , we utilize MobileSAM[51] to extract a feature map $f_t^r \in \mathbb{R}^{C \times H \times W}$ for the monocular RGB observation r_t . Using downsampled depth image d_t , the feature map f_t^r is mapped to 3D world position through the camera pose and intrinsics, resulting in a point cloud where each point is associated with color and feature information. To transform the point cloud into a feature-based 3DGS [53], we transfer the attributes of the points to Gaussians. Each Gaussian has six attributes: coordinates, color, scale, rotation, opacity, and feature. We migrate the coordi-

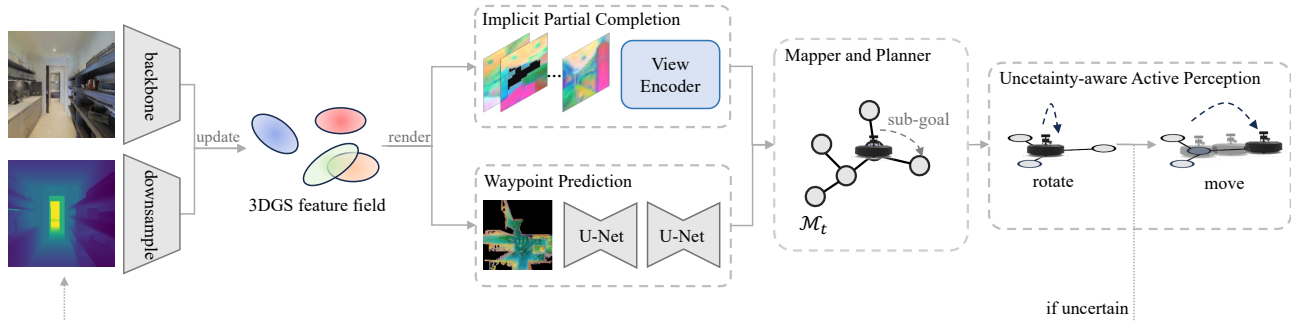


Figure 3. The overall framework of our method. We process observed RGB-D images to update the feature field for rendering panoramic feature maps and a BEV map, which are then processed by our Implicit Partial Completion(IPC) module and BEV-based waypoint predictor. Their predictions are fed into mapper and planner to predict a sub-goal. The robot moves toward this sub-goal or only rotates to achieve Uncertainty-aware Active Perception(UAP) based on its prediction uncertainty.

nates, color, and feature directly from the point cloud while setting the remaining attributes to fixed values. This allows us to treat 3DGS as a point cloud rendering method rather than its conventional use for scene reconstruction. This simple method provides acceptable rendering quality *without the need for training*.

3.4. BEV-based Waypoint Prediction

Panoramic VLN models are predominantly designed using topological maps for global planning and backtracking[11, 14, 34]. These approaches typically rely on a waypoint predictor to estimate the coordinates of nearby waypoints for constructing such maps[3, 21]. While these predictors are built upon panoramic depth images, we present a waypoint predictor for the monocular setting that utilizes a top-down view feature map (BEV feature) to accomplish this, where the BEV can be directly derived from the feature field.

As illustrated in Figure 3, the BEV feature is extracted from the feature field. To generate this map, we position the camera considerably higher above the agent and render an agent-centric BEV feature map $f_t^b \in \mathbb{R}^{C \times 192 \times 192}$. Each pixel in the BEV map corresponds to a $5cm \times 5cm$ regions. To avoid occlusions caused by ceilings, we filter out Gaussians located more than 1 meter above the agent’s height before rendering. The resulting BEV feature map is fed into the waypoint predictor composed of two U-Nets: the first U-Net completes the BEV feature map, while the second predicts a heatmap representing potential waypoints. Using Non-Maximum Suppression (NMS), discrete waypoints relative to the agent are extracted from the heatmap and converted to world coordinates via the agent’s pose. These coordinates are then used to update the topological map from \mathcal{M}_{t-1} to \mathcal{M}_t shown in Figure 2.

The waypoint predictor is trained independently using a heatmap loss [54], with the same dataset as previous works [21, 50]. Additionally, an inpainting loss is applied to ensure the completeness of the BEV feature map. For more implementation details, please refer to the Appendix.

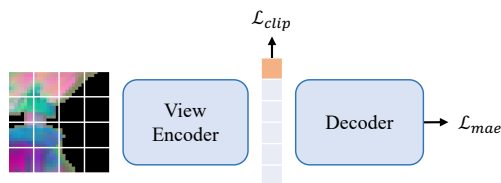


Figure 4. Training MAE on incomplete rendered feature map. The rendered incomplete feature map is randomly masked and input to the view encoder. Then the output is processed by the following decoder to reconstruct the full feature map.

3.5. Implicit Partial Completion Module

The feature fields rendered using 3DGS are often incomplete, particularly in unobserved areas (e.g., regions behind the agent), as demonstrated in Figure 4. For instance, unobserved structures such as stairs are rendered as black areas. To address this limitation, we propose an implicit partial completion module that leverages contextual relationships to infer representations of missing regions and extract comprehensive features from these partially incomplete views.

To infer features for missing regions, we leverage the MAE framework[18], which operates on partial features. The MAE view encoder takes rendered partial feature maps as input and generates corresponding latent representations. These latent features are then decoded to reconstruct complete feature maps, guided by the auxiliary loss inspired by MAE. Through this reconstruction-based training paradigm, the decoder learns to predict full feature maps from partial inputs, enabling the view encoder to implicitly infer missing visual information.

Formally, the 3DGS rendered partial feature maps denoted as $\{f_{t,i}^m\}_{i=1}^{12}$ that corresponding to 12 orientations, with a 30-degree angle between adjacent directions. Given the rendered i_{th} feature map $f_{t,i}^m$, we convert it to tokens like ViT[12]. These tokens undergo random masking(set to zero) and, along with an additional [CLS] token, are processed by a transformer-based view encoder. The encoded

tokens are subsequently decoded by another transformer to reconstruct the complete feature map. We train the completion encoder and decoder using the following reconstruction loss:

$$\mathcal{L}_{mae} = \frac{1}{|\mathcal{S}|} \sum_{x \in \mathcal{S}} \|x - x^{gt}\|^2, \quad (1)$$

where \mathcal{S} denotes the set of predicted masked tokens, x represents reconstructed tokens and x^{gt} corresponds to the ground truth tokens. We follow [18] to exclude non-masked patches from the loss computation. Notably, the reconstruction process covers both randomly masked patches and rendered black areas. Upon training completion, the [CLS] token is used to serve as the global representation f_i^v of the input feature map. For the rendered 12 views, we extract this global feature for each of them and obtain $\{f_{t,i}^v\}_{i=1}^{12}$. To maintain consistency with standard panoramic VLN training protocols that typically employ CLIP features, we also align $f_{t,i}^v$ to CLIP feature by maximizing feature similarity and minimizing distance:

$$\mathcal{L}_{clip} = 1 - \frac{f^v \cdot f^{clip}}{\|f^v\| \|f^{clip}\|} + \|f^v - f^{clip}\|_2^2 + \|f^v - f^{clip}\|_1, \quad (2)$$

f^{clip} is the corresponding CLIP feature. We omit script t, i here for clarity.

3.6. Mapping and Planning

With rendered panoramic features $\{f_{t,i}^v\}_{i=1}^{12}$ and predicted waypoints at each step, we construct a topological map and perform decision planning, following the general VLN models [3, 11, 21, 34] (see Section 3.1). Specifically, topological map \mathcal{M}_{t-1} is updated by these predicted features and waypoints to produce \mathcal{M}_t . Subsequently, the planner predicts an node a_t guided by the instruction \mathcal{I} :

$$\begin{aligned} p_t &= \text{Planner}(\mathcal{I}, \mathcal{M}_t), \\ a_t &= \arg \max p_t. \end{aligned} \quad (3)$$

Here, p_t is predicted probabilities and a_t corresponds to a node on \mathcal{M}_t and serves as a sub-goal for the navigation. The planner is trained using the ground-truth next node a_t^* by minimizing the cross-entropy loss at each step. The overall loss for the planner is defined as follows:

$$\mathcal{L}_{planner} = \frac{1}{T} \sum_{t=1}^T \text{CE}(p_t, a_t^*). \quad (4)$$

T is the total planning step and CE is cross entropy loss.

3.7. Uncertainty-aware Active Perception

Predictions made by the planner may be compromised, as the implicit partial completion module cannot retrieve information from entirely unobserved directions. To alleviate

this, we draw inspiration from human navigation and propose leveraging the agent’s subjective perspective through a simple yet effective active perception strategy. This strategy enables the agent to actively acquire visual observations when uncertain about its decisions. We focus on two key challenges: determining *when to look*, which involves identifying the steps for the agent to gather observations, and *where to look*, which involves selecting a informative directions for more visual observations.

(1) When to look? The agent should gather more information when it finds its prediction unreliable, as additional information have potential to improve prediction accuracy. We assess the reliability of a prediction by considering two conditions: *directional novelty* and *low confidence*. When both conditions are satisfied, the agent decides to gather more visual information. Directional novelty is defined as a direction pointing towards a previously unobserved sector, specifically any orientation outside the $\pm 30^\circ$ range of already viewed directions at current node. Low confidence is determined by thresholding the class-normalized entropy: $H_{norm} = H(p) / \log N > \tau$, where $H(p)$ is the entropy of prediction p_t and N is the number of candidates. Threshold τ is a hyper-parameter. This normalization enables stable thresholding across varying class numbers.

(2) Where to Look? When deciding to acquire additional visual information, we choose to have the agent look toward the orientation of the predicted node. Ideally, the optimal direction would be that of the ground truth(GT) next node. However, during inference, we cannot obtain this GT direction. Our strategy is to utilize the direction of the model-predicted node instead. This approach remains informative as the prediction model has been trained under the supervision of ground truth data.

Our solution to these questions culminates in the simple Uncertainty-aware Active Perception(UAP) strategy: the agent rotates to *where* the model-predicted node is *when* it is uncertain about its prediction. This inserts a checking step in the traditional combined rotate-then-move action, as depicted in Fig.3. The checking step enables the agent to determine whether to move forward or plan again(*i.e.* re-plan) with acquired visual information in the rotation. Re-plan may occur several times until the agent makes a certain decision or every view is observed.

4. Experiments

4.1. Setups

We evaluate our approach on the R2R-CE[27] and RxR-CE[29] datasets in simulated environments.

R2R-CE is a dataset based on MP3D[7] using Habitat simulator[42]. It includes 5,611 trajectories divided into train, validation seen, validation unseen, and test unseen splits. Each trajectory has 3 English instructions, with an

average length of 9.89 meters and average instruction length of 32 words. The camera’s FOV is 90 degrees.

RxR-CE is a larger multilingual VLN dataset containing 126K instructions in English, Hindi, and Telugu. It includes diverse trajectories in terms of length (the average is 15 meters), which is more challenging in continuous environments. The camera’s FOV for RxR-CE is 63 degrees.

Evaluation Metrics. We adopt standard metrics for evaluating the agent’s performance in VLN, including Navigation Error(NE), Success Rate(SR), SR given the Oracle stop policy(OSR), Success rate weighted by normalized inverse Path Length(SPL[4]), the normalized DTW distance between the agent’s trajectory and ground truth path, nDTW weighted by success rate(sDTW[23]).

Training of each module. The proposed BEV-based waypoint predictor (in Section 3.4) and implicit partial completion module (in Section 3.5) are trained independently before the overall navigation framework pipeline. Specifically, for training the BEV-based waypoint predictor, we randomly navigate between nodes in the graph, collecting rendered BEV features and their corresponding ground truth(GT) waypoint labels, following the methodology of prior works[21, 50], to train the predictor. We train the implicit partial completion module using the samples collected from the MP3D Simulator[7, 42]. To prevent information leakage, we exclude data from the MP3D unseen and test splits. We also include HM3D dataset[40] data, following prior works [49, 50], to enhance generalization.

Implementation Details. All models are trained with AdamW[33] optimizer. Scale of Gaussians is 2.5cm, with opacity 1 and no rotation. MAE mask ratio is set to 0.5. For the R2R dataset, the planning model is trained on two NVIDIA 3090 GPUs for 10^4 iterations, with batch size 4 per GPU, requiring approximately 2 days. The learning rate is initialized at 5×10^{-6} and follows a cosine decay schedule. During training, the planning model is initialized with weights pre-trained on panoramic setting. For the RxR dataset, the planning model is similarly trained for 2×10^4 iterations with learning rate 5×10^{-6} on 4 NVIDIA 3090 GPUs with batch size 2 per GPU, taking about 6 days.

4.2. Ablation Study

To evaluate the effectiveness of each module, we conduct ablation studies. Row 1 represents the baseline model, which is our proposed 3DGS-based navigation framework excluding the Implicit Partial Completion(IPC) module and the Uncertainty-Aware Active Perception(UAP) strategy. Row 2 incorporates the UAP strategy, while Row 3 adds the IPC module. Row 4 corresponds to the full model.

Comparing Row 1 with Row 2, the 1.7% success rate improvement confirms the effectiveness of our active perception strategy. Further analysis of Row 1 versus Row 3

| IPC | UAP | NE↓ | OSR↑ | SR↑ | SPL↑ |
|-----|-----|------|------|------|------|
| | | 4.83 | 59.1 | 51.1 | 41.7 |
| | ✓ | 4.87 | 59.8 | 52.8 | 43.2 |
| ✓ | | 4.70 | 60.7 | 54.0 | 44.4 |
| ✓ | ✓ | 4.61 | 62.4 | 54.8 | 44.4 |

Table 1. Ablation study.

reveals the critical role of the IPC module: its 2.9% performance gain demonstrates it extracts comprehensive representation from partially rendered feature maps. When integrating both UAP and IPC (Row 4), the full model achieves optimal performance across key metrics, notably +3.7% success rate. This synergy arises because the two components address information incompleteness in complementary ways: the UAP strategy proactively reduces significant incompleteness through active perception, while the IPC module implicitly enhances understanding of incomplete observations via implicit completion.

Discussion. Our framework employs a different backbone (MobileSAM), 3DGS-based feature field, and BEV-based waypoint predictor, distinguishing it from 3DFF, which uses CLIP, NeRF-based feature field, and semantic-map-based waypoint predictor. Even without IPC and UAP modules, our baseline outperforms it. These may be due to better feature field and BEV-based waypoint predictor that don’t rely on high-accuracy semantic segmentation. However, our main focus is on solving information incompleteness, so we skip detailed analysis of these components.

4.3. Comparison With Other VLN Methods

| Methods | Obs | Val Unseen | | | | Test Unseen | | | |
|----------------------|-----|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| | | NE↓ | OSR↑ | SR↑ | SPL↑ | NE↓ | OSR↑ | SR↑ | SPL↑ |
| ETPNav[3] | P | 4.71 | 65 | 57 | 49 | 5.12 | 63 | 55 | 48 |
| BEVBert[2] | P | 4.57 | 67 | 59 | 50 | 4.70 | 67 | 59 | 50 |
| PRET[34] | P | 4.58 | 66 | 59 | 49 | 4.74 | 62 | 56 | 47 |
| CM ² [15] | M | 7.02 | 41.5 | 34.3 | 27.6 | 7.74 | 39 | 32 | 24 |
| WS-MGMap[9] | M | 6.28 | 47.6 | 38.9 | 34.3 | 7.11 | 45 | 35 | 28 |
| NaVid[52] | M | 5.47 | 49.1 | 37.4 | 35.9 | - | - | - | - |
| 3DFF[50] | M | 5.95 | 55.8 | 44.9 | 30.4 | 6.24 | 54.4 | 43.7 | 28.9 |
| g3D-LF[46] | M | 5.70 | 59.5 | 47.2 | 34.6 | 6.00 | 57.5 | 46.3 | 32.2 |
| Ours | M | 4.61 | 62.4 | 54.8 | 44.4 | 4.97 | 60.5 | 53.6 | 44.9 |

Table 2. Evaluation on R2R dataset. Obs indicates the observation type. P means panoramic observation and M indicate monocular observation.

Comparison on R2R-CE. Table 2 compares our approach with previous VLN methods. Our model, built upon the PRET panoramic planning model, outperforms all other monocular approaches. Specifically, our method achieves a success rate approximately 10% higher than the 3DFF model. We also significantly outperform g3D-LF, suggesting that improving the feature field alone is insufficient for monocular observations, as information incompleteness remains a more critical issue. Importantly, our approach significantly narrows the performance gap between monocular

| Methods | Obs | Val Unseen | | | | |
|----------------------|-----|-------------|-------------|-------------|-------------|-------------|
| | | NE↓ | OSR↑ | SR↑ | SPL↑ | sDTW↑ |
| PRET[3] | P | 5.81 | 60.0 | 52.5 | 43.9 | 42.7 |
| ETPNav[3] | P | 5.64 | - | 54.8 | 44.8 | 45.3 |
| HNR[49] | P | 5.51 | - | 56.4 | 46.7 | 47.2 |
| CM ² [15] | M | 8.98 | 25.3 | 14.4 | 9.2 | - |
| WS-MGMap[9] | M | 9.83 | 29.8 | 15.0 | 12.1 | - |
| NaVid[52] | M | 8.41 | 34.5 | 23.8 | 12.2 | - |
| 3DFF[50] | M | 8.79 | 36.7 | 25.5 | 18.1 | - |
| Ours | M | 8.29 | 37.7 | 31.8 | 26.8 | 25.2 |

Table 3. Evaluation on RxR-CE dataset.

and panoramic models, reducing it from 12% to 4% in terms of success rate on the val unseen split.

Comparison on RxR-CE. Table 3 presents the performance comparison on the RxR-CE dataset. Our proposed monoVLN outperforms all previous monocular-based methods. Specifically, our method improves 6% on the success rate compared to previous monocular-based SOTA methods, indicating that the information incompleteness is also a critical challenge on RxR-CE dataset. Results on R2R-CE and RxR-CE demonstrate the effectiveness of our method on different benchmarks.

4.4. Design Choices of Each Module.

| | NE↓ | OSR↑ | SR↑ | SPL↑ |
|-----|------|------|------|------|
| (1) | 5.98 | 48.2 | 40.8 | 32.3 |
| (2) | 4.60 | 60.1 | 53.0 | 43.8 |
| (3) | 4.70 | 60.7 | 54.0 | 44.4 |

Table 4. Different ways to incorporate the reconstruction task. (1) Training with reconstruction loss (in Eq.1) only. (2) Training with joint CLIP aligning loss (in Eq.2) and reconstruction loss. (3) Pre-train with reconstruction then finetune with CLIP alignment.

How to incorporate the reconstruction auxiliary task in IPC module? The MAE view encoder is designed to achieve feature completion (Eq.1) and alignment with CLIP features (Eq.2). To this end, we explore three integration strategies for training: reconstruction-only, joint training, and pretrain-then-finetune. As demonstrated in Table 4, the exclusion of reconstruction during the training process results in substantially diminished performance metrics. The absence of CLIP alignment results in the lowest performance, primarily because the planning model is pre-trained solely with CLIP features. Through the integration of both reconstruction and clip alignment during the view encoder training phase, we observe a marked improvement in model performance. Moreover, pretraining with reconstruction loss followed by finetuning with CLIP alignment produces the best results, achieving a 1% improvement in success rate compared to joint training. Based on these empirical results, we adopt the pretrain-then-finetune paradigm in all other experiments.

Comprehensive study of our active perception strategy.

We evaluate the effectiveness of our Uncertainty-Aware Ac-

| Methods | initial rotation | NE↓ | OSR↑ | SR↑ | SPL↑ | num steps↓ |
|-------------|------------------|------|------|------|------|------------|
| w/o UAP | | 5.77 | 51.9 | 45.6 | 35.1 | 108 |
| ours | | 5.43 | 56.5 | 48.7 | 35.4 | 153 |
| rotate 360 | ✓ | 4.76 | 60.9 | 54.2 | 45.4 | 287 |
| random view | ✓ | 4.64 | 62.1 | 54.2 | 44.2 | 127 |
| w/o UAP | ✓ | 4.70 | 60.7 | 54.0 | 44.4 | 117 |
| ours | ✓ | 4.61 | 62.4 | 54.8 | 44.4 | 128 |

Table 5. Effectiveness of our Active Perception Strategy.

tive Perception(UAP) strategy, both with and without initial rotation, in Table 5. The initial rotation refers to the agent performing a 360-degree rotation before navigation, as implemented in 3DFF [50]. As shown, the first two rows (without initial rotation) demonstrate a significant improvement in success rate with our active perception strategy, achieving a 3.1% absolute gain. To ensure a fair comparison, we also test the setup with initial rotation¹. Rows 5 and 6 of the table reveal that the UAP strategy still enhances the success rate, albeit by a smaller margin of 0.8%. This reduced gain can be attributed to the diminished incompleteness in the rendered views when the initial rotation is performed. Under this experimental setup, we further explore the performance of the approach that involves a 360-degree rotation at every step in Row 3 and acquires a random view observation within the UAP strategy in Row 4. Notably, our UAP strategy outperforms the 360-degree rotation approach in terms of the success rate. The reason behind this is that panoramic views obtained through continuous 360-degree rotations may introduce redundant information. In contrast, our method is designed to selectively collect data only when uncertainty is detected, enabling it to focus on capturing the most relevant details. Moreover, our approach exhibits remarkable efficiency in terms of navigation steps. It significantly reduces the number of extra steps, from 170 in the 360-degree rotation approach to only 11, which further demonstrates the superiority of our UAP strategy.

Impact of entropy threshold in UAP. We investigate the optimal entropy threshold for the UAP strategy. As depicted in Figure 5, we train the model by utilizing a range of entropy thresholds, considering scenarios both with and without an initial rotation. A higher entropy threshold leads to a reduction in the frequency of replanning. This, in turn, results in a decrease in the number of navigation steps and model forward passes. When the threshold equals 1, no replanning occurs, equivalent to the baseline without UAP. In the scenario with an initial rotation, the optimal entropy threshold is 0.75, which is slightly larger than the optimal threshold of 0.5 in the scenario without an initial rotation. The underlying reason for this difference is that the 360-degree rotation at the onset of navigation provides a more comprehensive panoramic understanding.

¹This setup is adopted in other experiments for fair comparison.

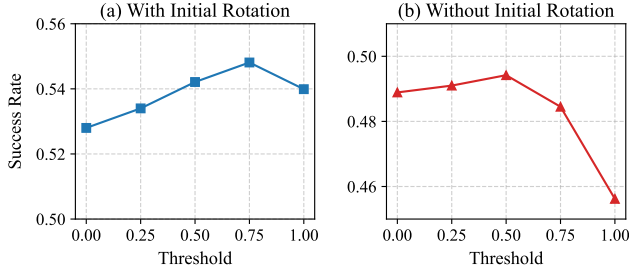


Figure 5. Success rate under different entropy threshold with or without initial rotation.

Discussion. Our experiments imply that monocular VLN can match panoramic performance with improved methods, and current panoramic methods can be further enhanced. As illustrated in Table 5, performing a full 360-degree rotation at each step does not yield the highest success rate. Moreover, according to Figure 5, the best performance is not attained when the maximum amount of visual information is collected at a threshold of 0. This suggests that current panoramic agents can be enhanced by eliminating redundant observations, thus revealing a promising direction for improving VLN agents.

4.5. Visualization

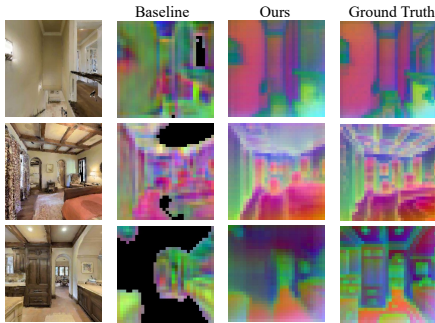


Figure 6. Visualization of completed feature maps in *unseen* environments. The feature map is visualized by PCA like DINOv2[37]. Baseline is the rendered feature map whereas ours is feature map completed by MAE decoder.

We visualize the rendered feature maps in *unseen* environments in Figure 6. As shown, compared with baseline, our pretrained module can reconstruct small background regions (e.g., row 2, the ceiling) and completes large black areas such as stairs and floors, even though these areas were not observed. Meanwhile, our approach not only reconstructs missing feature maps but also refines those that are blurry or noisy. For instance, in row 1, the rendered feature maps exhibit significant noise, whereas the refined versions are cleaner and more closely aligned with the ground truth, thereby reducing the reliance on high-quality feature fields.

Instruction: Walk out the room and turn right, move forward a bit and turn left, walk towards the corner and turn right, stop there.



Figure 7. Real world Example.

| GPU | backbone | waypoint | view | planner | total |
|------|----------|----------|-------|---------|-------|
| 4060 | 89ms | 140ms | 146ms | 8ms | 383ms |
| 3090 | 31ms | 27ms | 99ms | 10ms | 167ms |

Table 6. Inference time on different GPUs. Backbone is MobileSAM’s forward time. Waypoint includes BEV rendering and prediction time. View covers panoramic rendering and encoding time. Planner is VLN model forward time.

4.6. Properties of Our Method.

Fast Inference. Our model is efficient and achieves fast inference. We show the inference time of each module in Table 6. On 3090, our model’s inference is significantly faster than g3D-LF[46], which is running at more than 330ms on 4090. On 4060, the total inference time is 383ms, comparable to g3D-LF on 4090. The planner inference time on 4060 is smaller than 3090, this accounts for 4060’s higher clock speed, making it faster on small model inference.

Deployable on Real-world Robot. We deploy our model on Turtlebot4. The inference is done on a laptop with RTX 4060 GPU. As depicted in Figure 7, with given instruction, our model can navigate following it with monocular observation. We find a dataset bias that substantially degrades performance upon deployment. We provide more details in supplementary materials.

5. Conclusions

We propose a novel 3DGS-based vision and language navigation framework with monocular observation. We tackle the information incompleteness challenge by implicitly completing partial feature maps. We also design a simple uncertainty-aware active perception strategy to gather more visual information to mitigate the challenge. Our experiments imply monocular VLN can match panoramic performance with improved methods.

Limitations and future work. The generalization of our proposed monoVLN to real-world (e.g., our lab) deployment is limited due to the domain gap between the training data from the simulator and real-world scenarios. This limitation will be addressed in our future work.

Acknowledgments

This work was supported partially by NSFC (No. 62206315, 92470202, U21A20471), Guangdong NSF Project (No. 2024A1515010101, No. 2023B1515040025), Guangdong Key Research and Development Program(No.2024B0101040004), Guangzhou Basic and Applied Basic Research Scheme (No. 2024A04J4067).

References

- [1] Dong An, Yuankai Qi, Yan Huang, Qi Wu, Liang Wang, and Tieniu Tan. Neighbor-view enhanced model for vision and language navigation. In *ACMMM*, 2021. 2
- [2] Dong An, Yuankai Qi, Yangguang Li, Yan Huang, Liang Wang, Tieniu Tan, and Jing Shao. Bevbort: Multimodal map pre-training for language-guided navigation. In *ICCV*, 2023. 2, 6
- [3] 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. *PAMI*, 2024. 1, 2, 3, 4, 5, 6, 7
- [4] Peter Anderson, Angel X. Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir R. Zamir. On evaluation of embodied navigation agents. *CoRR*, 2018. 6
- [5] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian D. Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, 2018. 1, 2
- [6] Peter Anderson, Ayush Shrivastava, Joanne Truong, Arjun Majumdar, Devi Parikh, Dhruv Batra, and Stefan Lee. Sim-to-real transfer for vision-and-language navigation. In *CoRL*, 2020. 2
- [7] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D data in indoor environments. *3DV*, 2017. 5, 6
- [8] Kevin Chen, Junshen K. Chen, Jo Chuang, Marynel Vázquez, and Silvio Savarese. Topological planning with transformers for vision-and-language navigation. In *CVPR*, 2021. 2
- [9] Peihao Chen, Dongyu Ji, Kunyang Lin, Runhao Zeng, Thomas H. Li, Mingkui Tan, and Chuang Gan. Weakly-supervised multi-granularity map learning for vision-and-language navigation. In *NeurIPS*, 2022. 6, 7
- [10] Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. History aware multimodal transformer for vision-and-language navigation. *NeurIPS*, 2021. 1, 2
- [11] Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Think global, act local: Dual-scale graph transformer for vision-and-language navigation. In *CVPR*, 2022. 1, 2, 3, 4, 5
- [12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 4
- [13] Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. Speaker-follower models for vision-and-language navigation. In *NeurIPS*, 2018. 1, 2
- [14] Chen Gao, Xingyu Peng, Mi Yan, He Wang, Lirong Yang, Haibing Ren, Hongsheng Li, and Si Liu. Adaptive zone-aware hierarchical planner for vision-language navigation. In *CVPR*, 2023. 2, 4
- [15] Georgios Georgakis, Karl Schmeckpeper, Karan Wanchoo, Soham Dan, Eleni Miltsakaki, Dan Roth, and Kostas Daniilidis. Cross-modal map learning for vision and language navigation. In *CVPR*, 2022. 2, 6, 7
- [16] Pierre-Louis Guhur, Makarand Tapaswi, Shizhe Chen, Ivan Laptev, and Cordelia Schmid. Airbert: In-domain pretraining for vision-and-language navigation. In *ICCV*, 2021. 2
- [17] Weituo Hao, Chunyuan Li, Xiujuan Li, Lawrence Carin, and Jianfeng Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *CVPR*, 2020. 1, 2
- [18] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. 2, 4, 5
- [19] Keji He, Chenyang Si, Zhihe Lu, Yan Huang, Liang Wang, and Xinchao Wang. Frequency-enhanced data augmentation for vision-and-language navigation. In *NeurIPS*, 2023. 2
- [20] Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. Vln bert: A recurrent vision-and-language bert for navigation. In *CVPR*, 2021. 1, 2
- [21] Yicong Hong, Zun Wang, Qi Wu, and Stephen Gould. Bridging the gap between learning in discrete and continuous environments for vision-and-language navigation. In *CVPR*, 2022. 1, 2, 3, 4, 5, 6
- [22] Chenguang Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. Visual language maps for robot navigation. In *ICRA*, 2023. 2
- [23] Gabriel Ilharco, Vihan Jain, Alexander Ku, Eugene Ie, and Jason Baldridge. General evaluation for instruction conditioned navigation using dynamic time warping. In *NeurIPS Workshop*, 2019. 6
- [24] Aishwarya Kamath, Peter Anderson, Su Wang, Jing Yu Koh, Alexander Ku, Austin Waters, Yinfei Yang, Jason Baldridge, and Zarana Parekh. A new path: Scaling vision-and-language navigation with synthetic instructions and imitation learning. In *CVPR*, 2023. 2
- [25] Liyiming Ke, Xiujuan Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha S. Srinivasa. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *CVPR*, 2019. 2
- [26] Jacob Krantz and Stefan Lee. Sim-2-sim transfer for vision-and-language navigation in continuous environments. In *ECCV*, 2022. 2

- [27] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *ECCV*, 2020. 1, 2, 5
- [28] Jacob Krantz, Aaron Gokaslan, Dhruv Batra, Stefan Lee, and Oleksandr Maksymets. Waypoint models for instruction-guided navigation in continuous environments. In *ICCV*, 2021. 2
- [29] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *EMNLP*, 2020. 2, 5
- [30] Jialu Li, Hao Tan, and Mohit Bansal. Envedit: Environment editing for vision-and-language navigation. In *CVPR*, 2022. 2
- [31] Chong Liu, Fengda Zhu, Xiaojun Chang, Xiaodan Liang, Zongyuan Ge, and Yi-Dong Shen. Vision-language navigation with random environmental mixup. In *ICCV*, 2021. 2
- [32] Rui Liu, Wenguan Wang, and Yi Yang. Volumetric environment representation for vision-language navigation. In *CVPR*, 2024. 2
- [33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 6
- [34] Renjie Lu, Jingke Meng, and Wei-Shi Zheng. PRET: planning with directed fidelity trajectory for vision and language navigation. In *ECCV*, 2024. 1, 2, 3, 4, 5, 6
- [35] Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. Self-monitoring navigation agent via auxiliary progress estimation. In *ICLR*, 2019. 2
- [36] Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. Improving vision-and-language navigation with image-text pairs from the web. In *ECCV*, 2020. 2
- [37] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael G. Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision. *CoRR*, 2023. 8
- [38] Yanyuan Qiao, Yuankai Qi, Yicong Hong, Zheng Yu, Peng Wang, and Qi Wu. HOP: history-and-order aware pre-training for vision-and-language navigation. *CoRR*, 2022. 2
- [39] Yanyuan Qiao, Yuankai Qi, Yicong Hong, Zheng Yu, Peng Wang, and Qi Wu. HOP+: history-enhanced and order-aware pre-training for vision-and-language navigation. *TPAMI*, 2023. 2
- [40] Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M. Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X. Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-matterport 3d dataset (HM3D): 1000 large-scale 3d environments for embodied AI. In *NeurIPS*, 2021. 6
- [41] Sonia Raychaudhuri, Saim Wani, Shivansh Patel, Unnat Jain, and Angel X. Chang. Language-aligned waypoint (LAW) supervision for vision-and-language navigation in continuous environments. In *EMNLP*, 2021. 2
- [42] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *ICCV*, 2019. 5, 6
- [43] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *NAACL*, 2019. 1, 2
- [44] Hanqing Wang, Wenguan Wang, Wei Liang, Caiming Xiong, and Jianbing Shen. Structured scene memory for vision-language navigation. In *CVPR*, 2021. 2
- [45] Su Wang, Ceslee Montgomery, Jordi Orbay, Vighnesh Birodkar, Aleksandra Faust, Izzeddin Gur, Natasha Jaques, Austin Waters, Jason Baldridge, and Peter Anderson. Less is more: Generating grounded navigation instructions from landmarks. In *CVPR*, 2022. 2
- [46] Zihan Wang and Gim Hee Lee. g3d-1f: Generalizable 3d-language feature fields for embodied tasks. *CoRR*, 2024. 1, 2, 6, 8
- [47] Zun Wang, Jialu Li, Yicong Hong, Yi Wang, Qi Wu, Mohit Bansal, Stephen Gould, Hao Tan, and Yu Qiao. Scaling data generation in vision-and-language navigation. In *ICCV*, 2023. 2
- [48] Zun Wang, Jialu Li, Yicong Hong, Songze Li, Kunchang Li, Shoubin Yu, Yi Wang, Yu Qiao, Yali Wang, Mohit Bansal, and Limin Wang. Bootstrapping language-guided navigation learning with self-refining data flywheel. *CoRR*, 2024. 2
- [49] Zihan Wang, Xiangyang Li, Jiahao Yang, Yeqi Liu, Junjie Hu, Ming Jiang, and Shuqiang Jiang. Lookahead exploration with neural radiance representation for continuous vision-language navigation. In *CVPR*, pages 13753–13762, 2024. 2, 6, 7
- [50] Zihan Wang, Xiangyang Li, Jiahao Yang, Yeqi Liu, and Shuqiang Jiang. Sim-to-real transfer via 3d feature fields for vision-and-language navigation. In *CoRL*, 2024. 1, 2, 4, 6, 7
- [51] Chaoning Zhang, Dongshen Han, Yu Qiao, Jung Uk Kim, Sung-Ho Bae, Seungkyu Lee, and Choong Seon Hong. Faster segment anything: Towards lightweight sam for mobile applications. *arXiv preprint arXiv:2306.14289*, 2023. 3
- [52] 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. *RSS*, 2024. 1, 2, 6, 7
- [53] Shijie Zhou, Haoran Chang, Sicheng Jiang, Zhiwen Fan, Zehao Zhu, Dejia Xu, Pradyumna Chari, Suya You, Zhangyang Wang, and Achuta Kadambi. Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields. In *CVPR*, 2024. 3
- [54] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *CoRR*, 2019. 4