ProtoVis-Nav: Prototypical Visual Imagination for Visual-Spatial Aligned UAV Navigation

Guangyuan Sui **Yunzhong Hou*** **Chi Harold Liu**
Beijing Institute of Technology
{3120255778,yunzhong.hou,chiliu}@bit.edu.cn

Abstract

Vision language navigation (VLN) is very challenging for Unmanned Aerial Vehicles (UAVs) since small errors in 3D motion prediction can lead to navigation errors and collisions. Unable to fathom execution outcomes in 3D environments, existing methods often struggle to predict accurate navigation waypoints from language instructions. To solve this, we introduce the Prototypical Visual Imagination framework, shortened as ProtoVis-Nav, which forces the model to imagine an outcome-associated local visual representation for the intended next UAV movement. Specifically, we predict feature prototypes and use their linear combination to estimate the visual feature of the target region. In this manner, the VLN model first imagines the local visual representation of the next intended location, and then uses predicted prototypes as context for generating the final action, promoting alignment between visual context and waypoint outputs. Extensive experiments demonstrate the effectiveness of the ProtoVis-Nav framework and show substantial improvements over existing baselines. Data and code are available at link.

1 Introduction

Vision-Language Navigation (VLN) is a challenging problem in embodied AI, enabling agents to navigate complex environments while following natural language instructions [1, 2]. Although early VLN research predominantly centered on indoor navigation for ground-based robots [3, 4, 5, 6], a growing body of recent work focuses on Unmanned Aerial Vehicles (UAVs) [7, 8, 9, 10, 11], where navigation involves a more challenging 3D action space and requires precise motion control and reliable 3D trajectory planning [12]. Despite the rapid progress of Vision-Language Models (VLMs) [13, 14, 15, 16], applying them to UAV navigation remains difficult as they struggle to navigate 3D environments accurately and safely [17, 11].

A major reason for these navigation errors is the inability to associate motion predictions with their potential outcomes [18]. In fact, most methods directly predict continuous waypoints from visual observations and language instructions [19]. Since their training objective penalizes only the waypoint error, the model can hardly anticipate the execution outcomes of its own actions [20], increasing the risk of generating trajectories that are potentially unsafe in the 3D environment.

To address this problem, an emerging trend in visual language navigation is to equip the agent with future-aware visual predictive capabilities, often framed as navigation world models [21]. Specifically, they usually predict the future observation via a specific video generation model [20, 21] or as outputs of the same VLN model [22, 23, 24]. UAV navigation requires accurate fine-grained motion control grounded in local visual evidence [25]. In fact, *precise* flight actions are often determined by small and localized visual regions (*e.g.*, a narrow gap to fly through). This raises a key question: rather than generating the global future scene, can we enforce future-aware grounding by strictly focusing on these localized visual regions most critical to the upcoming movement?

*Corresponding author: yunzhong.hou@bit.edu.cn.

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

In this work, we propose the Prototypical Visual Imagination framework for UAV navigation, dubbed as ProtoVis-Nav. The method is motivated by a simple idea: rather than modeling an entire future scene or a long-horizon latent rollout, the agent should focus on the exact regions most relevant to its precise movement by imagining an outcome-associated local visual representation. To this end, we project the next intended UAV location onto the current image plane, crop the surrounding local region, and use its visual feature as the supervision target. Compared with using the entire future frame or the future-frame center, this projected region provides a supervision target that more directly grounds 3D motion prediction in the current image space, while avoiding the intractable imagination of future observations. Plus, we introduce a prototype-based feature prediction method. Instead of using multi-layer perceptrons to directly estimate the target feature, this approach predicts a set of Prototypical Visual Imagination (PVI) tokens, which are then combined through a softmax-normalized mixing matrix, so that their linear combination matches the target-region feature. The predicted PVI tokens are less susceptible to overfitting and are then used as context for waypoint prediction, encouraging tighter alignment between visual context and continuous action outputs.

We experimentally verify the effectiveness of the proposed method on OpenUAV [12] and HUGE-Bench [26]. Results show that ProtoVis-Nav substantially improves navigation performance over strong baselines and leads to more reliable continuous UAV navigation.

2 Related Works

Vision language navigation for UAVs. Vision-Language Navigation (VLN) has traditionally focused on guiding ground agents through structured, near-planar indoor environments [3, 4, 5, 6, 27, 28, 29]. Recently, the field has naturally extended to Unmanned Aerial Vehicles (UAVs). UAV-based VLN requires agents to ground language instructions into complex, continuous flight dynamics, demanding a stronger understanding of 3D spatial reasoning. Several realistic and continuous UAV VLN benchmarks have recently emerged [9, 11, 12, 17], providing an essential foundation for open-world UAV VLN research. Building upon them, TravelUAV [12] directly regresses the next waypoint based on current observations and UAV kinematics. NavFoM [30] further explores a cross-embodiment and cross-task navigation foundation model trained on large-scale navigation data, showing strong generalization across different navigation tasks and robotic platforms, including UAV navigation. To better process historical context, LongFly [31] introduces a history-aware spatiotemporal modeling strategy that transforms fragmented and redundant historical data into structured, compact, and expressive representations. To improve policy learning, OpenVLN [32] formulates a reinforcement learning framework to optimize VLMs for UAV navigation tasks. It efficiently fine-tunes the VLM using rule-based policies under limited training data, and incorporates a long-horizon trajectory planner that generates precise UAV actions via value-based rewards.

Despite these advancements, most continuous UAV VLN methods still mainly rely on current observations, historical context, or geometric priors for waypoint prediction. Since their training objective focuses on waypoint error, the model receives limited guidance for identifying visual cues associated with the next intended location of the UAV, increasing the risk of generating trajectories that are potentially unsafe in the 3D environment.

Navigation world models. An emerging trend in visual-language navigation is to equip agents with future-aware visual predictive capabilities, often framed as navigation world models. For example, Pathdreamer studies visual world modeling for indoor navigation, generating future 360° RGB, depth, and semantic observations along hypothetical trajectories with a two-stage structure-to-image generator [20]. Dreamwalker further adapts this idea to VLN-CE by constructing an explicit environment graph and using a scene synthesizer to imagine future panoramic observations, which are then evaluated through mental planning with Monte Carlo Tree Search [33]. HNR-VLN predicts multi-level CLIP-based semantic features for candidate future viewpoints via a hierarchical neural radiance representation, enabling more efficient path evaluation [34]. NWM scales this paradigm to video generation, where a diffusion transformer simulates future visual trajectories conditioned on actions and ranks or plans routes according to goal similarity [21].

More recent methods emphasize compact and adaptive latent prediction: UNeMo predicts the next visual state from visual features, language instructions, and candidate actions, feeding the predicted state back to refine navigation decisions [23]. MonoDream uses latent panoramic dreaming to predict current and future panoramic RGB-D features from a forward-facing RGB view, thereby

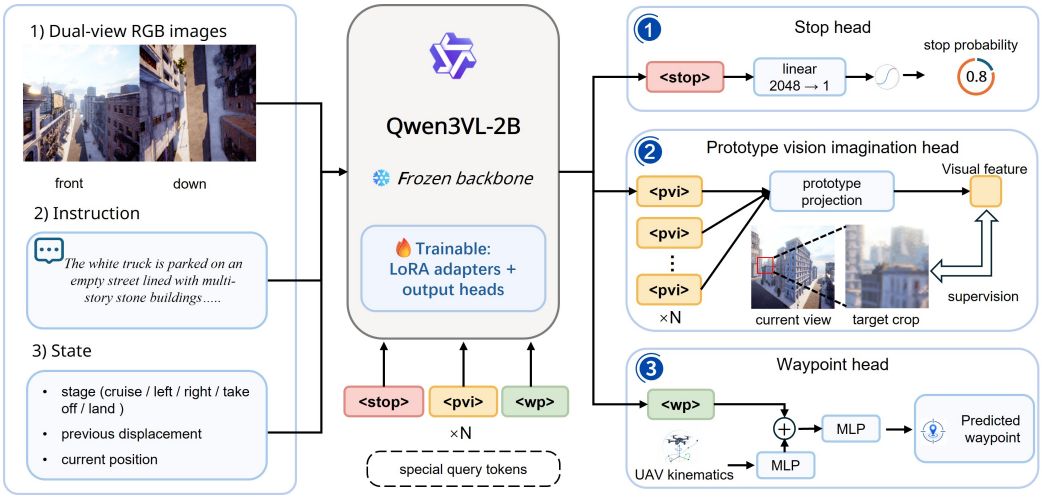


Figure 1: Overall pipeline of ProtoVis-Nav.

compensating for limited field of view without explicit panoramic sensors [24]. DriveVLA-W0 uses future-image prediction to supervise driving representations [35], while ANWM predicts action-conditioned future observations for UAV trajectory evaluation [36].

While effective for future-aware visual world models, they focus on generating global scene representations, such as full-frame videos or panoramic latents. However, UAVs require accurate fine-grained motion control, grounded in local visual evidence. Predicting the background and surroundings can provide general orientation, but *precise* flight actions are often determined by small and localized visual regions (*e.g.*, a narrow gap to fly through). To bridge this gap, our proposed ProtoVis-Nav framework departs from the global imagination paradigm. Instead of predicting complete future frames or global features, we explicitly project the intended future 3D location onto the current 2D image plane and predict prototypical visual tokens strictly for that local region. This localized, prototype-based imagination encourages the UAV to ground its motion decisions in the most critical visual context for more accurate and reliable continuous aerial navigation in 3D space.

Regarding local visual supervision, our method differs from ReconVLA [37], which reconstructs gaze regions around the objects to be manipulated in the current image with a diffusion model to guide visual attention. ProtoVis-Nav constructs its local target by projecting the next intended 3D UAV location onto the current 2D image plane, linking local visual supervision to the intended execution outcome of the next action.

3 ProtoVis-Nav

3.1 Overall Pipeline

As shown in Fig. 1, ProtoVis-Nav uses Qwen3VL-2B-Instruct [13] as the VLM backbone. The input consists of dual-view RGB observations (front-view and down-view images), the language instruction, and the current UAV state, including the previous displacement and current position. The Qwen3VL backbone is kept frozen, while the LoRA adapters inserted into the backbone and the newly added output heads are trained.

We adopt three types of special query tokens to the model input: `<stop>`, `<pvi>`, and `<wp>`. The `<stop>` token predicts whether the UAV should terminate the current navigation episode. The `<pvi>` tokens are short for Prototypical Visual Imagination (PVI), which produce visual prototypes that capture the local visual features around the next intended UAV location. These PVI tokens are then used as context for waypoint prediction. The hidden state of the `<wp>` token is used by the waypoint head to predict a waypoint for short-horizon trajectory generation.

Specifically, the hidden state of `<stop>` is passed through a linear layer and a sigmoid activation to produce a stop probability p_{stop}. When p_{stop} exceeds a predefined threshold τ_{stop}, the model issues a stop signal. Details of stop-head supervision are provided in Appendix A.1.

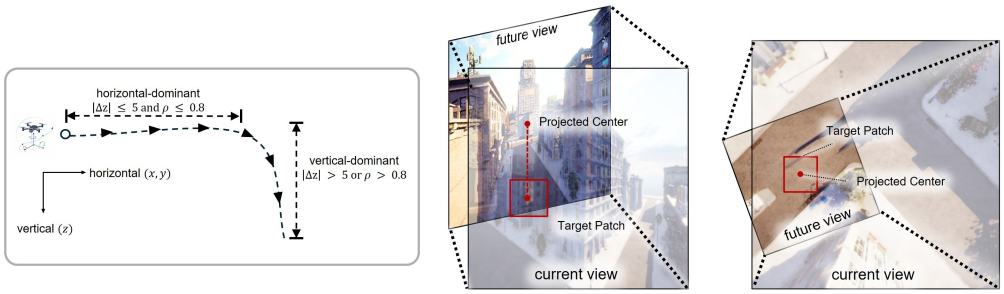


Figure 2: Illustration of target-region visual feature selection during training. The left panel shows our motion-dominance analysis for selecting the front or down view. Given the selected view, the next intended UAV location is projected onto the current image plane, and a local target patch is cropped around the projected center to supervise prototypical visual imagination.

For visual imagination, the K <pvi> tokens predict the feature prototypes, which are combined through a softmax-normalized linear weighting to produce an outcome-associated local visual representation of the target local region. This encourages the model to anticipate what the relevant visual context should look like before predicting the next action. See Section 3.3.

Finally, the <wp> token considers the <pvi> tokens as context within the VLM, and the waypoint head combines its hidden state with UAV kinematic information to predict the next waypoint. Details of the waypoint head are in the Appendix A.2.

3.2 Visual Imagination Target

During training, for the current frame t and the future frame $t+1$, we obtain their UAV positions $\boldsymbol{p}_t, \boldsymbol{p}_{t+1} \in \mathbb{R}^3$ and orientations $R_t, R_{t+1} \in \mathbb{R}^{3\times3}$ from the recorded expert trajectory. We also record the frontal and downward duo-views as $\boldsymbol{x}_t^{\text{front}} \in \mathbb{R}^{3\times256\times256}$ and $\boldsymbol{x}_t^{\text{down}} \in \mathbb{R}^{3\times256\times256}$, both of which are of a 256×256 spatial resolution.

For visual imagination supervision, the target-region is a local visual patch in the current observation $\boldsymbol{x}_t \in \left\{\boldsymbol{x}_t^{\text{front}}, \boldsymbol{x}_t^{\text{down}}\right\}$, which provides an outcome-associated local visual representation of the next intended UAV location. The entire future frame mostly describes background regions that are not directly related to the precise 3D motions, and predicting the future frame would introduce an intractable prediction. Thus, we do not use the future observation $\boldsymbol{x}_{t+1} \in \left\{\boldsymbol{x}_{t+1}^{\text{front}}, \boldsymbol{x}_{t+1}^{\text{down}}\right\}$, either as an entire frame or using its center, for the imagination target. Instead, we only focus on a small patch from the current observation that corresponds to the future UAV location to visually ground the 3D waypoint. In this design, the VLM essentially only needs to point out the future location from the current observation, offering a robust supervision target.

To locate this region from the current observation, we use the next drone location $\boldsymbol{p}_{t+1}$ and project this point onto the current camera image plane,

$$\boldsymbol{p}_{\text{proj}} = \boldsymbol{R}_t^\top (\boldsymbol{p}_{t+1} - \boldsymbol{p}_t), \tag{1}$$

where $\boldsymbol{p}_{\text{proj}}$ denotes the projected location in the current UAV reference frame. We then crop a 32×32 patch around the projected coordinate,

$$\tilde{\boldsymbol{x}} = \texttt{patch}\left(\boldsymbol{x}_t, \boldsymbol{p}_{\text{proj}}\right), \tag{2}$$

where $\tilde{\boldsymbol{x}} \in \mathbb{R}^{3\times32\times32}$ denotes the 32×32 image patch from $\boldsymbol{x}_t$. The visual feature of this local patch is then used as the supervision target for prototypical visual imagination.

Before we can crop out the target region $\tilde{\boldsymbol{x}}$, one remaining issue is to determine which one of the duo-view observations to use. While both front and down views are fed into the VLM (see Fig. 1), only one of them best reflects the intended future location. We use a simple intuition to select the view: front view is more suitable for forward motion, while down view is more informative for take-off, landing, or other vertical-dominant movements. Specifically, we consider two cues.

First, we compute the vertical-to-horizontal movement ratio,

$$\rho = \frac{\|\boldsymbol{p}_{\text{proj}}^z\|_2}{\|\boldsymbol{p}_{\text{proj}}^{xy}\|_2}, \tag{3}$$

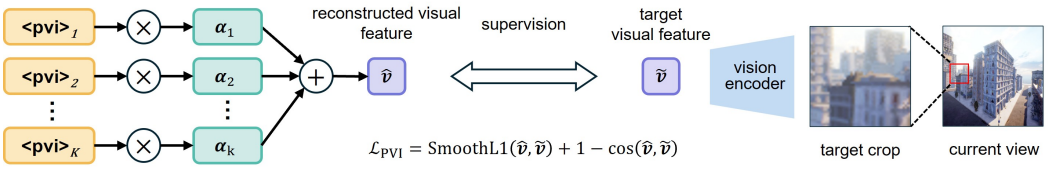


Figure 3: Illustration of the prototypical visual imagination head with K <pvi> tokens and one output visual token. The hidden states of multiple <pvi> tokens are combined by softmax-normalized weights into a predicted target-region visual feature, which is supervised by the target crop feature extracted from the visual encoder.

where $\boldsymbol{p}^{xy}_{\text{proj}} \in \mathbb{R}^2$ and $\boldsymbol{p}^{z}_{\text{proj}} \in \mathbb{R}$ denote the horizontal and vertical displacements.

Second, we check whether the UAV is clearly ascending or descending in a short temporal window around the current step. Specifically, we compare the altitude before the current step with the altitude a few steps ahead,

$$\Delta z = z_{t+2} - z_{t-1}. \tag{4}$$

A positive Δz indicates ascending motion, while a negative value indicates descending motion. We use its absolute value to cover both cases.

Overall, the motion is regarded as vertical-dominant if the short-window altitude change satisfies $|\Delta z| > 5$ or the body-frame vertical-to-horizontal ratio satisfies $\rho > 0.8$; otherwise, it is regarded as horizontal-dominant. For vertical-dominant motion, the target patch is extracted from the down view, while for horizontal-dominant motion, it is extracted from the front view:

$$\boldsymbol{x}_t = \begin{cases} \boldsymbol{x}_t^{\text{down}}, & \text{if } |\Delta z| > 5 \text{ or } \rho > 0.8, \\ \boldsymbol{x}_t^{\text{front}}, & \text{otherwise.} \end{cases} \tag{5}$$

where $\boldsymbol{x}_t$ determines the duo-view selection in Eq. 2. Fig. 2 illustrates the motion-based view selection and the target-region construction. The left panel shows how horizontal-dominant and vertical-dominant motions determine which view is used. After selecting the view, the next intended UAV location is projected onto the current image plane, and a local target patch is cropped around the projected center as the visual imagination target, as shown in the right panel.

3.3 Prototypical Visual Imagination Head

As shown in Fig. 3, the Prototypical Visual Imagination Head is a lightweight module that treats the <pvi> token outputs as visual feature prototypes [38]. This module learns a linear combination of these prototypes to match the target-region visual feature. This design uses the <pvi> token outputs as a set of bases for encoding the target-region visual cues, which works better at providing context for the <wp> prediction. We verify this empirically and show that our prototype design outperforms the MLP alternative, see Section 4.4.

We consider $K = 16$ <pvi> tokens $\boldsymbol{h}_k,\ k \in \{1, \ldots, K\}$. Using a learned softmax mixing matrix,

$$\boldsymbol{A} = \{a_{n,k}\}, \quad \alpha_{n,k} = \frac{\exp(a_{n,k})}{\sum_{j=1}^{K} \exp(a_{n,j})}, \tag{6}$$

where N is the number of target visual tokens for describing $\tilde{\boldsymbol{x}}$ in Eq. 2, $\alpha_{n,k}$ denotes the mixing weight from the k-th PVI token to the n-th visual token, $\boldsymbol{A} \in \mathbb{R}^{N \times K}$ is the overall mixing logits. The predicted visual tokens are obtained by

$$\hat{\boldsymbol{v}}_n = \sum_{k=1}^{K} \alpha_{n,k} \boldsymbol{h}_k, \quad n = \{1, \ldots, N\}. \tag{7}$$

For the 32×32 target crop, the Qwen3-VL visual encoder produces $N = 1$, so the head represents the target-region visual feature as a soft combination of 16 PVI tokens.

The supervision target is the feature extracted by the Qwen3VL visual encoder,

$$\{\tilde{\boldsymbol{v}}_n\} = \texttt{encoder}(\tilde{\boldsymbol{x}}), \quad n = \{1, \ldots, N\}. \tag{8}$$

The predicted features $\{\hat{\boldsymbol{v}}\}$ are supervised to match the target features $\{\tilde{\boldsymbol{v}}\}$ using this PVI loss,

$$\mathcal{L}_{\text{PVI}} = \frac{1}{N}\sum_{n=1}^{N}\left(\text{SmoothL1}(\hat{\boldsymbol{v}}_n, \tilde{\boldsymbol{v}}_n) + 1 - \cos(\hat{\boldsymbol{v}}_n, \tilde{\boldsymbol{v}}_n)\right). \tag{9}$$

Through this supervision, the <pvi> tokens are encouraged to encode the visual feature of the local region around the next intended UAV location, and their hidden states provide context for the waypoint token within the VLM.

4 Experiments

4.1 Experimental Setup

Benchmarks and platform. We conduct experiments on OpenUAV and the obstacle-aware region traversal tasks in HUGE-Bench. OpenUAV is built on the TRAVEL platform [12]. It provides diverse outdoor environments, realistic flight dynamics, and closed-loop simulation support for continuous UAV navigation. The platform supports physics-based UAV control with 6-DoF pose representation and configurable sensors such as RGB/depth cameras. The OpenUAV benchmark contains human-operated UAV trajectories paired with multi-view RGB-D observations, language instructions, UAV states, and waypoint annotations. OpenUAV evaluation uses the L1 assistant setting. HUGE-Bench combines 3D Gaussian Splatting with mesh geometry to support photorealistic rendering and collision-aware evaluation of UAV tasks [26].

Data split. We follow the official data split of OpenUAV. The training set contains 9,152 trajectories. For evaluation, we report results on the Test Seen and Test Unseen splits. The Test Seen split contains 1,410 trajectories from objects and scenes seen in the training set. The Test Unseen split contains 1,587 trajectories from maps or objects that do not appear in the training set. Following the official OpenUAV setting, test episodes are further grouped into Full, Easy, and Hard subsets according to trajectory length. Specifically, trajectories shorter than 250 meters are labeled as Easy, while trajectories at or above 250 meters are labeled as Hard. Unless otherwise specified, we report results on the Full subset for variant studies and report Full, Easy, and Hard results for state-of-the-art comparison. For HUGE-Bench, we use the obstacle-aware region traversal task subset [26], which requires traversing a designated area while avoiding obstacles. We train and evaluate models separately within each benchmark.

Evaluation metrics. We evaluate OpenUAV navigation performance using four metrics: Navigation Error (NE), Success Rate (SR), Oracle Success Rate (OSR), and Success weighted by Path Length (SPL) [12]. NE measures the distance between the UAV's final position and the target position, and is reported in meters; a lower NE indicates that the UAV stops closer to the target. SR measures the percentage of episodes in which the UAV successfully stops within a predefined distance threshold from the target, and is reported as a percentage. Following the official OpenUAV setting, an episode is regarded as successful if the UAV stops within 20 meters of the target. OSR measures whether the UAV reaches the target region at any time during navigation, regardless of whether it finally stops there, and is reported as a percentage. SPL jointly considers navigation success and path efficiency by weighting successful episodes according to the ratio between the shortest and the executed path length, and is also reported as a percentage. For NE, lower is better; for SR, OSR, and SPL, higher is better. For HUGE-Bench, higher normalized Dynamic Time Warping (nDTW) indicates closer alignment with the expert trajectory. Higher Collision-aware Success weighted by Path Length (CSPL) scores and a lower Collision Rate (CR) indicate better performance.

Baselines. We compare ProtoVis-Nav with representative UAV VLN baselines on OpenUAV, including Random Action, CMA [39], TravelUAV [12], NavFoM [30], and LongFly [31]. Random Action serves as a simple lower-bound policy. CMA is a classical cross-modal attention baseline. TravelUAV is the original VLM-based method on the OpenUAV benchmark. NavFoM is a navigation foundation model evaluated on OpenUAV, and LongFly is a recent spatiotemporal context modeling method for long-horizon UAV VLN. We also report Base (Ours), which uses the same VLM backbone, LoRA setting, stop head, and waypoint head as ProtoVis-Nav, but removes the <pvi> tokens and the PVI Head. On HUGE-Bench, we compare ProtoVis-Nav with Base.

Implementation details. ProtoVis-Nav adopts Qwen3VL-2B-Instruct [13] as the VLM backbone. During training, we freeze the backbone parameters and train only the LoRA adapters and the newly

Table 1: Comparison with state-of-the-art methods on the OpenUAV benchmark.

Test Seen

Method	Full				Easy				Hard			
	NE ↓	SR ↑	OSR ↑	SPL ↑	NE ↓	SR ↑	OSR ↑	SPL ↑	NE ↓	SR ↑	OSR ↑	SPL ↑
Random Action	222.20	0.14	0.21	0.07	142.07	0.26	0.39	0.13	320.12	0.00	0.00	0.00
CMA	135.73	8.37	18.72	7.90	84.89	11.48	24.52	10.68	197.77	4.57	11.65	4.51
TravelUAV	106.28	16.10	44.26	14.30	68.78	18.84	47.61	16.39	152.04	12.76	40.16	11.76
NavFoM	93.05	29.17	49.24	25.03	58.98	32.91	53.16	27.87	143.83	23.58	43.40	20.80
LongFly	60.02	36.39	65.87	31.07	38.10	38.52	**71.90**	31.24	85.20	33.94	58.94	30.88
Base	55.67	46.47	62.69	41.14	42.46	49.04	62.71	41.94	**71.96**	43.31	62.68	40.14
ProtoVis-Nav	**51.77**	**50.78**	**66.50**	**44.68**	**35.13**	**54.92**	68.84	**46.68**	72.28	**45.67**	**63.62**	**42.21**

Test Unseen

Method	Full				Easy				Hard			
	NE ↓	SR ↑	OSR ↑	SPL ↑	NE ↓	SR ↑	OSR ↑	SPL ↑	NE ↓	SR ↑	OSR ↑	SPL ↑
Random Action	225.64	0.06	0.06	0.06	164.66	0.19	0.19	0.19	280.58	0.00	0.00	0.00
CMA	147.27	4.98	12.41	4.74	102.54	8.03	17.52	7.52	191.30	2.76	7.53	2.71
TravelUAV	130.60	11.41	31.13	10.45	96.27	12.47	33.31	11.29	167.49	10.62	28.91	9.80
NavFoM	118.34	15.63	30.46	14.21	89.77	16.98	32.22	15.35	155.69	14.35	27.79	13.16
LongFly	91.84	24.19	**43.86**	20.84	**69.16**	22.89	**43.24**	18.66	112.02	25.36	44.41	22.76
Base	98.04	27.91	40.71	25.30	78.25	27.92	37.01	24.63	116.70	27.91	44.19	25.92
ProtoVis-Nav	**90.85**	**30.12**	41.52	**27.53**	75.43	**29.22**	38.44	**26.06**	**105.39**	**30.97**	**44.43**	**28.92**

added output heads. ProtoVis-Nav takes 256×256 front-view and down-view RGB images, the language instruction, and UAV state information as input on OpenUAV. HUGE-Bench uses front-view RGB images. The UAV pose and waypoint annotations are used during training to construct the projected target-region supervision for Prototypical Visual Imagination. By default, we use $K = 16$ `<pvi>` tokens and a 32×32 target crop. The Prototypical Visual Imagination Head uses a softmax-normalized mixing matrix to combine PVI token hidden states into a target-region visual representation for feature supervision. We train ProtoVis-Nav using AdamW with a learning rate of 5×10^{-5} and a global batch size of 64. During evaluation, the maximum number of waypoint steps is set to 200. Training is conducted on 4 NVIDIA RTX 5090 GPUs, and inference is performed on a single NVIDIA RTX A6000 GPU.

4.2 Comparison with State-of-the-Art Methods

We compare ProtoVis-Nav with representative UAV VLN methods on the OpenUAV benchmark and the obstacle-aware region traversal tasks of HUGE-Bench. As shown in Table 1, ProtoVis-Nav achieves competitive performance across both seen and unseen evaluation settings.

Results on OpenUAV Test Seen. On the Test Seen split, ProtoVis-Nav obtains strong results across the Full, Easy, and Hard subsets. For the Full subset, ProtoVis-Nav achieves an NE of 51.77m, an SR of 50.78%, an OSR of 66.50%, and an SPL of 44.68%, outperforming all compared methods on the four metrics. These results support our motivation that predicting a localized target-region visual feature can provide useful future-aware grounding for continuous waypoint prediction.

On the Easy subset, ProtoVis-Nav achieves an NE of 35.13m, an SR of 54.92%, an OSR of 68.84%, and an SPL of 46.68%. It obtains the best NE, SR, and SPL, while LongFly achieves the best OSR. On the Hard subset, ProtoVis-Nav obtains an NE of 72.28m, an SR of 45.67%, an OSR of 63.62%, and an SPL of 42.21%. It achieves the best SR, OSR, and SPL, while Base (Ours) has a slightly lower NE. This shows that ProtoVis-Nav improves success-oriented metrics on longer trajectories, although it does not uniformly improve final-position error in the Hard setting.

Compared with Base (Ours), ProtoVis-Nav brings consistent gains on the Full subset, reducing NE from 55.67m to 51.77m and improving SR from 46.47% to 50.78%. It also improves OSR from 62.69% to 66.50% and SPL from 41.14% to 44.68%. These results show that prototypical visual imagination offers a useful signal beyond direct waypoint supervision.

Results on OpenUAV Test Unseen. On the Test Unseen split, ProtoVis-Nav achieves competitive generalization performance. For the Full subset, ProtoVis-Nav obtains an NE of 90.85m, an SR of

Table 2: Comparison on HUGE-Bench obstacle-aware region traversal split.

Model	nDTW ↑	SR (%) ↑	CSPL ↑	CR (%) ↓
Base	0.1599	36.39	0.3179	54.46
ProtoVis-Nav	**0.1762**	**47.47**	**0.4161**	**39.76**
p-value	0.3242	**0.0458**	**0.0309**	**0.0269**

Table 3: Comparison of safety metrics for Base and ProtoVis-Nav on OpenUAV.

Model	CR (%) ↓	Near-collision rate (%) ↓			Clearance (m) ↑
		< 0.5 m	< 1.0 m	< 1.5 m	
Base	36.44	0.3804	0.5912	0.7938	16.7205
ProtoVis-Nav	**34.41**	**0.3746**	**0.5630**	**0.7555**	**16.8192**

30.12%, an OSR of 41.52%, and an SPL of 27.53%. It achieves the best NE, SR, and SPL among the compared methods, while LongFly obtains a slightly higher OSR. The drop from Test Seen to Test Unseen indicates that navigation in unseen maps or with unseen objects remains challenging.

On the Easy subset, ProtoVis-Nav achieves an NE of 75.43m, an SR of 29.22%, an OSR of 38.44%, and an SPL of 26.06%, obtaining the best SR and SPL. On the Hard subset, ProtoVis-Nav achieves an NE of 105.39m, an SR of 30.97%, an OSR of 44.43%, and an SPL of 28.92%, obtaining the best results on all four metrics.

Compared with Base (Ours), ProtoVis-Nav improves the Full-set SR from 27.91% to 30.12% and reduces NE from 98.04m to 90.85m. It also improves OSR from 40.71% to 41.52% and SPL from 25.30% to 27.53%. These results suggest that PVI supervision remains useful for unseen environments, while generalization in continuous UAV VLN is still difficult.

Results on HUGE-Bench Obstacle-Aware Region Traversal Tasks. Evaluation results on these tasks vary substantially across runs, so Table 2 reports the mean over five independent training-and-evaluation runs. Compared with Base, ProtoVis-Nav increases nDTW from 0.1599 to 0.1762, SR from 36.39% to 47.47%, and CSPL from 0.3179 to 0.4161, and reduces CR from 54.46% to 39.76%. These gains indicate that PVI supports goal reaching and collision-aware trajectory execution, as traversal requires waypoint choices that account for nearby obstacles.

4.3 Safety Analysis on OpenUAV

On OpenUAV, CR measures the percentage of episodes with a collision. Near-collision rate measures the percentage of frames whose obstacle clearance falls below a given threshold, and mean obstacle clearance measures the average distance to the nearest obstacle. We use clearance thresholds of 0.5m, 1.0m, and 1.5m to assess obstacle proximity at different distances.

As shown in Table 3, ProtoVis-Nav reduces CR from 36.44% to 34.41% and increases mean obstacle clearance from 16.7205m to 16.8192m. At these thresholds, near-collision rates decrease from 0.3804%, 0.5912%, and 0.7938% for Base to 0.3746%, 0.5630%, and 0.7555% for ProtoVis-Nav, respectively. These reductions indicate less frequent exposure to nearby obstacles during flight.

Table 4: Variant study on different visual imagination targets and PVI head design.

Variant	NE ↓	SR ↑	OSR ↑	SPL ↑
Front view only	62.94	50.07	61.99	44.15
Center crop	69.06	47.25	59.59	41.86
Future full frame	55.76	47.25	64.60	41.66
Future center crop	79.15	42.67	56.42	37.65
MLP	56.29	46.76	64.81	41.31
ProtoVis-Nav	**51.77**	**50.78**	**66.50**	**44.68**

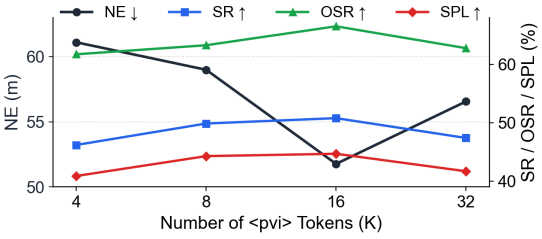


Figure 4: Effect of PVI token count.

4.4 Variant Study

We conduct variant studies on the Test Seen split to analyze three design choices of ProtoVis-Nav: the visual imagination target, the PVI head design, and the number of `<pvi>` tokens.

Visual imagination target. We first evaluate different choices of the visual imagination target. *Front view only* always extracts the target-region feature from the current front-view image, without adaptive selection between front and down views. *Center crop* uses the image center as the crop location, instead of projecting the next intended UAV location onto the current image plane. *Future full frame*

uses the visual feature of the full future observation, while *Future center crop* uses the center crop of the future observation. As shown in Table 4, ProtoVis-Nav achieves the best overall performance. Compared with *Center crop*, ProtoVis-Nav reduces NE from 69.06m to 51.77m, showing that the projected target region is more informative than a fixed image-center target. Compared with *Future full frame* and *Future center crop*, ProtoVis-Nav also performs better, suggesting that focusing on the localized region around the next intended UAV location offers a more direct supervision target for aligning visual context with waypoint prediction.

PVI head design. We then study the design of the visual imagination head. The *MLP* variant directly regresses the target-region feature with an MLP, while ProtoVis-Nav predicts feature prototypes and linearly combines them to estimate the target feature. As shown in Table 4, ProtoVis-Nav outperforms the MLP variant across all metrics. This suggests that the goal of visual imagination is not merely to fit the target feature with a high-capacity output head. Instead, it is more important to force the `<pvi>` tokens to encode visual cues that are useful for waypoint prediction. By forming the prediction as a linear combination of `<pvi>` tokens, the prototype-based design makes the imagination objective act more directly on the token representations used by the `<wp>` token.

Number of `<pvi>` tokens. We analyze the effect of the number of PVI tokens. The value of K controls the number of visual prototypes for representing the imagined target-region feature. We compare $K = \{4, 8, 16, 32\}$ to study how prototype capacity affects navigation performance. As shown in Fig. 4, the best overall performance is achieved at $K = 16$. When K is too small, the PVI tokens may not provide enough capacity to represent the target-region visual feature. When K becomes too large, the target-region information may be scattered across redundant PVI tokens, making the context less compact for the `<wp>` token and weakening its connection to waypoint prediction. Therefore, we use $K = 16$ as the default setting.

4.5 PVI Spatial Visualization and Waypoint Token Attention

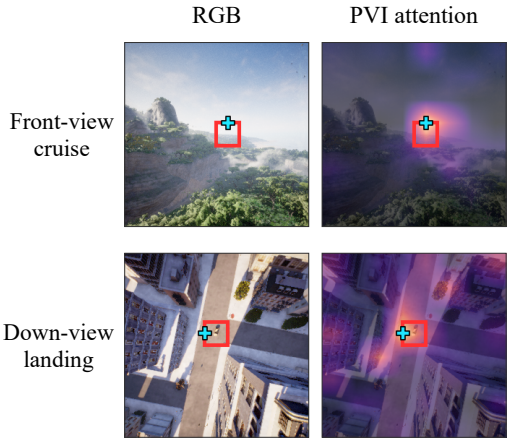


Figure 5: Visualization of PVI attention maps over the current images. Red boxes mark the ground-truth projected target regions, and cyan markers indicate the PVI attention peaks.

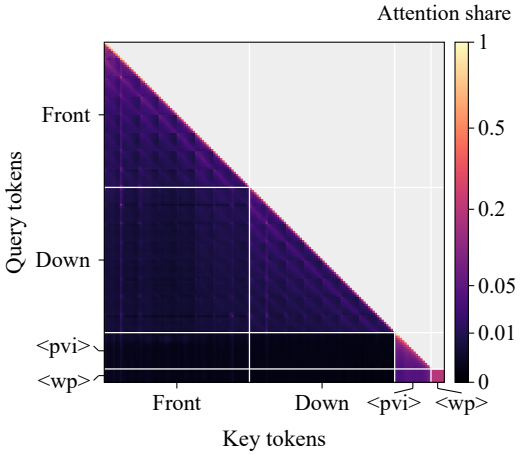


Figure 6: Mean causal attention over 1,000 frames. Text and UAV-state tokens are omitted; each row is renormalized over the remaining groups.

Spatial visualization. We visualize two examples by overlaying the PVI attention map on the current image (Fig. 5). The cruise and landing examples connect the intended next UAV movement to local visual regions in the front and down views, respectively. In both cases, the attention peak lies within or near the projected target region. This suggests that PVI focuses on visual regions relevant to the next intended UAV movement.

Waypoint token attention. We analyze the waypoint token's causal attention over 1,000 frames (Fig. 6). We average the attention matrices across frames to summarize interactions between image, PVI, and waypoint tokens. Its attention concentrates on the PVI tokens. The waypoint token can combine this movement-related visual context with the instruction and UAV state through causal

attention. Together, the visualizations and analysis suggest that PVI identifies locally action-relevant visual evidence and influences the waypoint token's representation through attention.

4.6 Qualitative Analysis



Figure 7: Qualitative example of ProtoVis-Nav on OpenUAV. The UAV follows the instruction and gradually approaches the target region. The green check mark indicates successful navigation.

We provide qualitative examples to illustrate how ProtoVis-Nav navigates toward the target. As shown in Fig. 7, the instruction describes a white truck parked on an empty street surrounded by multi-story stone buildings. In this example, the UAV gradually moves along the street, adjusts its viewpoint across consecutive observations, and finally stops near the target truck. The sequential observations show that ProtoVis-Nav maintains target-relevant visual context for waypoint prediction under changing viewpoints. This result is consistent with our motivation that prototypical visual features provide useful context for waypoint prediction.

4.7 Limitations and Social Impact

Despite the improvements, ProtoVis-Nav still faces challenges in complex aerial environments. Based on our observation, most failure cases are caused by collisions, especially with trees and thin obstacles such as branches, poles, and wires. These obstacles often occupy only a small image region and may be partially occluded or visually ambiguous. This makes obstacle boundaries and available clearance difficult to estimate from RGB observations. In future work, we will explore stronger obstacle-aware perception and planning mechanisms, including depth or LiDAR inputs, explicit collision-risk modeling, and anti-collision trajectory refinement for safer UAV navigation. Our closed-loop evaluation is conducted in simulation. Future work will extend validation to real-world UAV flights and hardware-in-the-loop experiments to assess closed-loop navigation performance under physical sensing and control conditions.

ProtoVis-Nav may benefit applications such as search-and-rescue, disaster response, and infrastructure inspection by improving UAV navigation from language instructions. At the same time, UAV navigation systems may raise safety, privacy, and security concerns if deployed without sufficient safeguards, and should therefore be used with human oversight and deployment constraints.

5 Conclusion

We propose ProtoVis-Nav, a Prototypical Visual Imagination framework for UAV vision-language navigation. The key idea is to ground waypoint prediction in local visual context associated with the next intended movement, which is especially important for UAVs where small 3D motion errors may lead to unsafe navigation behavior. ProtoVis-Nav imagines an outcome-associated local visual representation for the intended next UAV movement. Specifically, it predicts feature prototypes and linearly combines them to estimate the target-region visual feature, and the prototype token representations provide context for waypoint prediction. This design promotes better alignment between visual context and waypoint outputs. Extensive experiments on OpenUAV and HUGE-Bench demonstrate the effectiveness of ProtoVis-Nav.

Acknowledgments and Disclosure of Funding

This paper was sponsored by the National Natural Science Foundation of China (No. U25A20434).

References

[1] Jing Gu, Eliana Stefani, Qi Wu, Jesse Thomason, and Xin Wang. Vision-and-language navigation: A survey of tasks, methods, and future directions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7606–7623, 2022.

[2] Sang-Min Park and Young-Gab Kim. Visual language navigation: A survey and open challenges. *Artificial Intelligence Review*, 56(1):365–427, 2023.

[3] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision*, pages 104–120. Springer, 2020.

[4] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4392–4412, 2020.

[5] Gengze Zhou, Yicong Hong, and Qi Wu. Navgpt: Explicit reasoning in vision-and-language navigation with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 7641–7649, 2024.

[6] Chenguang Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. Visual language maps for robot navigation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10608–10615. IEEE, 2023.

[7] Hengxing Cai, Jinhan Dong, Jingjun Tan, Jingcheng Deng, Sihang Li, Zhifeng Gao, Haidong Wang, Zicheng Su, Agachai Sumalee, and Renxin Zhong. Flightgpt: Towards generalizable and interpretable uav vision-and-language navigation with vision-language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 6670–6687, 2025.

[8] Shubo Liu, Hongsheng Zhang, Yuankai Qi, Peng Wang, Yanning Zhang, and Qi Wu. Aerialvln: Vision-and-language navigation for uavs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15384–15394, 2023.

[9] Ruipu Wu, Yige Zhang, Jinyu Chen, Linjiang Huang, Shifeng Zhang, Xu Zhou, Liang Wang, and Si Liu. Aeroduo: Aerial duo for uav-based vision and language navigation. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 2576–2585, 2025.

[10] Yuhang Zhang, Haosheng Yu, Jiaping Xiao, and Mir Feroskhan. Grounded vision-language navigation for uavs with open-vocabulary goal understanding. *arXiv preprint arXiv:2506.10756*, 2025.

[11] Jianqiang Xiao, Yuexuan Sun, Yixin Shao, Boxi Gan, Rongqiang Liu, Yanjin Wu, Weili Guan, and Xiang Deng. Uav-on: A benchmark for open-world object goal navigation with aerial agents. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 13023–13029, 2025.

[12] Xiangyu Wang, Donglin Yang, Ziqin Wang, Hohin Kwan, Jinyu Chen, Wenjun Wu, Hongsheng Li, Yue Liao, and Si Liu. Towards realistic uav vision-language navigation: Platform, benchmark, and methodology. *arXiv preprint arXiv:2410.07087*, 2024.

[13] Shuai Bai, Yuxuan Cai, Rui Chen, Kaixin Chen, Xionghui Chen, Ziyu Cheng, Keming Zhu, et al. Qwen3-vl technical report, 2025.

[14] Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025.

[15] Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, et al. Glm-4.5 v and glm-4.1 v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning. *arXiv preprint arXiv:2507.01006*, 2025.

[16] Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, SH Cai, Yuan Cao, Y Charles, HS Che, Cheng Chen, Guanduo Chen, et al. Kimi k2. 5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*, 2026.

[17] Xiangyu Wang, Donglin Yang, Yue Liao, Wenhao Zheng, wenjun wu, Bin Dai, Hongsheng Li, and Si Liu. UAV-flow colosseo: A real-world benchmark for flying-on-a-word UAV imitation learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025.

[18] Dujun Nie, Xianda Guo, Yiqun Duan, Ruijun Zhang, and Long Chen. Wmnav: Integrating vision-language models into world models for object goal navigation. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2392–2399. IEEE, 2025.

[19] Muhammad Yeasir Arafat, Muhammad Morshed Alam, and Sangman Moh. Vision-based navigation techniques for unmanned aerial vehicles: Review and challenges. *Drones*, 7(2):89, 2023.

[20] Jing Yu Koh, Honglak Lee, Yinfei Yang, Jason Baldridge, and Peter Anderson. Pathdreamer: A world model for indoor navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14738–14748, October 2021.

[21] Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 15791–15801, 2025.

[22] Xuan Yao, Junyu Gao, and Changsheng Xu. Navmorph: A self-evolving world model for vision-and-language navigation in continuous environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5536–5546, 2025.

[23] Changxin Huang, Lv Tang, Zhaohuan Zhan, Lisha Yu, Runhao Zeng, Zun Liu, Zhengjie Wang, and Jianqiang Li. Unemo: Collaborative visual-language reasoning and navigation via a multimodal world model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 18315–18323, 2026.

[24] Shuo Wang, Yongcai Wang, Zhaoxin Fan, Yucheng Wang, Maiyue Chen, Kaihui Wang, Zhizhong Su, Wanting Li, Xudong Cai, Yeying Jin, et al. Monodream: Monocular vision-language navigation with panoramic dreaming. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 10074–10082, 2026.

[25] Xichen Ding, Jianzhe Gao, Cong Pan, Wenguan Wang, and Jie Qin. History-enhanced two-stage transformer for aerial vision-and-language navigation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 18225–18233, 2026.

[26] Jingyu Guo, Ziye Chen, Ziwen Li, Zhengqing Gao, Jiaxin Huang, Hanlue Zhang, Fengming Huang, Yu Yao, Tongliang Liu, and Mingming Gong. Huge-bench: A benchmark for high-level uav vision-language-action tasks. In *European Conference on Computer Vision*, pages 373–388. Springer, 2026.

[27] Yusheng Zhao, Jinyu Chen, Chen Gao, Wenguan Wang, Lirong Yang, Haibing Ren, Huaxia Xia, and Si Liu. Target-driven structured transformer planner for vision-language navigation. In *Proceedings of the 30th ACM international conference on multimedia*, pages 4194–4203, 2022.

[28] Dong An, Hanqing Wang, Wenguan Wang, Zun Wang, Yan Huang, Keji He, and Liang Wang. Etpnav: Evolving topological planning for vision-language navigation in continuous environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(7):5130–5145, 2025.

[29] Lingfeng Zhang, Xiaoshuai Hao, Qinwen Xu, Qiang Zhang, Xinyao Zhang, Pengwei Wang, Jing Zhang, Zhongyuan Wang, Shanghang Zhang, and Renjing Xu. MapNav: A novel memory representation via annotated semantic maps for VLM-based vision-and-language navigation. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13032–13056, Vienna, Austria, July 2025. Association for Computational Linguistics.

[30] Jiazhao Zhang, Anqi Li, Yunpeng Qi, Minghan Li, Jiahang Liu, Shaoan Wang, Haoran Liu, Gengze Zhou, Yuze Wu, Xingxing Li, et al. Embodied navigation foundation model. *arXiv preprint arXiv:2509.12129*, 2025.

[31] Wen Jiang, Li Wang, Kangyao Huang, Wei Fan, Jinyuan Liu, Shaoyu Liu, Hongwei Duan, Bin Xu, and Xiangyang Ji. Longfly: Long-horizon uav vision-and-language navigation with spatiotemporal context integration. *arXiv preprint arXiv:2512.22010*, 2025.

[32] Peican Lin, Gan Sun, Chenxi Liu, Fazeng Li, Weihong Ren, and Yang Cong. Openvln: Open-world aerial vision-language navigation. *arXiv preprint arXiv:2511.06182*, 2025.

[33] Hanqing Wang, Wei Liang, Luc Van Gool, and Wenguan Wang. Dreamwalker: Mental planning for continuous vision-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10873–10883, October 2023.

[34] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13753–13762, June 2024.

[35] Yingyan Li, Shuyao Shang, Weisong Liu, Bing Zhan, Haochen Wang, Yuqi Wang, Yuntao Chen, Xiaoman Wang, Yasong An, Chufeng Tang, et al. Drivevla-w0: World models amplify data scaling law in autonomous driving. In *International Conference on Learning Representations*, volume 2026, pages 7890–7911, 2026.

[36] Weichen Zhang, Peizhi Tang, Xin Zeng, Fanhang Man, Shiquan Yu, Zichao Dai, Baining Zhao, Hongjin Chen, Yu Shang, Wei Wu, et al. Aerial world model for long-horizon visual generation and navigation in 3d space. *arXiv preprint arXiv:2512.21887*, 2025.

[37] Wenxuan Song, Ziyang Zhou, Han Zhao, Jiayi Chen, Pengxiang Ding, Haodong Yan, Yuxin Huang, Feilong Tang, Donglin Wang, and Haoang Li. Reconvla: Reconstructive vision-language-action model as effective robot perceiver. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 18549–18557, 2026.

[38] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

[39] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3674–3683, 2018.

A Additional Implementation Details

A.1 Stop Supervision

We design a simple stop supervision strategy to exploit the VLM's internal grounding ability for deciding when the UAV should terminate navigation. We supervise the stop head with a distance-aware soft label, encouraging the VLM to ground the stopping decision in the current visual observation and its distance to the target.

Given the output hidden state of the <stop> token, we predict the stopping probability with a linear layer followed by a sigmoid activation:

$$p_{\text{stop}} = \sigma\left(\text{Linear}(\boldsymbol{h}_{stop})\right), \tag{10}$$

where $\boldsymbol{h}_{stop}$ is the hidden state of <stop> and $p_{\text{stop}} \in [0, 1]$ is the predicted stop probability.

During training, we construct the supervision label according to the distance d between the current UAV position and the target position:

$$y_{\text{stop}} = \sigma\left(\frac{20 - d}{5}\right). \tag{11}$$

This soft label assigns $y_{\text{stop}} = 0.5$ when the UAV is 20 meters away from the target. As the UAV gets closer to the target, y_{stop} increases toward 1; when it is farther away, the label decreases toward 0. We train the stop head with binary cross-entropy:

$$\mathcal{L}_{stop} = \text{BCE}\left(p_{\text{stop}}, y_{\text{stop}}\right). \tag{12}$$

During inference, the UAV sends a stop signal when:

$$p_{\text{stop}} \geq 0.75. \tag{13}$$

This design provides a soft and distance-aware stopping objective, encouraging the model to stop only when its visual-language grounding indicates that the UAV is sufficiently close to the target.

A.2 Waypoint Head

Following TravelUAV [12], we use the same waypoint head design to predict a coarse waypoint from the hidden state of the <wp> token and the UAV kinematic context. Let $\boldsymbol{h}_{wp} \in \mathbb{R}^{B\times D}$ denote the output hidden state at the <wp> token position, where B is the batch size and D is the token hidden dimension of the VLM. Since Qwen3VL-2B-Instruct uses a hidden size of 2048, we set $D = 2048$.

Besides the <wp> token feature, the waypoint head also uses the historical trajectory and the current UAV orientation. For each sample, the historical trajectory is represented as

$$\boldsymbol{H}_i = \{\boldsymbol{p}_{i,1}, \boldsymbol{p}_{i,2}, \ldots, \boldsymbol{p}_{i,T_i}\}, \quad \boldsymbol{p}_{i,t} \in \mathbb{R}^3, \tag{14}$$

where T_i is the trajectory length of the i-th sample. Each historical position is first encoded by an MLP, and then averaged over time:

$$\boldsymbol{h}_{hist} = \frac{1}{T_i}\sum_{t=1}^{T_i} \text{MLP}_{hist}(\boldsymbol{p}_{i,t}), \quad \boldsymbol{h}_{hist} \in \mathbb{R}^{B\times D}. \tag{15}$$

The UAV orientation $\boldsymbol{O} \in \mathbb{R}^{B\times 3}$ is encoded by another MLP:

$$\boldsymbol{h}_{ori} = \text{MLP}_{ori}(\boldsymbol{O}), \quad \boldsymbol{h}_{ori} \in \mathbb{R}^{B\times D}. \tag{16}$$

The kinematic context is obtained by summing the trajectory and orientation features:

$$\boldsymbol{h}_{ctx} = \boldsymbol{h}_{hist} + \boldsymbol{h}_{ori}. \tag{17}$$

We combine the <wp> token feature, the kinematic context, and a learnable waypoint embedding:

$$\boldsymbol{h} = \boldsymbol{h}_{wp} + \boldsymbol{h}_{ctx} + \boldsymbol{e}_{wp}, \tag{18}$$

where $\boldsymbol{e}_{wp} \in \mathbb{R}^D$ is the learnable waypoint embedding. An MLP predicts the coarse waypoint:

$$\hat{\boldsymbol{w}} = \text{MLP}_{wp}(\boldsymbol{h}), \quad \hat{\boldsymbol{w}} \in \mathbb{R}^{B\times 4}. \tag{19}$$

The trajectory completion module converts the waypoint to an executable short-horizon trajectory.

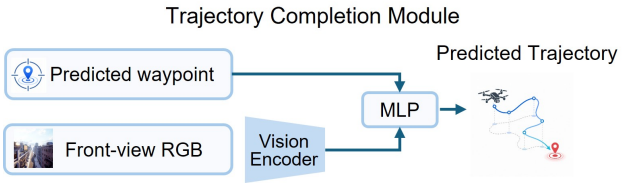


Figure 8: Illustration of the trajectory completion module. The predicted coarse waypoint and the current front-view RGB image are fused to generate a fine-grained local trajectory for UAV execution.

A.3 Trajectory Completion

Following TravelUAV [12], we use the same trajectory completion module to convert a coarse waypoint into a short local trajectory, as illustrated in Fig. 8. In the dataset, each coarse waypoint is associated with a short trajectory segment. Specifically, one waypoint corresponds to 5 local trajectory steps used for execution. Therefore, after the waypoint head predicts a coarse waypoint, we use the trajectory completion module to convert this single waypoint into a short local trajectory. During training, the module is supervised to predict 7 local trajectory points for a richer short-horizon signal, while during execution we only use the first 5 predicted points to control the UAV.

The trajectory completion module can be viewed as a visual-assisted trajectory refiner: given the coarse waypoint predicted by the LLM-based waypoint head, it uses the current front-view image to complete a short local trajectory. This module does not participate in the VLM forward pass and does not read the language prompt.

Let $\boldsymbol{I}_t^{front}$ denote the current front-view image, and let $\boldsymbol{w}_{llm} \in \mathbb{R}^3$ denote the coarse waypoint predicted by the waypoint head and transformed into the current UAV coordinate system. We first extract visual features from the front-view image using a ViT encoder:

$$\boldsymbol{V}_t = \mathrm{ViT}(\boldsymbol{I}_t^{front}). \tag{20}$$

After removing the CLS token, the remaining patch features are passed through a visual MLP and then averaged to obtain a compact image feature:

$$\boldsymbol{f}_{img} = \mathrm{Pool}\left(\mathrm{MLP}_{vis}(\boldsymbol{V}_t^{patch})\right). \tag{21}$$

The image feature and the coarse waypoint are then concatenated:

$$\boldsymbol{z} = [\boldsymbol{f}_{img}; \boldsymbol{w}_{llm}]. \tag{22}$$

Finally, an MLP predicts a sequence of local trajectory points:

$$\hat{\boldsymbol{T}} = \mathrm{MLP}_{traj}(\boldsymbol{z}), \quad \hat{\boldsymbol{T}} \in \mathbb{R}^{7\times 3}. \tag{23}$$

The predicted trajectory contains 7 local points during training. At inference time, we take the first 5 points from $\hat{\boldsymbol{T}}$, transform them back to the world coordinate system, and use them for UAV execution in AirSim. This module does not re-interpret the instruction or re-localize the target. Instead, it refines the coarse waypoint into a smoother and more executable short-horizon trajectory using the current front-view visual context.

B PVI Spatial Consistency and Waypoint Accuracy

We evaluate spatial consistency on 1,000 Test Seen and 1,000 Test Unseen frames from OpenUAV, each at 256×256 resolution. For each frame, we compute a cosine-similarity map between the predicted PVI feature and local image-patch features, then apply soft-argmax to obtain the PVI-indicated location. PVI–GT measures its pixel distance to the projected expert target, PVI–WP measures its distance to the projected predicted waypoint, and WP–GT measures the distance between the projected predicted and expert waypoints. PVI–GT uses all 2,000 frames; comparisons involving projected waypoints use 1,959 valid frames.

Table 5: Spatial alignment on OpenUAV. Values are percentages with Wilson 95% confidence intervals in brackets.

Model	Relation	$\leq$ 16 px	$\leq$ 24 px	$\leq$ 32 px
Test Seen				
ProtoVis-Nav	PVI–GT	55.10 [52.00, 58.16]	81.90 [79.39, 84.16]	87.20 [84.99, 89.13]
	PVI–WP	53.93 [50.80, 57.03]	80.29 [77.68, 82.66]	86.11 [83.80, 88.13]
	WP–GT	82.12 [79.60, 84.40]	87.54 [85.32, 89.46]	92.54 [90.73, 94.03]
Base	WP–GT	81.70 [79.15, 84.00]	87.12 [84.87, 89.07]	92.23 [90.38, 93.75]
Test Unseen				
ProtoVis-Nav	PVI–GT	58.00 [54.92, 61.02]	79.80 [77.20, 82.17]	88.40 [86.27, 90.24]
	PVI–WP	57.55 [54.43, 60.61]	81.53 [78.98, 83.84]	90.10 [88.07, 91.82]
	WP–GT	80.51 [77.91, 82.87]	86.43 [84.14, 88.43]	90.92 [88.96, 92.56]
Base	WP–GT	79.92 [77.30, 82.31]	86.34 [84.05, 88.35]	90.21 [88.19, 91.92]

At the 32-pixel threshold, PVI–GT alignment reaches 87.20% on Test Seen and 88.40% on Test Unseen, and PVI–WP alignment reaches 86.11% and 90.10%, respectively (Table 5). ProtoVis-Nav also improves WP–GT alignment over Base at all three thresholds on both splits. These results show spatial agreement among PVI, the projected target region, and the predicted waypoint.

We further measure mean waypoint prediction error on 1,000 sampled OpenUAV test frames. As shown in Table 6, PVI reduces the mean error from 0.4630m to 0.4576m, corresponding to a 5.4 mm improvement in the continuous waypoint output.

Table 6: Mean waypoint prediction error on 1,000 sampled OpenUAV test frames.

Model	Mean waypoint error (m) $\downarrow$
Base	0.4630
ProtoVis-Nav	**0.4576**

C PVI Feature Similarity and Retrieval

We evaluate the same 1,000 Seen and 1,000 Unseen frames used in Table 5. Each 256×256 frame is densely sampled into 841 candidate patches of size 32×32 with a stride of 8 pixels.

First, we measure the direct cosine similarity between the predicted PVI representation and the ground-truth target-patch feature, using a randomly selected patch as the baseline. Second, we rank all candidate patches by cosine similarity and report the first-relevant rank. A candidate is considered relevant if its center is less than 32 pixels from the ground-truth target center, within one visual-token width. We report spatial Recall@1, Recall@5, and Recall@10 under the same criterion.

Table 7: Direct PVI feature similarity and spatial retrieval accuracy. Values in brackets are 95% confidence intervals.

Metric	Test Seen	Test Unseen
PVI–GT cosine similarity	0.8440 [0.8402, 0.8477]	0.8449 [0.8415, 0.8482]
PVI–random cosine similarity	0.7666 [0.7604, 0.7728]	0.7535 [0.7464, 0.7605]
Mean first-relevant rank (< 32 px) $\downarrow$	11.19 [9.22, 13.57]	15.56 [13.38, 18.05]
Recall@1 (< 32 px)	27.00% [24.34%, 29.84%]	21.10% [18.68%, 23.74%]
Recall@5 (< 32 px)	63.50% [60.47%, 66.43%]	54.90% [51.80%, 57.96%]
Recall@10 (< 32 px)	80.40% [77.83%, 82.74%]	69.30% [66.37%, 72.08%]

As shown in Table 7, the PVI–target cosine similarity reaches 0.8440 on Seen and 0.8449 on Unseen, compared with 0.7666 and 0.7535 for random patches. Among 841 candidate patches, the mean first-relevant rank is 11.19 on Seen and 15.56 on Unseen. At the 32-pixel threshold, Recall@1/5/10 reaches 27.00%/63.50%/80.40% on Seen and 21.10%/54.90%/69.30% on Unseen. These results show that the PVI representation contains information about the ground-truth target patch.

D Robustness to Observation and Execution Perturbations

We evaluate ProtoVis-Nav on OpenUAV Test Seen under three perturbations. Camera image noise affects the visual inputs, pose estimation noise affects the position and yaw estimates supplied to the model, and execution mismatch introduces action execution errors and horizontal drift.

Table 8: Robustness of ProtoVis-Nav to observation and execution perturbations on OpenUAV Test Seen.

Variant	NE (m) $\downarrow$	SR (%) $\uparrow$	OSR (%) $\uparrow$	SPL (%) $\uparrow$
ProtoVis-Nav	**51.77**	**50.78**	**66.50**	**44.68**
+ Camera image noise	52.33	49.51	64.80	44.17
+ Position and yaw noise	56.24	46.42	63.11	41.25
+ Execution mismatch	59.92	47.69	62.41	42.02

As shown in Table 8, camera image noise has the smallest effect on SR and SPL, reducing them from 50.78% and 44.68% to 49.51% and 44.17%. Pose estimation noise produces the lowest SR (46.42%), and execution mismatch produces the highest NE (59.92m), illustrating the effect of state estimation and execution accuracy on closed-loop navigation.

E Evaluation on UAV-Flow

We compare ProtoVis-Nav with Base on UAV-Flow [17]. Normalized Dynamic Time Warping (nDTW) measures trajectory alignment. We use double-blind human annotation to determine the success rate based on manual inspection of whether the trajectory semantically satisfies the instruction, consistent with official UAV-Flow settings.

Table 9: Closed-loop results on UAV-Flow.

Model	Trajectories	nDTW ↑	Successes	SR (%) ↑
Base	273	0.2971	185	67.77
ProtoVis-Nav	273	**0.3104**	**197**	**72.16**

On the full test split of UAV-Flow, ProtoVis-Nav improves closed-loop nDTW from 0.2971 to 0.3104 and SR from 67.77% to 72.16% over Base (Table 9), indicating closer alignment with the reference trajectories and higher task success. UAV-Flow evaluates fine-grained flight skills, so these gains suggest that target-region visual features also provide useful context for short-range motion control.

F Backbone Comparison

We compare Qwen3-VL and Wan2.1-VACE in a controlled backbone-transfer experiment on OpenUAV Test Seen, using the same PVI design, waypoint heads, action representation, training data, and evaluation protocol. Inference latency is measured for a single forward pass on the same GPU.

Table 10: Navigation performance and inference latency of different backbones on OpenUAV Test Seen.

Backbone	NE (m) ↓	SR (%) ↑	OSR (%) ↑	SPL (%) ↑	Latency (ms) ↓
Wan2.1-VACE	111.46	17.25	22.30	14.79	333.83
Qwen3-VL	**51.77**	**50.78**	**66.50**	**44.68**	**185.48**

As shown in Table 10, Qwen3-VL increases SR from 17.25% to 50.78% and SPL from 14.79% to 44.68%, while reducing latency from 333.83ms to 185.48ms. World action models (WAMs) offer explicit physical foresight for longer-horizon planning, with diffusion-based generation incurring iterative inference costs. ProtoVis-Nav trades this multi-step foresight for efficient local waypoint prediction, conditioning the next waypoint on an outcome-associated local visual representation in a single forward pass.

G Effect of inference-time PVI interventions

We further examine the contribution of PVI context through inference-time interventions. We evaluate four PVI intervention settings on Test Seen Full. *Intact* uses the PVI representations predicted for the current sample; *Shuffled* exchanges these representations across samples; *Masked* blocks PVI information from reaching the waypoint predictor; and *Random* replaces it with random features. As shown in Table 11, Intact achieves the highest SR, OSR, and SPL and the lowest NE across these settings. Shuffling reduces SR from 50.78% to 41.09%, while masking and random replacement further reduce it to 37.87% and 36.75%, respectively. These results show that the waypoint predictor uses sample-specific PVI information during inference, with coherent local visual context contributing to successful navigation.

Table 11: Effect of inference-time PVI interventions on OpenUAV Test Seen Full.

Model	PVI setting	NE (m) ↓	SR (%) ↑	OSR (%) ↑	SPL (%) ↑
Base	–	55.67	46.47	62.69	41.14
ProtoVis-Nav	Intact	**51.77**	**50.78**	**66.50**	**44.68**
	Shuffled	57.36	41.09	54.56	36.88
	Masked	70.99	37.87	49.65	34.74
	Random	77.18	36.75	47.69	33.38

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: We summarize the main contributions in the Introduction section.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss the limitations in Section 4.7.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Yes, the paper fully discloses all necessary methodological details to reproduce the main experimental results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

Answer: [Yes]

Justification: We provide the source code via the GitHub link in the abstract.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We detail the experimental setup, including data splits, hyperparameters, and optimizer choices, in the experiment section.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: We evaluate our model using standard metrics (NE, SR, OSR, SPL) on established VLN benchmarks, where reporting standard deviations is not the standard protocol.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We report the exact GPU models and the number of GPUs used for training and testing.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conducted in this paper conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The paper discusses both limitations and social impact in Section 4.7. It notes positive potential applications of ProtoVis-Nav, such as search-and-rescue, disaster response, and infrastructure inspection. It also discusses possible negative impacts of UAV navigation systems, including safety, privacy, and security concerns if deployed without sufficient safeguards, and recommends human oversight and appropriate deployment constraints.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We properly cite the original creators of all datasets, simulators, and pre-trained models used in our experiments, and we strictly adhere to their respective licenses and terms of use.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: The model proposed in this paper is clearly documented.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

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

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [N/A]

    Justification: We only use LLM to spot grammar issues within the script.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.