VoLN: Vision-Only Long-Horizon Navigation —Paradigm, Benchmark, and Method

Anonymous authors
Paper under double-blind review

Abstract

Vision-and-Language Navigation (VLN) guides agents with language instructions, while image-goal navigation specifies destinations through images. We propose Vision-Only Long-Horizon Navigation (VoLN), a paradigm for long-horizon navigation in open environments with visual goals and no separate language-instruction input. Semantic beacons embedded in the scene provide cues for sustained reasoning as the agent encounters them along a route, analogous to how people consult signs during navigation. The agent must interpret successive cues in relation to its goal and observation history and update local plans. We instantiate VoLN for aerial navigation through VoLN-UAV, a 7,210-episode benchmark across 17 simulated environments, combining continuous 3D flight, substantial viewpoint variation, and held-out scene evaluation. Active beacons provide local guidance and passive beacons introduce competing cues, creating an intermediate setting between instruction-guided VLN and image-goal navigation. We further present VoLN-MLLM, a two-stage visual–semantic planning framework that aligns self-supervised visual features with a structured semantic space and predicts short-horizon waypoints and stopping decisions from observation history, goal views, retrieved concepts, and proprioception. On Test-Unseen, it achieves success rates of 7.4%, 4.5%, and 1.8% on Easy, Normal, and Hard episodes, respectively. These results highlight remaining challenges in integration of observation history and reliable long-horizon execution. Code is available at `https://anonymous.4open.science/r/VoLN-UAV-DDC6/`.

1 Introduction

Navigation in open environments requires decisions that remain consistent over extended trajectories. A destination may remain outside the current field of view, and the appearance of local landmarks can change substantially as an agent moves. Locally available observations provide only a partial view of the environment, while small planning errors can accumulate over time. Reaching a visually specified destination therefore involves interpreting successive scene cues in relation to the goal, integrating observation history, and revising local plans as navigation unfolds.

We introduce Vision-Only Long-Horizon Navigation (VoLN), a paradigm for long-horizon navigation and sustained visual–semantic reasoning in open environments. VoLN specifies the destination through goal images, without the separate language-instruction input used in Vision-and-Language Navigation (VLN) (Anderson et al., 2018b). It adopts an image-goal interface for navigation in unfamiliar environments and focuses on connecting successive scene cues with goal-directed decisions over extended trajectories.

Semantic beacons embedded in the scene act as reasoning cues encountered through egocentric imagery, analogous to signs that people consult along a route. At successive stages, the agent interprets these signals against its goal and observation history to update the next action sequence. This design provides an intermediate setting between instruction-guided VLN and image-goal navigation.

We instantiate VoLN in aerial navigation through VoLN-UAV, a benchmark for unmanned aerial vehicles (UAVs) with 7,210 episodes from 17 simulated environments. Aerial navigation introduces substantial changes in altitude, viewpoint, and apparent scale, while flight dynamics constrain continuous 3D trajectories. Sparse active beacons provide route-relevant guidance, while passive

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

beacons with similar visual forms introduce competing cues. Evaluation in held-out environments jointly tests cross-view goal matching, context-dependent cue selection, and closed-loop control through measurements of goal convergence, trajectory quality, and execution reliability.

We further present VoLN-MLLM, a two-stage visual–semantic planning framework. The first stage aligns self-supervised visual features with a shared semantic space for observations, goal views, and visible scene cues. The second combines recent observations, goal views, retrieved concepts, and proprioception to predict short-horizon waypoint segments and stopping decisions. The policy executes each segment and replans from subsequent observations. Our contributions are:

- **Paradigm.** We formulate VoLN for long-horizon navigation in open environments, where visually encountered semantic cues support sustained, multi-stage reasoning toward image-specified goals.
- **Benchmark.** We introduce VoLN-UAV, a 7,210-episode aerial benchmark with continuous 3D flight, active and passive beacons, and seen/unseen environment splits.
- **Method.** We present VoLN-MLLM, a two-stage visual–semantic planning framework that aligns visual representations with a semantic bank and predicts short-horizon trajectories in closed loop.

2 Related Work

2.1 Navigation task interfaces

The task interface determines how intent reaches the agent and what must be inferred during execution. In VLN, language grounding connects instructions to sequential actions (Anderson et al., 2018b). Explicit language-model reasoning supports action selection from descriptions of observations and navigation history (Zhou et al., 2024). To organize decisions across a route, MapGPT provides topological context for multi-step planning (Chen et al., 2024), while task-activated skill libraries connect language-level plans to executable behaviors (Lou et al., 2025). Recent foundation models extend this direction across tasks and embodiments through shared multimodal navigation representations (Zhang et al., 2026).

Image-goal navigation specifies the destination visually, making cross-view correspondence central to goal localization. Such correspondence can emerge in end-to-end policies through visual pretraining (Bono et al., 2024). Explicit reconstruction provides another approach: GaussNav preserves scene geometry and appearance for instance-image matching (Lei et al., 2025). IGL-Nav incrementally constructs a Gaussian representation for localization against the goal image (Guo et al., 2025). Together, these studies support visual goal localization through learned representations or explicit scene reconstruction. VoLN connects this image-goal formulation with the semantic guidance of VLN, providing an intermediate setting for sustained, closed-loop decision-making in open environments. It retains image-based goal specification and provides guidance through semantic beacons encountered within the scene, without separate language instructions.

2.2 Aerial navigation

Aerial navigation couples scene interpretation with continuous 3D motion. Feasible flight can be addressed through constrained trajectory optimization, as in HHPSO (Lou et al., 2024), or learned control, exemplified by reinforcement learning for drone racing (Kaufmann et al., 2023). Simulation supports controlled evaluation of visual flight policies under onboard resource constraints (Krishnan et al., 2021). Platforms such as Air-M extend this infrastructure to large-scale multi-agent learning (Lou et al., 2023).

At the task level, AerialVLN connects language instructions with navigation through aerial scenes (Liu et al., 2023). Realistic vehicle dynamics are incorporated in OpenUAV (Wang et al., 2025), while OpenFly automates outdoor trajectory and instruction collection (Gao et al., 2026). Policies must then translate semantic intent into executable flight. End-to-end multimodal grounding supports this translation in UAV-VLN (Saxena et al., 2025). Hierarchical subgoals and global memory maintain context across longer routes (Zhang et al., 2025b); supervised and reinforcement fine-tuning further support structured decision making in FlightGPT (Cai et al., 2025). These efforts

connect semantic task understanding with executable flight through advances in control, simulation, and navigation policies. VoLN-UAV examines this connection under visual goals and semantic beacons, focusing on sustained cue interpretation and closed-loop flight over long routes in open environments.

2.3 VISUAL–SEMANTIC GROUNDING AND PLANNING

Scene semantics help determine where to move before a target becomes visible. For object-goal search, vision–language value maps guide frontier selection (Yokoyama et al., 2024). Pixel-guided navigation connects visual target selection to local movement (Cai et al., 2024), while multi-channel score maps support searching for several objects (Choi et al., 2025). For destinations identified by textual labels, signs and numbering patterns provide explicit spatial guidance (Cao et al., 2026).

Extended execution requires preserving and revising scene context. Text-based maps expose spatial semantics to language-model planners (Zhang et al., 2025a). Experience-dependent updates allow navigation behavior to adapt after unexpected events (Kim et al., 2025), and spatio-temporal retrieval provides access to longer observation histories (Anwar et al., 2025).

Prediction complements memory when routes enter unobserved regions. Generative semantic maps estimate likely target locations (Zhang et al., 2024), while occupancy and semantic prediction support exploration-goal selection (Shah et al., 2025). At the observation level, NavigateDiff predicts future views to condition image-goal policies (Qin et al., 2025). WMNav uses predicted outcomes and subsequent feedback to revise object-goal navigation decisions (Nie et al., 2025). Semantic grounding, memory, and prediction offer complementary ways to relate local observations to task context and future actions. VoLN-MLLM focuses on aligning goal views and observations within a shared semantic space, then combining retrieved concepts with observation history to generate short-horizon trajectories in closed loop.

3 THE VOLN PARADIGM

VoLN is a goal-directed navigation paradigm for long-horizon, multi-stage reasoning in open environments. Each episode provides a visual goal set $\mathcal{V}$ composed of images captured near the destination. The agent grounds these views across viewpoint changes, interprets semantic cues encountered along the route in the context of its goal and history, and updates its actions through closed-loop interaction. The policy receives goal views, onboard observations, and proprioception, without task-level route instructions or global navigation signals. Figure 1 summarizes the interaction; Fig. 6 in Appendix A shows individual observations from the same urban episode.

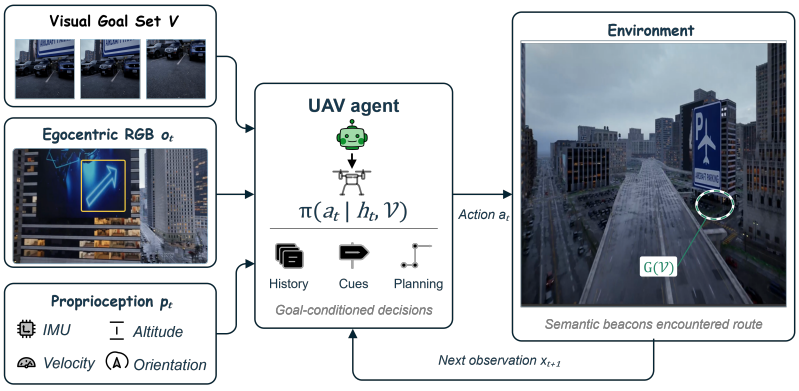


Figure 1: VoLN interaction in an urban VoLN-UAV episode. The policy conditions actions a_t on goal views $\mathcal{V}$, observations o_t, and proprioception p_t; $G(\mathcal{V})$ denotes the evaluation goal region.

We model each episode as a partially observable sequential decision process with latent state $s_t \in \mathcal{S}$. At time step t, the agent receives an observation $x_t = (o_t, p_t)$, where o_t denotes the egocentric RGB observation and p_t denotes proprioception, implemented as platform-provided onboard signals

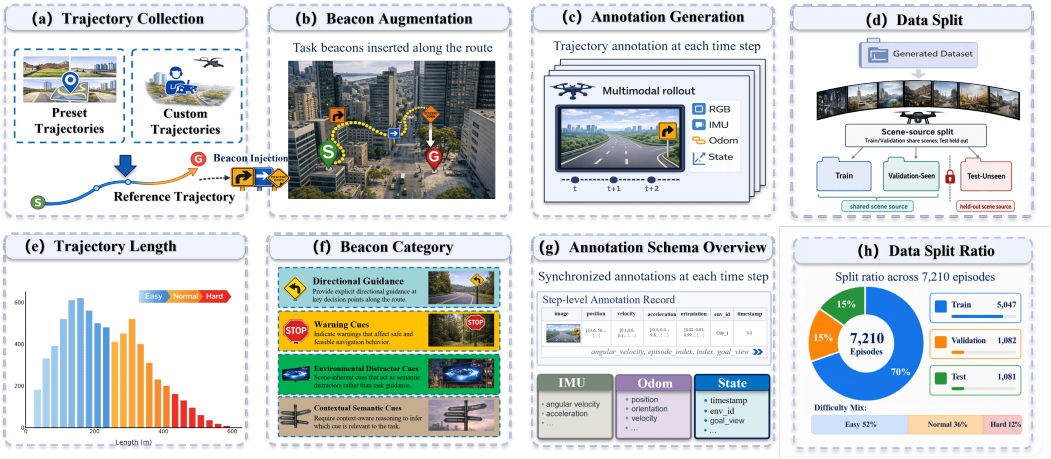


Figure 2: Overview of VoLN-UAV. Panels (a)–(d) show trajectory collection, beacon augmentation, annotation generation, and scene-source splitting; panels (e)–(h) summarize trajectory length, beacon categories, annotation fields, and split statistics.

such as IMU measurements, altitude, velocity, and orientation; GPS and world-frame position are excluded. The agent outputs an action $a_t \in \mathcal{A}$, which may be continuous or discrete and includes an explicit stop decision, and the environment evolves according to the transition kernel P and observation function Ω:

$$s_{t+1} \sim P(\cdot \mid s_t, a_t), \qquad x_{t+1} \sim \Omega(\cdot \mid s_{t+1}). \tag{1}$$

The policy conditions on interaction history $h_t = (x_0, a_0, \ldots, x_t)$ and selects actions according to

$$a_t \sim \pi(\cdot \mid h_t, \mathcal{V}). \tag{2}$$

VoLN targets episodes in which the destination remains outside the current view over substantial portions of the trajectory and route-relevant cues are encountered at multiple decision points. At successive stages, the agent connects cue interpretation with goal matching and local action selection. The policy uses h_t to integrate observations across the trajectory, interpreting semantic beacons and naturally occurring landmarks in o_t in relation to $\mathcal{V}$ and the accumulated interaction context. For evaluation, each task instance associates $\mathcal{V}$ with a goal region $G(\mathcal{V}) \subset \mathcal{S}$. Starting from a designated initial pose, the agent has at most T steps and succeeds by issuing the stop action inside $G(\mathcal{V})$.

4 The VoLN-UAV Benchmark

4.1 Simulation Environments

VoLN-UAV instantiates the VoLN paradigm in aerial navigation using Unreal Engine and Microsoft AirSim for high-fidelity rendering and continuous 3D flight. The environment pool contains 17 distinct environments drawn from selected scenes adapted from existing open-source aerial VLN benchmarks, including AerialVLN (Liu et al., 2023) and OpenUAV (Wang et al., 2025), together with additional custom-built environments, $\mathcal{E} = \mathcal{E}^{\text{open}} \cup \mathcal{E}^{\text{custom}}$. This hybrid design preserves compatibility with existing aerial benchmarks while expanding scene diversity. As shown in Fig. 7 in Appendix A.2, the benchmark spans natural and built environments, from deserts, forests, and mountains to urban canyons, tunnels, and industrial corridors, with substantial variation in layout, visibility, altitude change, and landmark density.

4.2 Benchmark Construction Pipeline

We instantiate VoLN-UAV through the trajectory-centric pipeline illustrated in Fig. 2. The following paragraphs detail its construction stages and resulting data organization.

Trajectory collection. Panels (a) and (e) relate reference-route collection to the resulting path-length distribution. For each episode, we sample a scene from $\mathcal{E}$ and select a reference route from the corresponding route pool. The pool combines predefined trajectories derived from AerialVLN (Liu et al., 2023) with custom trajectories collected by trained human operators under a standardized recording protocol. The predefined trajectories follow the route and goal settings of the source dataset, while the custom trajectories expand coverage of underrepresented flight patterns, including long corridors, sharp turns, altitude transitions, and ambiguous junctions. VoLN-UAV stratifies episodes by accumulated reference-path length, $L_{\mathrm{ref}}(\xi) = \sum_{t=1}^{T} \|\mathbf{r}_t - \mathbf{r}_{t-1}\|_2$, where $\mathbf{r}_t$ denotes the reference position at step t. Episodes are labeled **Easy** ($L_{\mathrm{ref}} < L_1$), **Normal** ($L_1 \le L_{\mathrm{ref}} < L_2$), or **Hard** ($L_{\mathrm{ref}} \ge L_2$), with $L_1 = 300\,\mathrm{m}$ and $L_2 = 450\,\mathrm{m}$ across all splits.

Beacon augmentation and cue categories. Panels (b) and (f) show beacon placement and the corresponding cue categories. Each reference trajectory is augmented with three to five active beacons, sparsely placed at decision points as task-relevant cues. Each environment is additionally populated with approximately 150 passive beacons that remain fixed across episodes and provide semantic clutter. The beacons span four semantic categories: directional guidance, warning cues related to feasible flight, environmental distractors, and contextual cues whose relevance depends on the current task. During execution, the UAV accesses beacons only through its egocentric observation o_t, which may contain both task-relevant active beacons and passive beacons present in the scene.

Multimodal rollout and annotation schema. Panels (c) and (g) present the synchronized multimodal rollout and its step-level annotation schema. For each reference trajectory $\xi = \{s_0, \ldots, s_T\}$, the simulator records synchronized egocentric RGB observations $\{o_0, \ldots, o_T\}$ and proprioceptive states p_t at a fixed interval $\Delta t = 2\,\mathrm{s}$. The final three RGB observations form the episode-level visual goal set:

$$\mathcal{V}(\xi) = \{o_{T-2}, o_{T-1}, o_T\}. \tag{3}$$

For each step t, the observation and proprioceptive state are paired with the episode-level goal set $\mathcal{V}(\xi)$ to form a training sample, while the following H states along the reference route define the short-horizon waypoint target $W_{t:t+H}$.

Dataset split and statistics. Panels (d) and (h) summarize the scene-source split and episode distribution. The dataset contains 7,210 episodes over 17 distinct environments: **Train** contains 5,047 episodes from 12 environments; **Validation-Seen** (VS) contains 1,082 episodes whose trajectories are disjoint from training but are drawn from 5 environments within the training pool; and **Test-Unseen** (TU) contains 1,081 episodes from 5 additional environments belonging to a held-out scene source. The splits correspond to an approximately 70%/15%/15% episode ratio, and the aggregate difficulty mix is 52% Easy, 36% Normal, and 12% Hard.

5 The VoLN-MLLM Method

VoLN requires visual–semantic grounding and closed-loop trajectory generation: the agent must interpret locally observed route cues in relation to the goal views and generate executable waypoint segments from these observations. We address these requirements with **VoLN-MLLM**, a two-stage visual–semantic planning framework (Fig. 3). Phase I aligns DINO visual features with CLIP's joint image–text space, allowing observations to retrieve relevant concepts from a fixed semantic bank. Phase II employs a pretrained language-model planner. Aligned visual features from the recent observation history and goal views, retrieved visual–semantic tokens, and proprioception are projected into the planner's embedding space and jointly encoded for short-horizon waypoint and stopping prediction. The predicted segment is executed by a low-level flight controller, and the model replans from the subsequent observation. The semantic bank is constructed offline and shared across episodes.

Visual–Semantic Alignment. Given an observation o_t, a frozen DINO backbone extracts a visual representation. A lightweight trainable adapter, following the cross-space alignment principle of Talking to DINO (Barsellotti et al., 2025), maps this representation into the CLIP image-embedding space, producing a normalized student embedding E_{stu}. During training of the adapter, a frozen

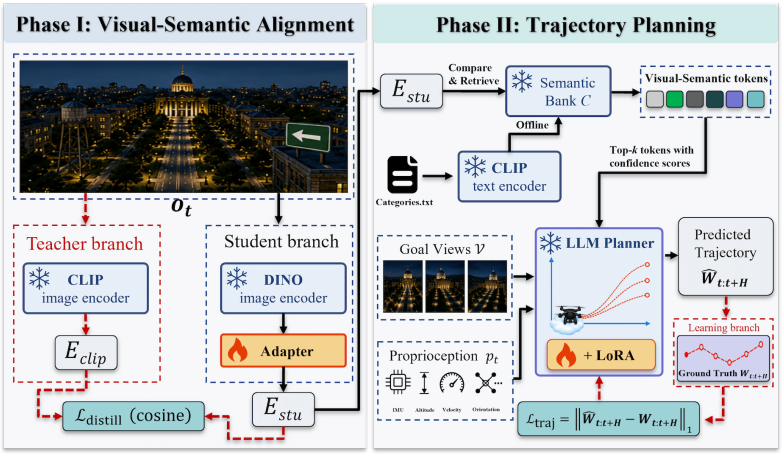


Figure 3: VoLN-MLLM overview. Phase I learns visual–semantic alignment; Phase II predicts short-horizon waypoints and stopping decisions. Dashed arrows indicate training branch; the stopping head is omitted for clarity.

CLIP image encoder provides the normalized teacher embedding E_{clip} for the same image. The adapter is optimized by a distillation objective:

$$\mathcal{L}_{\text{distill}} = \ell(E_{\text{stu}}, E_{\text{clip}}), \tag{4}$$

where $\ell(\cdot,\cdot)$ is instantiated as cosine distance. This stage updates only the adapter parameters and keeps the DINO backbone and CLIP encoders fixed.

Visual–semantic tokenization. The semantic-bank specification comprises 64 textual category descriptors, grouped in Table 4. A frozen CLIP text encoder maps these descriptors to offline embeddings. At each time step, the aligned visual embedding is compared with the bank embeddings using cosine similarity, and the top-k entries are retained. Their category identifiers are encoded with the planner tokenizer, while their similarity scores are mapped to learned confidence embeddings. The resulting visual–semantic tokens are passed to the planner.

Trajectory decoding. At each decision step, the aligned features of a fixed window of recent observations and the goal views are mapped to the planner's embedding dimension by a shared visual projector. A separate state projector maps proprioception p_t to a state token. These embeddings are concatenated with the retrieved visual–semantic tokens and learned structural tokens marking the goal, history, semantics, state, and planning fields. The language-model planner jointly encodes this sequence, and the hidden state of the planning token is passed to a trajectory head and a binary stopping head. The two heads predict a short-horizon sequence of relative waypoints $\hat{W}_{t:t+H}$ in the current UAV body frame and stopping probability $\hat{z}_t$, respectively. The predicted sequence is tracked by the shared low-level flight controller, after which the next observation is acquired and the planner is invoked again.

Planner adaptation and supervision. We keep the pretrained language-model backbone frozen and adapt its attention and feed-forward projections using low-rank adaptation (LoRA) (Hu et al., 2022). The visual and state projectors, trajectory head, and stopping head are trained jointly with the LoRA parameters. At each step, the trajectory head predicts a segment of length H, supervised against demonstrations with an ℓ_1 loss:

$$\mathcal{L}_{\text{traj}} = \left\| \hat{W}_{t:t+H} - W_{t:t+H} \right\|_1, \tag{5}$$

where $\hat{W}_{t:t+H}$ denotes the predicted waypoint sequence and $W_{t:t+H}$ denotes the corresponding demonstrated sequence. Let $z_t \in \{0,1\}$ indicate whether the reference state lies inside the success region. The stopping head is trained with binary cross-entropy, and the complete objective is

$$\mathcal{L} = \mathcal{L}_{\text{traj}} + \lambda_{\text{stop}} \, \text{BCE}(\hat{z}_t, z_t). \tag{6}$$

At inference time, the policy stops when $\hat{z}_t$ exceeds a threshold τ selected on Validation-Seen; otherwise, it executes the predicted waypoint segment and replans.

6 EXPERIMENTS

6.1 EXPERIMENTAL SETUP

Implementation details. All methods use the VoLN-UAV inputs of egocentric RGB, proprioception, and $\mathcal{V}$; world-frame poses are restricted to supervision and evaluation. At each decision step, the policy predicts $H = 8$ relative 3D waypoints and a stop signal; a low-level controller executes the segment, and episodes end at stop or after $T = 128$ steps. Our reference baseline, VoLN-MLLM (Sec. 5), consists of a frozen DINOv3 ViT-B16 visual backbone, a lightweight adapter that aligns DINO features to the CLIP ViT-B/16 image-embedding space, and a frozen Vicuna-7B-v1.5 planning backbone (Zheng et al., 2023). Learned visual and proprioceptive projectors map the permitted task inputs to the Vicuna embedding dimension, and LoRA modules of rank 16 adapt its attention and feed-forward projections. A trajectory head predicts eight such waypoints, while a binary stopping head produces the stop signal. Experiments run on a single NVIDIA A100 GPU with 80 GB of GPU memory and 512 GB of system memory. All methods operate in closed loop and share this action interface and stopping criterion.

Baselines. We compare VoLN-MLLM with a random policy and three visual-goal (VG) variants of representative instruction-following architectures, replacing their language inputs with the VoLN interface. All learned methods receive the same observation history, goal views, proprioception, and semantic tokens retrieved from the frozen bank, and share the same visual encoder, waypoint action space, and training targets. **Random** samples feasible actions uniformly. **Seq2Seq-VG**, based on Anderson et al. (2018b), uses a recurrent encoder–decoder over visual–proprioceptive history, the goal representation, and retrieved semantic tokens. **CMA-VG**, based on Krantz et al. (2020), attends from the current visual state to history, goal, semantic tokens, and proprioception. **LAG-VG**, based on Liu et al. (2023), separately attends to observation history and semantic tokens before waypoint prediction.

Evaluation metrics. Following common practice (Krantz et al., 2020), we report Success Rate (SR, stopping inside the $\epsilon = 4\,\text{m}$ goal region), Oracle Success Rate (OSR, entering the success region at any point), Navigation Error (NE, the final Euclidean distance to the goal), normalized Dynamic Time Warping (nDTW) (Ilharco et al., 2019), and Success weighted by Path Length (SPL) (Anderson et al., 2018a):

$$\text{SPL} = \frac{1}{N} \sum_{i=1}^{N} S_i \cdot \frac{l_i}{\max(p_i, l_i)}\,, \tag{7}$$

where $S_i \in \{0, 1\}$ indicates whether episode i is successful, l_i denotes the shortest-path distance from the start position to the goal (computed offline for evaluation only), and p_i is the actual path length executed by the agent. NE is reported in meters and the remaining metrics as percentages. Results are averaged over five independent runs; across-run standard deviations range from 1.2–2.4 m for NE and 0.1–0.5 percentage points for the percentage metrics.

6.2 MAIN RESULTS

Table 1 compares VoLN-MLLM with the baselines on Validation-Seen and Test-Unseen across the Easy, Normal, and Hard subsets.

Overall performance. VoLN-MLLM yields the highest reported point estimate for every metric in Table 1. On Test-Unseen, its SR reaches 7.4%, 4.5%, and 1.8% across the three difficulty levels, compared with 2.3%, 1.2%, and 0.4% for the strongest baseline, LAG-VG. Performance declines with increasing route difficulty for all methods, but the relative advantage of VoLN-MLLM persists. Nevertheless, the low absolute SR, particularly on the Hard subset, shows that long-horizon visual-only navigation remains challenging.

Table 1: Results on Validation-Seen (VS) and Test-Unseen (TU) across difficulty levels.

Method	Split	NE/m↓			SR/%↑			OSR/%↑			nDTW/%↑			SPL/%↑		
		Easy	Normal	Hard	Easy	Normal	Hard	Easy	Normal	Hard	Easy	Normal	Hard	Easy	Normal	Hard
Random	VS	268.5	308.7	398.9	0.6	0.0	0.0	1.8	0.8	0.2	28.3	20.0	12.1	0.4	0.0	0.0
Seq2Seq-VG	VS	205.8	251.6	307.4	1.2	0.5	0.1	5.4	2.9	1.0	30.1	21.0	11.8	0.9	0.3	0.1
CMA-VG	VS	170.2	211.9	261.3	1.9	0.9	0.2	7.6	4.3	1.9	34.5	25.2	16.1	1.3	0.6	0.1
LAG-VG	VS	118.7	154.9	203.6	2.8	1.5	0.5	7.8	4.6	1.9	29.7	20.4	12.7	1.9	1.0	0.3
VoLN-MLLM	VS	**92.4**	**126.8**	**171.5**	**8.7**	**5.4**	**2.1**	**13.4**	**10.6**	**4.1**	**54.8**	**40.9**	**25.7**	**6.5**	**3.8**	**1.4**
Random	TU	270.1	310.4	395.2	0.4	0.0	0.0	1.4	0.6	0.2	30.1	22.7	15.1	0.3	0.0	0.0
Seq2Seq-VG	TU	208.6	254.8	309.9	1.0	0.4	0.1	4.8	2.5	0.9	28.9	21.4	13.0	0.7	0.3	0.0
CMA-VG	TU	174.5	216.8	266.1	1.6	0.8	0.2	6.5	3.9	1.7	33.2	26.4	18.5	1.1	0.6	0.1
LAG-VG	TU	122.4	158.3	206.7	2.3	1.2	0.4	6.4	3.8	1.7	28.1	20.5	14.0	1.5	0.7	0.2
VoLN-MLLM	TU	**97.1**	**131.4**	**176.8**	**7.4**	**4.5**	**1.8**	**14.6**	**10.1**	**4.5**	**53.1**	**41.2**	**28.0**	**5.7**	**3.2**	**1.3**

Trajectory quality and efficiency. Beyond terminal success, VoLN-MLLM has lower reported NE, indicating that its final positions are closer to the goal. Its higher reported nDTW corresponds to greater agreement between the executed and reference trajectories. These improvements are observed across both splits and all three path-length strata. VoLN-MLLM also records the highest SPL, reflecting stronger combined performance in navigation success and path efficiency.

6.3 Supplementary Analyses

In the ablation study, we additionally report cycle time (CT), measured from observation input to action output, and execution error rate (EER), defined as the percentage of planning cycles that exceed the time budget or produce invalid outputs. Results are averaged over five seeds, with CT standard deviations below 0.03 s.

Table 2: Test-Unseen ablations (CT in seconds).

Variant	**CT↓**	**EER↓**	**NE↓**	**SR↑**	**OSR↑**	**nDTW↑**	**SPL↑**
VoLN-MLLM	1.42	**0.5**	**119.0**	**5.7**	**11.8**	**45.8**	**4.3**
– No-Align	**1.36**	0.9	162.8	2.3	7.1	29.6	1.2
– No-LoRA	1.45	5.8	176.9	2.8	7.8	27.2	1.4
– CLIP-Input	1.98	1.5	158.6	2.9	8.2	30.9	1.6

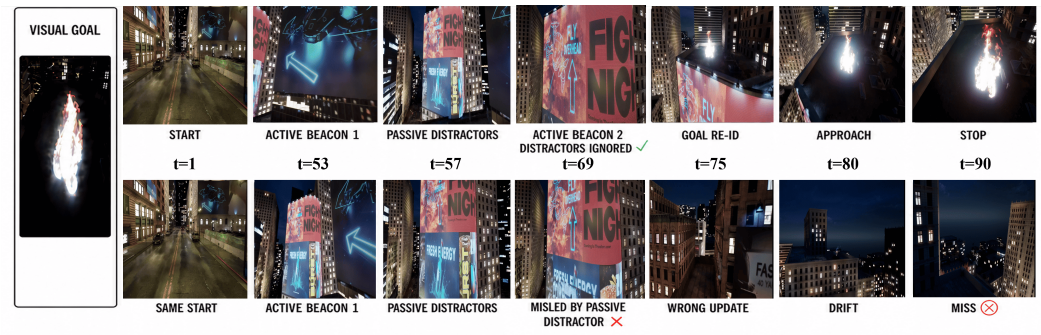


Figure 4: Success and failure cases in a nighttime urban scene. The upper row follows active beacons toward the goal; the lower row depicts a deviation near passive billboards.

Ablation studies. On Test-Unseen, we ablate three components. No-Align retains the dimensional projection but removes CLIP-teacher supervision. No-LoRA freezes the planner backbone

and removes the LoRA branch, while retaining the trained input projectors and prediction heads. CLIP-Input replaces the DINO backbone with CLIP image encoder. All variants use the same training split, seed pool, and evaluation budget. As shown in Table 2, No-Align causes the largest drop in SR ($5.7 \rightarrow 2.3$), while nDTW decreases from 45.8 to 29.6, highlighting the importance of compatibility between visual representations and the semantic space for effective grounding. Removing the LoRA branches yields the highest EER (5.8%) and the lowest nDTW (27.2), suggesting that planner adaptation improves output reliability and trajectory fitting. CLIP-Input increases the cycle time from 1.42 s to 1.98 s and reduces SR to 2.9%.

Success and failure cases. Figure 4 compares two nighttime urban paths at corresponding stages under a shared visual goal and initial context. The upper sequence passes active-beacon locations and approaches the goal, whereas the lower diverges near passive billboards and ends outside the goal region. Figure 8 in Appendix D provides additional examples across urban rooftops, forested terrain, daytime urban canyons, and open hills.

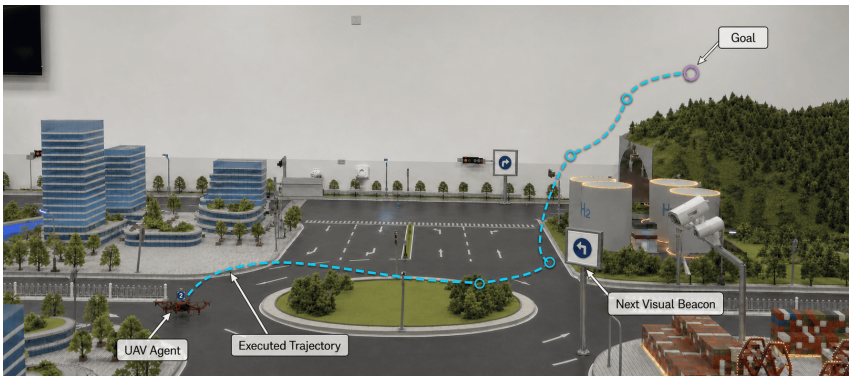


Figure 5: Physical testbed and a representative VoLN rollout.

Physical testbed demonstration. We conduct a preliminary physical demonstration of the VoLN task interface in a controlled indoor testbed. The testbed comprises a scaled urban scene with roads, a roundabout, building clusters, vegetated terrain, and miniature directional beacons (Fig. 5). At the navigation-policy level, the UAV receives goal views, onboard RGB observations, and proprioception, following the same input interface used in simulation. In one representative rollout, the UAV traverses the roundabout and changes direction near the beacon-marked junction toward the hilltop goal region. This demonstration shows how the VoLN task interface and closed-loop navigation pipeline can be instantiated on a controlled physical platform.

7 Conclusion

We introduce VoLN, a navigation paradigm for long-horizon, multi-stage visual–semantic reasoning in open environments. Given goal views and no separate language instructions, the agent interprets semantic beacons encountered en route, integrates observation history, and updates executable plans. VoLN-UAV instantiates this paradigm in aerial navigation, with 7,210 episodes across 17 simulated environments, continuous 3D control, and held-out scene evaluation. Its active and passive beacons provide a controlled setting for studying sustained cue selection, serving as an intermediate step between instruction-guided VLN and image-goal navigation.

VoLN-MLLM couples visual–semantic alignment with closed-loop waypoint and stopping prediction from goal views, recent observations, and proprioception. It achieves the highest reported point estimates among the evaluated baselines across both splits and all three difficulty levels. Test-Unseen success rates of 7.4%, 4.5%, and 1.8% for Easy, Normal, and Hard episodes indicate that reliable navigation over extended routes remains challenging. A controlled physical testbed demonstration further instantiates the task interface beyond simulation. Future work should improve integration of observation history, cue relevance assessment, and recovery from accumulated execution errors.

AI USE STATEMENT

Generative AI tools were used to refine the wording of research hypotheses proposed by the authors and to assist with translation. In addition, these tools were used for routine code revisions and LaTeX formatting, and to help improve the clarity of the presentation of research ideas developed by the authors. Generative AI tools were not used to design the research methodology or experiments, implement the proposed method, conduct qualitative or thematic analysis, or interpret the experimental results. The remaining required disclosure items were not applicable to this work. The authors verified the cited literature and supporting sources, tested AI-assisted code modifications, checked all reported results against the original experimental records, and reviewed all text processed with the assistance of generative AI. The method presented in this work was developed by the authors, the experiments were designed and conducted by the authors, the core method was implemented by the authors, and the related analyses were performed independently by the authors. The authors take full responsibility for the entire submission.

REFERENCES

Peter Anderson, Angel 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. arXiv preprint arXiv:1807.06757, 2018a.

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 (CVPR)*, pp. 3674–3683, 2018b. doi: 10.1109/CVPR.2018.00387.

Abrar Anwar, John Welsh, Joydeep Biswas, Soha Pouya, and Yan Chang. ReMEmbR: Building and reasoning over long-horizon spatio-temporal memory for robot navigation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2838–2845. IEEE, 2025. doi: 10.1109/ICRA55743.2025.11127706.

Luca Barsellotti, Lorenzo Bianchi, Nicola Messina, Fabio Carrara, Marcella Cornia, Lorenzo Baraldi, Fabrizio Falchi, and Rita Cucchiara. Talking to DINO: Bridging self-supervised vision backbones with language for open-vocabulary segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22025–22035, 2025. doi: 10.1109/ICCV51701.2025.02045.

Guillaume Bono, Leonid Antsfeld, Boris Chidlovskii, Philippe Weinzaepfel, and Christian Wolf. End-to-end (instance)-image goal navigation through correspondence as an emergent phenomenon. In *International Conference on Learning Representations (ICLR)*, 2024.

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*, pp. 6659–6676. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.338.

Wenzhe Cai, Siyuan Huang, Guangran Cheng, Yuxing Long, Peng Gao, Changyin Sun, and Hao Dong. Bridging zero-shot object navigation and foundation models through pixel-guided navigation skill. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5228–5234, 2024. doi: 10.1109/ICRA57147.2024.10610499.

Jing Cao, Nishanth Kumar, and Aidan Curtis. Follow the signs: Using textual cues and LLMs to guide efficient robot navigation. *arXiv preprint arXiv:2601.06652*, 2026.

Jiaqi Chen, Bingqian Lin, Ran Xu, Zhenhua Chai, Xiaodan Liang, and Kwan-Yee K. Wong. MapGPT: Map-guided prompting with adaptive path planning for vision-and-language navigation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9796–9810. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.529.

Daniel Choi, Angus Fung, Haitong Wang, and Aaron Hao Tan. Find everything: A general vision language model approach to multi-object search. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 19936–19943, 2025. doi: 10.1109/IROS60139.2025.11246991.

Yunpeng Gao, Chenhui Li, Zhongrui You, Junli Liu, Zhen Li, Pengan Chen, Qizhi Chen, Zhonghan Tang, Liansheng Wang, Penghui Yang, Yiwen Tang, Yuhang Tang, Shuai Liang, Songyi Zhu, Ziqin Xiong, Yifei Su, Xinyi Ye, Jianan Li, Yan Ding, Dong Wang, Zhigang Wang, Bin Zhao, and Xuelong Li. OpenFly: A comprehensive platform for aerial vision-language navigation. In *International Conference on Learning Representations (ICLR)*, 2026.

Wenxuan Guo, Xiuwei Xu, Hang Yin, Ziwei Wang, Jianjiang Feng, Jie Zhou, and Jiwen Lu. IGL-Nav: Incremental 3d gaussian localization for image-goal navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6808–6817, 2025. doi: 10.1109/ICCV51701.2025.00640.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

Gabriel Ilharco, Vihan Jain, Alexander Ku, Eugene Ie, and Jason Baldridge. General evaluation for instruction conditioned navigation using dynamic time warping. In *NeurIPS Visually Grounded Interaction and Language (ViGIL) Workshop*, 2019.

Elia Kaufmann, Leonard Bauersfeld, Antonio Loquercio, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Champion-level drone racing using deep reinforcement learning. *Nature*, 620(7976):982–987, 2023. doi: 10.1038/s41586-023-06419-4.

Chan Kim, Keonwoo Kim, Mintaek Oh, Hanbi Baek, Jiyang Lee, Donghwi Jung, Soojin Woo, Younkyung Woo, John Tucker, Roya Firoozi, Seung-Woo Seo, Mac Schwager, and Seong-Woo Kim. E2Map: Experience-and-emotion map for self-reflective robot navigation with language models. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 12811–12817, 2025. doi: 10.1109/ICRA55743.2025.11128669.

Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *Computer Vision – ECCV 2020*, volume 12373 of *Lecture Notes in Computer Science*, pp. 104–120. Springer, 2020. doi: 10.1007/978-3-030-58604-1_7.

Srivatsan Krishnan, Behzad Boroujerdian, William Fu, Aleksandra Faust, and Vijay Janapa Reddi. Air learning: A deep reinforcement learning gym for autonomous aerial robot visual navigation. *Machine Learning*, 110(9):2501–2540, 2021. doi: 10.1007/s10994-021-06006-6.

Xiaohan Lei, Min Wang, Wengang Zhou, and Houqiang Li. GaussNav: Gaussian splatting for visual navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(5):4108–4121, 2025. doi: 10.1109/TPAMI.2025.3538496.

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 (ICCV)*, pp. 15384–15394, 2023.

Jiabin Lou, Wenjun Wu, Shuhao Liao, and Rongye Shi. Air-M: A visual reality many-agent reinforcement learning platform for large-scale aerial unmanned system. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5598–5605. IEEE, 2023. doi: 10.1109/IROS55552.2023.10341405.

Jiabin Lou, Rong Ding, and Wenjun Wu. HHPSO: A heuristic hybrid particle swarm optimization path planner for quadcopters. *Drones*, 8(6):221, 2024. doi: 10.3390/drones8060221.

Jiabin Lou, Rongye Shi, Yuxin Lin, Qunbo Wang, and Wenjun Wu. TALKER: A task-activated language model based knowledge-extension reasoning system. *IEEE Robotics and Automation Letters*, 10(2):1026–1033, 2025. doi: 10.1109/LRA.2024.3511434.

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)*, pp. 2392–2399, 2025. doi: 10.1109/IROS 60139.2025.11246684.

Yiran Qin, Ao Sun, Yuze Hong, Benyou Wang, and Ruimao Zhang. NavigateDiff: Visual predictors are zero-shot navigation assistants. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 12002–12009. IEEE, 2025. doi: 10.1109/ICRA55743.2025.11127507.

Pranav Saxena, Nishant Raghuvanshi, and Neena Goveas. UAV-VLN: End-to-end vision language guided navigation for UAVs. In *2025 European Conference on Mobile Robots (ECMR)*, pp. 1–6, 2025. doi: 10.1109/ECMR65884.2025.11163198.

Hardik Shah, Jiaxu Xing, Nico Messikommer, Boyang Sun, Marc Pollefeys, and Davide Scaramuzza. ForesightNav: Learning scene imagination for efficient exploration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 5275–5284, 2025. doi: 10.1109/CVPRW67362.2025.00519.

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. In *International Conference on Learning Representations (ICLR)*, 2025.

Naoki Yokoyama, Sehoon Ha, Dhruv Batra, Jiuguang Wang, and Bernadette Bucher. VLFM: Vision-language frontier maps for zero-shot semantic navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 42–48, 2024. doi: 10.1109/ICRA57147.2024.10610712.

Jiazhao Zhang, Anqi Li, Yunpeng Qi, Minghan Li, Jiahang Liu, Shaoan Wang, Haoran Liu, Gengze Zhou, Yuze Wu, Xingxing Li, Yuxin Fan, Wenjun Li, Zhibo Chen, Fei Gao, Qi Wu, Zhizheng Zhang, and He Wang. Embodied navigation foundation model. In *International Conference on Learning Representations (ICLR)*, 2026.

Mike Zhang, Kaixian Qu, Vaishakh Patil, Cesar Cadena, and Marco Hutter. Tag map: A text-based map for spatial reasoning and navigation with large language models. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2120–2146. PMLR, 2025a.

Sixian Zhang, Xinyao Yu, Xinhang Song, Xiaohan Wang, and Shuqiang Jiang. Imagine before go: Self-supervised generative map for object goal navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16414–16425, 2024. doi: 10.1109/CVPR52733.2024.01553.

Weichen Zhang, Chen Gao, Shiquan Yu, Ruiying Peng, Baining Zhao, Qian Zhang, Jinqiang Cui, Xinlei Chen, and Yong Li. CityNavAgent: Aerial vision-and-language navigation with hierarchical semantic planning and global memory. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 31292–31309. Association for Computational Linguistics, 2025b. doi: 10.18653/v1/2025.acl-long.1511.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, pp. 46595–46623. Curran Associates, Inc., 2023. doi: 10.52202 /075280-2020.

Gengze Zhou, Yicong Hong, and Qi Wu. NavGPT: Explicit reasoning in vision-and-language navigation with large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(7):7641–7649, 2024. doi: 10.1609/aaai.v38i7.28597.

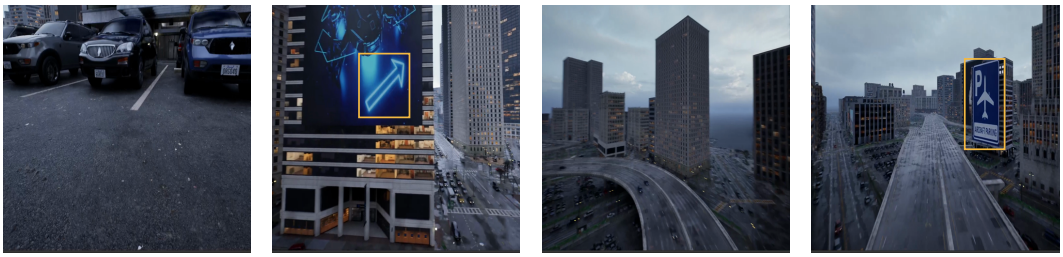
APPENDIX

A DATA AND BEACON CONSTRUCTION

A.1 TASK INTERFACE

Figure 6 shows the visual goal and successive observations from the urban episode used in Figure 1. Each episode provides a visual goal set $\mathcal{V}$ containing views captured near the destination. At step t, the agent receives an egocentric RGB observation o_t and proprioception p_t, including IMU measurements, altitude, velocity, and orientation. The policy conditions on the interaction history h_t and goal views to select an action a_t. Executing the action changes the UAV state and produces the next observation, completing the closed-loop interaction.

Semantic beacons are encountered within the RGB observations. Their interpretation is conditioned on the goal views and accumulated interaction history. The goal region $G(\mathcal{V})$, shown in the environment panel of Figure 1, defines the region used to evaluate successful stopping; it is not an additional policy input.



(a) Visual goal (b) Directional beacon (c) Spatial context (d) Semantic landmark

Figure 6: Elements of the VoLN paradigm illustrated by the urban episode in Figure 1. (a) A visual goal specifies the destination. (b)–(d) Successive onboard observations expose a directional beacon, spatial context, and a semantic landmark. Yellow boxes highlight example cues.

A.2 ENVIRONMENT COVERAGE

The benchmark contains 17 environments drawn from scenes adapted from AerialVLN (Liu et al., 2023) and OpenUAV (Wang et al., 2025), together with custom-built scenes, as described in Section 4. Figure 7 presents nine representative scene types. Natural environments include deserts, forests, and mountains. Built environments include urban canyons, nighttime urban scenes, campuses or parks, tunnels, industrial corridors, and ruins or open sites.

These environments vary in layout, visibility, altitude change, and landmark density. The collection covers both open outdoor areas and constrained passages such as tunnels and industrial corridors. Episodes within these environments are organized by reference-route length, with separate evaluation splits for seen environments and environments from a held-out scene source.

A.3 CONSTRUCTION PROTOCOL

The following details specify the benchmark construction protocol and reference implementation.

Episode construction and split protocol. Reference trajectories are recorded at a 2 s sampling interval. Their three-dimensional arc lengths determine the difficulty: Easy is below 300 m, Normal is from 300 m to below 450 m, and Hard is at least 450 m. The final three RGB observations form the visual goal set. Train, Validation-Seen, and Test-Unseen contain 5,047, 1,082, and 1,081 episodes, respectively. Validation-Seen uses disjoint trajectories from five environments within the 12-environment training pool; Test-Unseen uses five additional environments from a held-out scene source. The split validator checks episode and environment counts, trajectory overlap, and scene-source separation.



Figure 7: Representative VoLN-UAV environments.

Decision-point selection. Let $\mathbf{q}_i$ denote the position at reference step i. An interior step is a candidate decision point if the absolute change in horizontal heading between $\mathbf{q}_{i-1} \to \mathbf{q}_i$ and $\mathbf{q}_i \to \mathbf{q}_{i+1}$ is at least 0.45 rad (approximately $25.8°$), or if the absolute vertical displacement between $\mathbf{q}_{i-1}$ and $\mathbf{q}_{i+1}$ is at least 1 m. Candidates are scanned in temporal order, retaining a minimum separation of eight reference steps. Each route receives three, four, or five active beacons for Easy, Normal, or Hard episodes. The constructor first selects candidate decision points in route order. If fewer candidates are available, it fills the remaining positions using evenly spaced nonterminal reference indices, followed by unused indices if necessary. The terminal index is reserved for the target, and the selected beacon indices are stored with the episode.

Active-beacon placement. The simulator instantiates the stored episode-level beacon list. For a selected route index i, placement is anchored to reference step $\max(0, i-2)$ and uses that step's position and heading. Table 3 gives the offsets in the local horizontal frame. Forward jitter is sampled uniformly from $[-3, 3]$ m, and lateral offsets are sampled uniformly within the indicated intervals. Positive vertical offsets point downward in AirSim's north–east–down (NED) convention. For the BrushifyUrban scene, a fixed ground coordinate $z = -1$ m overrides the vertical offset. Route-sign yaw is offset by $90°$ from the anchor heading. Explicit asset tags take precedence over semantic labels; otherwise, directional labels select the corresponding sign, with local motion used as the fallback. A separate target asset is placed 40 m forward of the terminal reference pose, with zero lateral and vertical offset, a $180°$ yaw offset, and a scale factor of 1.6. Navigation success is evaluated relative to the reference endpoint.

Table 3: Active-beacon placement in the reference configuration. Offsets are in meters; forward positions include uniform jitter of ± 3 m.

Difficulty	Beacons	Forward	Lateral interval	Vertical (NED)
Easy	3	32	$[-5, -2]$	8
Normal	4	45	$[-7, -3]$	10
Hard	5	55	$[-10, -4]$	12

Passive beacons. The constructor generates 150 passive beacons per environment and reuses their world positions across episodes. Each placement samples an anchor from the environment's reference states, with forward, lateral, and vertical offsets drawn uniformly from $[-20, 40]$ m,

$[-30, 30]$ m, and $[-5, 5]$ m, respectively. Its yaw is the anchor yaw plus $180°$. Categories are cycled through the scene-compatible entries in the semantic bank. A deterministic seed derived from the environment identifier and construction seed fixes the layout. Route-local nearest indices are then attached for annotation without changing the beacon positions. Active-beacon jitter is likewise deterministic for each episode identifier and construction seed.

B Semantic Bank and Model Implementation

Semantic descriptors and text encoding. Table 4 specifies a 64-entry vocabulary covering beacon attributes, navigation cues, route geometry, scene context, landmarks, passages and obstacles, and spatial relations. Each descriptor is passed directly to the OpenCLIP tokenizer and the frozen CLIP ViT-B/16 text encoder with OpenAI pretrained weights. No prompt template is appended. The resulting 512-dimensional vectors are ℓ_2-normalized and computed once. At each step, the aligned current-image embedding is normalized and compared with all bank entries by cosine similarity; the top eight entries are retrieved in descending score order. The planner tokenizes each selected descriptor with the Vicuna tokenizer, averages its frozen input-token embeddings, and adds a learned embedding of the retrieval score.

Table 4: Semantic-bank descriptors grouped by semantic category.

Category	Descriptors
Beacon attributes	`beacon-blue`, `beacon-red`, `beacon-yellow`, `beacon-green`
Navigation cues	`road-sign`, `turn-left`, `turn-right`, `ascend`, `descend`, `go-straight`, `stop-sign`, `warning-sign`, `information-sign`, `directional-arrow`
Route geometry	`junction`, `t-junction`, `crossroads`, `fork`, `roundabout`, `road-bend`, `road-end`, `path-entrance`, `path-exit`, `branching-path`
Scene context	`urban-canyon`, `forest-trail`, `industrial-corridor`, `open-field`, `desert`, `mountain-slope`, `river-bank`, `campus`, `courtyard`, `construction-site`, `parking-lot`
Landmarks	`building-entrance`, `building-corner`, `rooftop`, `tower`, `bridge`, `archway`, `gate`, `statue`, `billboard`, `water-tank`
Passages and obstacles	`tunnel`, `tree`, `rock`, `wall`, `fence`, `overhead-obstacle`, `narrow-passage`, `open-passage`, `blocked-passage`, `steep-slope`, `drop-off`
Spatial relations	`passage-left-of-building`, `passage-right-of-building`, `passage-between-buildings`, `passage-under-bridge`, `space-above-rooftop`, `road-along-river`, `path-through-trees`, `beacon-at-junction`

Visual alignment and projectors. Images are resized to 224×224 pixels. The frozen DINOv3 ViT-B/16 backbone produces 768-dimensional features. The alignment adapter comprises a $768 \to 1024$ linear layer, GELU, a $1024 \to 512$ linear layer, LayerNorm, and output ℓ_2 normalization. It is trained against the frozen CLIP ViT-B/16 image encoder using cosine distillation. Both DINOv3 and the trained adapter are frozen during planner training. Table 5 specifies the remaining projections and heads, where d is the Vicuna hidden width. The visual projector is shared by goal and observation embeddings.

Table 5: Planner projections and prediction heads. Each arrow denotes a linear layer.

Module	Architecture
Visual projector	$512 \to d$, GELU, $d \to d$
State projector	$9 \to d$, GELU, $d \to d$
Confidence projector	$1 \to d$, Tanh, $d \to d$
Trajectory head	$d \to d$, GELU, $d \to 24$
Stop head	$d \to 1$, followed by sigmoid at inference

Temporal context and planner inputs. The history window contains the eight most recent RGB observations, including the current frame. At the beginning of an episode, the first observation is repeated on the left to fill the window. The nine-dimensional state vector contains body-frame linear velocity, angular velocity, and relative displacement. Five learned structural markers delimit the goal, history, semantic, state, and planning fields. The causal input sequence contains three goal tokens, eight history tokens, eight retrieved semantic tokens, one state token, and five markers, for a total of 25 tokens. The final planning marker attends to all preceding fields. Its hidden state produces eight body-frame 3D waypoints and a stop logit.

Planner adaptation. The planner uses Vicuna-7B-v1.5 with its pretrained weights frozen. LoRA is applied to the attention query, key, value, and output projections and the feed-forward gate, up, and down projections. The rank is 16, scaling parameter α is 32, and dropout is 0.05. The input projectors, structural markers, prediction heads, and LoRA parameters are optimized in the second training stage.

C Training and Evaluation Details

Optimization. Table 6 summarizes the two training stages. Both use AdamW with weight decay 10^{-4} and a constant learning rate. The alignment stage minimizes cosine distillation loss and selects the checkpoint with the lowest epoch-average training loss. The planning stage selects the checkpoint with the lowest Validation-Seen loss. Its objective is $\mathcal{L}_{\text{plan}} = \mathcal{L}_{\text{waypoint}} + 0.1\mathcal{L}_{\text{stop}}$, where waypoint loss is the mean absolute error over the predicted coordinates and stop loss is binary cross-entropy with logits. Stop supervision is positive when the recorded position is within 4 m of the reference endpoint. Training uses one NVIDIA A100 80 GB GPU. Frozen image embeddings are precomputed in batches of 64 for planner training; Vicuna uses bfloat16 and gradient checkpointing. The reference configuration sets the construction and training seed to 7 and initializes Python, NumPy, and PyTorch random generators.

Table 6: Training hyperparameters (effective planner batch size: 16).

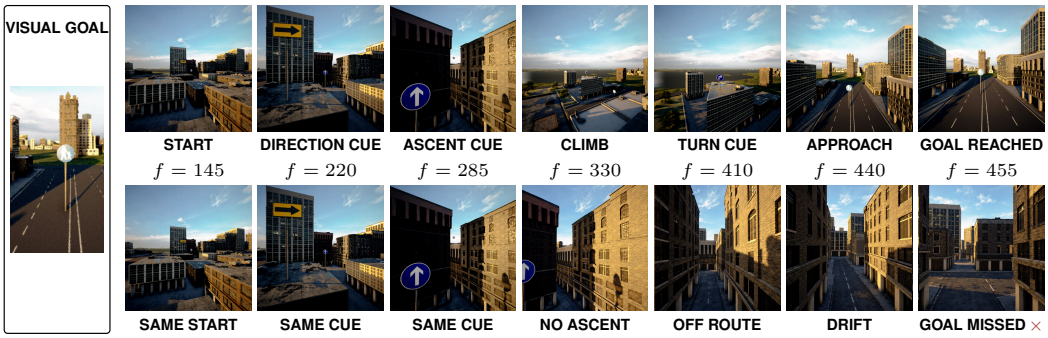
Parameter	Visual alignment	Planning
Optimizer	AdamW	AdamW
Learning rate	5×10^{-4}	3×10^{-4}
Weight decay	10^{-4}	10^{-4}
Batch size per iteration	32	1
Gradient accumulation steps	1	16
Training epochs	10	20
Maximum gradient norm	—	1.0

Stopping and execution. After checkpoint selection, the stop threshold is calibrated on Validation-Seen over $\{0.05, 0.10, \ldots, 0.95\}$. The selected threshold maximizes the F1 score of stop predictions; ties favor the value closest to 0.5, and any remaining tie retains the first value in ascending order. The threshold is saved with the checkpoint and used during evaluation. Each episode permits at most 128 decisions, with a 2 s planning budget per decision. SR requires an explicit stop within the 4 m goal region; OSR records whether the executed trajectory enters that region. Distances use all three spatial coordinates. SPL uses independently computed shortest-path lengths supplied with the episode metadata.

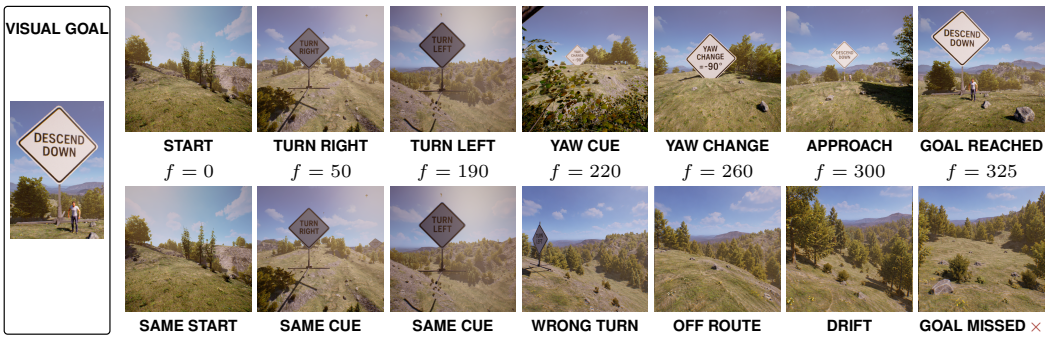
D Additional Navigation Examples

Figure 8 supplements the nighttime urban example in Figure 4 with four settings: urban rooftops, forested terrain, a daytime urban canyon, and open hills. The examples cover successive direction, turning, heading, and altitude cues.

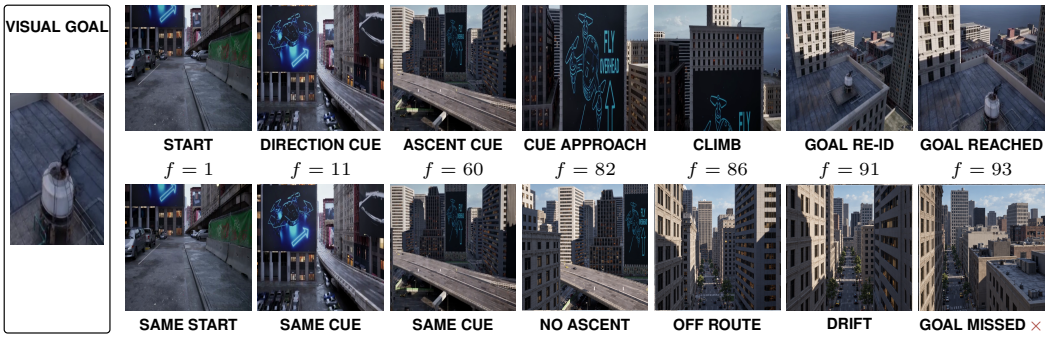
(a) Urban rooftops: sequential cue following



(b) Forested terrain: turn and yaw cues



(c) Daytime urban canyon: direction and altitude cues



(d) Open hills: heading and altitude changes



Figure 8: Additional navigation examples across four environments. Upper rows show recorded reference routes; lower rows illustrate corresponding deviations from the reference routes. Frame indices f refer to the upper rows.