AirNav: A Large-Scale UAV Vision-and-Language Navigation Dataset with Natural and Diverse Instructions

Anonymous ACL submission

Abstract

Existing UAV vision-and-language navigation (VLN) benchmarks rarely provide realistic aerial scenes, natural process-level instructions, and sufficient scale simultaneously, making it difficult to systematically train and evaluate UAV VLN agents under realistic settings. To address this, we propose **AirNav**, a large-scale benchmark built on real urban aerial data, comprising 137K navigation samples with natural and diverse instructions generated via a human–LLM collaborative pipeline with 10 user personas. We conduct a systematic evaluation of representative approaches on AirNav, ranging from traditional models to multimodal large language models (MLLMs), under unified metrics with open-source implementations. We further propose **AirVLN-R1**, trained via supervised fine-tuning (SFT) and reinforcement fine-tuning (RFT), achieving state-of-the-art performance with a 51.82% success rate on the test-unseen split. Real-world experiments on a physical UAV platform provide preliminary evidence of sim-to-real transferability, and our dataset and code are publicly available.[1]

1 Introduction

Unmanned Aerial Vehicle (UAV) Vision-and-Language Navigation (VLN) aims to enable UAVs to navigate complex urban environments by following natural language instructions, supporting applications such as emergency rescue, urban patrol, and infrastructure inspection. Compared with traditional rule-based approaches, language-guided navigation enables more flexible human–machine interaction and allows UAVs to autonomously accomplish diverse tasks across a wide range of environments. Unlike indoor or ground-level VLN, UAV VLN requires agents to reason over long-horizon aerial views, large-scale urban landmarks, and route instructions that often include intermediate cues and conditional behaviors. To facilitate research on this task, several benchmarks have been proposed, such as AerialVLN (Liu et al., 2023), CityNav (Lee et al., 2024), OpenUAV (Wang et al., 2024), and OpenFly (Gao et al., 2025). Recent advances in multimodal large language models (MLLMs) have opened new opportunities for language-grounded navigation; however, the lack of benchmarks that jointly provide realistic scenes, natural instructions, and sufficient scale remains a critical bottleneck for progress in UAV VLN.

First, realistic aerial perception requires real urban data with complex spatial structures and rich visual textures, while many existing datasets still rely on simulated environments. **Second**, language instructions in existing datasets often lack naturalness and diversity. Some datasets only provide target descriptions while ignoring critical intermediate cues during navigation, such as landmark references and conditional behaviors. Others include procedural descriptions but rely heavily on template-based generation, leading to monotonous language styles that do not reflect realistic usage. **Finally**, existing datasets are often limited in scale or language variation, hindering both the training of large-scale models and their ability to generalize across diverse environments and instruction styles.

To address these limitations, we propose **AirNav**, a large-scale UAV VLN benchmark constructed from real urban aerial data, with natural and diverse instructions produced through a human–LLM collaborative pipeline. AirNav contains 137K navigation samples and explicitly models instruction diversity through 10 user personas. We further conduct a unified evaluation of representative approaches, from traditional models to MLLMs, and propose **AirVLN-R1**, a navigation model optimized via a two-stage training paradigm combining supervised fine-tuning (SFT) and reinforcement fine-tuning (RFT). AirVLN-R1 achieves state-of-

[1] https://anonymous.4open.science/r/AirNav-FB4C

the-art performance on AirNav and shows preliminary transferability on a real-world UAV platform.

The main contributions of this work are summarized as follows:

1. We introduce AirNav, a large-scale UAV VLN benchmark built upon real urban aerial data, featuring 137K samples with natural and diverse instructions generated via a human–LLM collaborative pipeline with 10 user personas.

2. A comprehensive evaluation of representative approaches—from traditional models to multimodal large language models (MLLMs)—is conducted under unified metrics, with open-source implementations released to facilitate further research.

3. We propose AirVLN-R1, a navigation model combining SFT and RFT, achieving state-of-the-art performance on the AirNav benchmark, including a 51.82% success rate on the test-unseen split.

4. We deploy AirVLN-R1 on a physical UAV platform and conduct real-world navigation experiments across indoor and outdoor scenarios, providing initial evidence for practical deployment and sim-to-real transfer.

2 Related Work

Several benchmark datasets have been proposed for UAV VLN, as summarized in Table 1. These benchmarks exhibit different research emphases. Some datasets are constructed from real aerial imagery, rather than simulation-based virtual environments, offering higher visual realism and greater scene diversity. Others introduce explicit sub-goals, making them well suited for analyzing the alignment between language understanding and action decision-making. However, these datasets typically struggle to simultaneously offer real-data-based environments, natural instructions with complete navigation processes, and sufficient data scale for comprehensive training and evaluation. Detailed analyses are provided in Appendix A.

3 AirNav benchmark

3.1 Task Definition

The UAV VLN task guides a UAV to complete navigation missions in environments using language instructions. Starting from an initial position, the agent interacts with the environment over multiple steps of "Perception-Decision-Execution", during which it repeatedly predicts a series of action sequences that guide the UAV to the target.

We formulate this task as a partially observable sequential decision-making problem. At step t, the agent receives the multimodal observation:

$$O_{\leq t} = \{v_1, \ldots, v_t;\ S_t;\ A_{1:t-1};\ L\},$$

where v_i denotes the first-person image captured by the UAV at step i, S_t represents the current UAV state including its spatial position and heading angle, $A_{1:t-1}$ is the sequence of actions executed from the start up to the previous step, and L denotes the instruction describing the target and providing path-related cues.

The agent is required to learn a policy function π that, at each step t, generates an action sequence $\hat{A}_t$ conditioned on the accumulated observation $O_{\leq t}$:

$$\hat{A}_t = \pi(O_{\leq t}), \quad \hat{A}_t = \{a_t^{(1)}, a_t^{(2)}, \ldots, a_t^{(k)}\},$$

where $a_t^{(i)}$ denotes the i-th action in the sequence, and k is the number of actions predicted at the current step. The model can output a variable-length sequence of discrete actions at each step, with possible actions including **move forward**, **turn left**, **turn right**, and **stop**.

Success Criteria. The task is considered successful if, after the UAV has completed the navigation, the Euclidean distance between its final position and the target location is smaller than a predefined threshold (e.g., 20 meters).

Metrics. To comprehensively evaluate navigation accuracy and path efficiency, we follow standard VLN evaluation protocols (e.g., SOON (Zhu et al., 2021), CityNav (Lee et al., 2024)) and adopt the following metrics: Navigation Error (NE), Success Rate (SR), Oracle Success Rate (OSR), and Success weighted by Path Length (SPL). Detailed definitions are provided in Appendix B.

3.2 Benchmark Construction

We propose a four-step pipeline for benchmark construction, as shown in Fig. 1.

Data Sources and Environment AirNav is built upon the SensatUrban (Hu et al., 2022) and CityRefer (Miyanishi et al., 2023) datasets. SensatUrban provides high-density 3D point cloud data with rich geographic structures, covering two cities, Cambridge and Birmingham. CityRefer supplements natural language descriptions for objects appearing in the SensatUrban. The environment is constructed based on the CityFlight (Lee et al., 2024),

Dataset	Data Source	Action Space	Dataset Size	Sub-goals	Instruction Naturalness	Vocabulary Size
LANI (Misra et al., 2018)	Virtual	2 DoF	6,000	Yes	Medium	2.3K
AVDN (Fan et al., 2022)	Real data	3 DoF	3,064	Yes	Medium	3.3K
AerialVLN (Liu et al., 2023)	Virtual	4 DoF	8,446	Yes	Medium	4.5K
CityNav (Lee et al., 2024)	Real data	4 DoF	32,637	No	N/A	6.4K
OpenUAV (Wang et al., 2024)	Virtual	6 DoF	12,149	No	N/A	10.8K
OpenFly (Gao et al., 2025)	Virtual + Real data	4 DoF	100k	Yes	Medium	15.6K
AirNav (Ours)	Real data	4 DoF	137k	Yes	High	20.3K

Note. Here, "Real data" indicates that the benchmark is constructed from real aerial data, rather than synthetic or game-engine-based environments.

Table 1: Comparison of representative UAV VLN benchmarks across data, scale, and instructions.

which aligns the SensatUrban data with OpenStreetMap. CityFlight forms an interactive flight environment and provides interfaces for accessing various types of information, such as environmental images and object coordinates. SensatUrban, CityRefer, and CityFlight are all released under the MIT License.

Step 1: Start and Target Selection. The start point is randomly sampled from the map as a feasible coordinate to initialize the navigation episode. A geographical object with well-defined spatial boundaries is selected as the navigation target. Using an MLLM, a natural language description of the target is generated from the endpoint's perspective, and samples with ambiguous or confusing target descriptions are filtered out.

Step 2: Landmark Planning. Given the selected start and target locations, the MLLM is prompted to identify representative geographic objects between them and generate corresponding descriptions as intermediate landmarks. To preserve perceptual continuity along the route, we impose a maximum distance constraint between consecutive landmarks and discard samples in which landmarks are overly sparse. In addition, we perform semantic refinement for landmarks. For each landmark, the model further verifies the factual correctness of its description and rewrites it to ensure semantic clarity and disambiguation.

Step 3: Trajectory Synthesis. For each pair of consecutive nodes, such as the start point and the first landmark or two adjacent landmarks, we apply a look-ahead strategy (Liu et al., 2023) to generate an executable action sequence for the corresponding path segment. All segment-level action sequences are then concatenated to form a complete trajectory that spans from the start to the target.

Step 4: Instruction Generation. The trajectory, map, and the spatial positions and semantic descriptions of all nodes are provided as inputs to GPT-4o, which generates navigation instructions covering target descriptions, path guidance, spatial relations, and trigger conditions. To model linguistic realism and diversity, following principles from User-Centered Design (Norman, 2013; Pruitt and Grudin, 2003) and sociolinguistic studies on linguistic variation (Labov, 1973; Tagliamonte, 2011), we construct 10 representative user personas (see Appendix C) based on age group, social role, and expression preferences, covering typical urban navigation scenarios and diverse language styles. During instruction generation, persona-specific settings are incorporated to guide GPT-4o to produce navigation instructions with diverse language styles and variations in expression. To further improve linguistic naturalness, human-authored real navigation instructions are included as few-shot examples within the prompt.

Quality Control. To ensure data quality at scale, we adopt an iterative quality control process throughout the pipeline. After each round of generation, we randomly sample examples and manually inspect outputs across all stages, including the validity of start and target points, the clarity of target descriptions, the spatial consistency of landmark annotations, and the continuity of synthesized trajectories. Identified issues are categorized by error type (e.g., ambiguous landmarks, visually indistinguishable targets, excessive gaps between consecutive landmarks), and the corresponding generation prompts or filtering rules are updated accordingly. The data is then regenerated under the revised settings. This inspect-revise cycle is repeated over multiple iterations until the sampled error rate stabilizes at a low level, after which we proceed to large-scale generation.

3.3 Dataset and Instruction Analysis

Dataset Splits. Following evaluation protocols in prior UAV VLN benchmarks (Liu et al., 2023; Lee et al., 2024), we split the AirNav dataset into four subsets: Train, Validation Seen (val-seen), Validation Unseen (val-unseen), and Test Unseen (test-

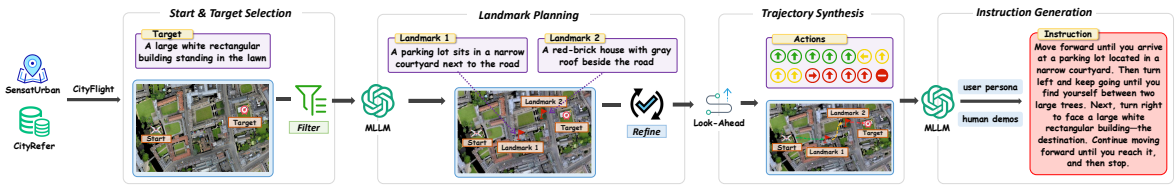


Figure 1: Overview of the AirNav Benchmark Construction Pipeline.

unseen). The val-seen split shares the same environments as Train and is used to evaluate model performance in known scenes, while val-unseen and test-unseen are sampled from novel environments to assess generalization under unseen conditions. Fig. 2 (a) summarizes key statistics for each subset.

Distance Distribution and Task Difficulty. We categorize navigation paths into three difficulty levels based on path length percentiles of the training set: *Easy* ($<$ 135m), *Medium* (135–230m), and *Hard* ($\geq$ 230m). Fig. 2 (b) presents the distance distributions for different difficulty categories over the entire evaluation set.

Number of Intermediate Landmarks. As shown in Fig. 2 (c), most trajectories contain between 2 and 6 intermediate landmarks, with 4 to 5 being the most prevalent, supporting step-wise modeling of the navigation process and spatial-semantic alignment between instructions and actions.

Instruction Length and Vocabulary Statistics. As shown in Fig. 2 (d), instruction lengths span a wide range, with a peak around 100 words. The dataset includes both concise instructions with compact structures and longer instructions that provide detailed descriptions of intermediate operations, reflecting the coexistence of different information densities and narrative granularities. In addition, AirNav exhibits a vocabulary size of 20.3k, which is significantly larger than that of existing UAV VLN datasets, indicating higher linguistic diversity and reduced repetition in instruction expressions.

Persona-conditioned Instruction Characteristics. AirNav explicitly models instruction diversity through the introduction of user personas. This design captures systematic differences among user groups in both instruction length and information organization, providing a more comprehensive representation of human instruction behaviors across varying information densities and narrative granularities. As illustrated in Figure 2 (e), distinct personas show clear separation in the median values and distribution ranges of instruction length. For example, Retired Adults tend to produce longer and more explanatory instructions, whereas University Students or Advanced Navigation Users generally prefer concise expressions. In Appendix D, we further conduct a case-study analysis of instructions generated under different personas.

Instruction Naturalness Analysis. To quantitatively evaluate instruction naturalness, we conduct an LLM-based automated assessment, sampling 2,000 instructions per dataset and scoring them on a 5-point scale using GPT-4o; evaluation details and the scoring prompt are provided in Appendix E. To validate the reliability of this evaluation, we further conduct a human annotation study, achieving a Krippendorff's α of 0.70 (inter-annotator agreement) and a Spearman's ρ of 0.74 (human–LLM correlation), confirming strong consistency with human judgments (see Appendix F). As shown in Figure 2 (f), AirNav achieves the highest naturalness score (3.75), significantly outperforming all other benchmarks, indicating that its instructions more closely resemble natural language requests from real users rather than templated formulations. A case-study comparison of instructions from different datasets is provided in Appendix G.

4 AirVLN-R1 Model

4.1 Overall Architecture

We model the UAV VLN task as a multi-step "Perception-Decision-Execution" loop. As shown in Fig. 3, at each step, the model receives multimodal input and predicts a sequence of actions to control the UAV's movement, continuing until the output is **stop** or the maximum number of steps is reached. To further optimize the policy, we adopt a dedicated two-stage training strategy.

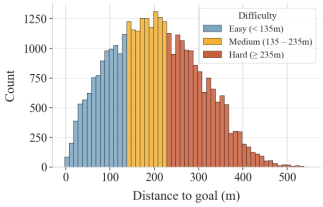
4.2 Input, Output and Prompt Design

4.2.1 Input

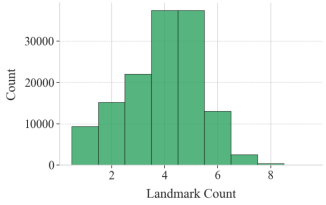
At each step, the AirVLN-R1 model processes multimodal inputs, which include both textual and vi-

Split	Desc.	Scene	Easy	Medium	Hard
Train	102,968	24	–	–	–
Val Seen	7,657	24	2,486	2,447	2,724
Val Unseen	9,928	4	2,474	3,453	4,001
Test Unseen	16,787	6	4,291	6,387	6,109

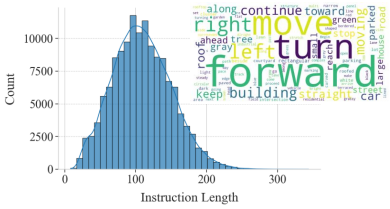
(a) Dataset Split Sets



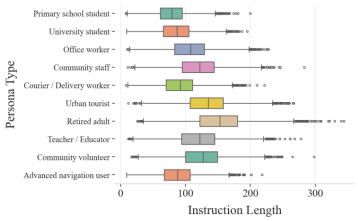
(b) Distance to goal



(c) Landmark Count



(d) Instruction Length



(e) Instruction Length by Persona

Dataset	Naturalness Score
LANI	3.00
AVDN	3.47
AerialVLN	2.59
OpenFly	3.12
AirNav (Ours)	**3.75**

(f) Naturalness Score

Figure 2: Dataset Analysis and Instruction Naturalness of AirNav.

sual information.

For the textual input, it consists of the following three components: (1) **Instruction**: A natural language description of the target location and path clues; (2) **Current State**: The UAV's coordinates and heading angle; (3) **Historical Action Sequence**: The sequence of actions already executed by the UAV from the starting point to the current step.

The visual input provides the perception of the surrounding environment, including: (1) **Current View Image**: An image captured from the UAV's first-person perspective at the current step; (2) **Historical View Images**: A set of key images selected from historical observations to construct a visual memory. To control the input size and enhance information efficiency, we adopt a **Progressive Interval Sampling** strategy for visual memory construction. See Section 4.3 for details.

4.2.2 Output

The model outputs a variable-length sequence of up to 8 discrete actions, representing the actions required to proceed from the current state.

4.2.3 Prompt Design

We design a structured prompt template specifying the task role, input structure, output requirements, and action space definition, with the complete template provided in Appendix H.

4.3 Historical View Image Selection

Directly incorporating the full sequence of historical view images introduces substantial input redundancy and computational overhead. To address this issue, we propose a lightweight historical view image sampling mechanism—**Progressive Interval Sampling**, which preserves dense observations from recent steps while sparsely sampling distant ones, thereby reducing the historical input size without losing critical contextual information.

At step t, we select at most N historical view images from the past observations. Sampling starts from the most recent view and proceeds backward with an interval that grows linearly over time. Specifically, the offset of the i-th selected historical view is defined recursively as:

$$s_i = s_{i-1} + i, \quad s_{-1} = 1, \quad i = 0, 1, \ldots, N-1.$$

Based on the offsets $\{s_i\}$, the visual memory at step t is constructed as:

$$\mathcal{H}_t = \{v_{t-s_i} \mid 0 \le i \le N-1,\ t - s_i \ge 1\},$$

where v_{t-s_i} represents the view image captured at step $t - s_i$.

4.4 Training Paradigm

To enhance the model's perceptual understanding and decision–making capability, AirVLN-R1 adopts a two-stage training paradigm inspired by DeepSeek-R1 (DeepSeek-AI et al., 2025), consisting of SFT followed by RFT.

4.4.1 Supervised Fine-Tuning

In the first stage, we fine-tune the model end-to-end on the training set following the prompt format defined in Section 4.2, guiding the model to build a mapping from multimodal inputs to executable action sequences via next-token prediction with cross-entropy loss.

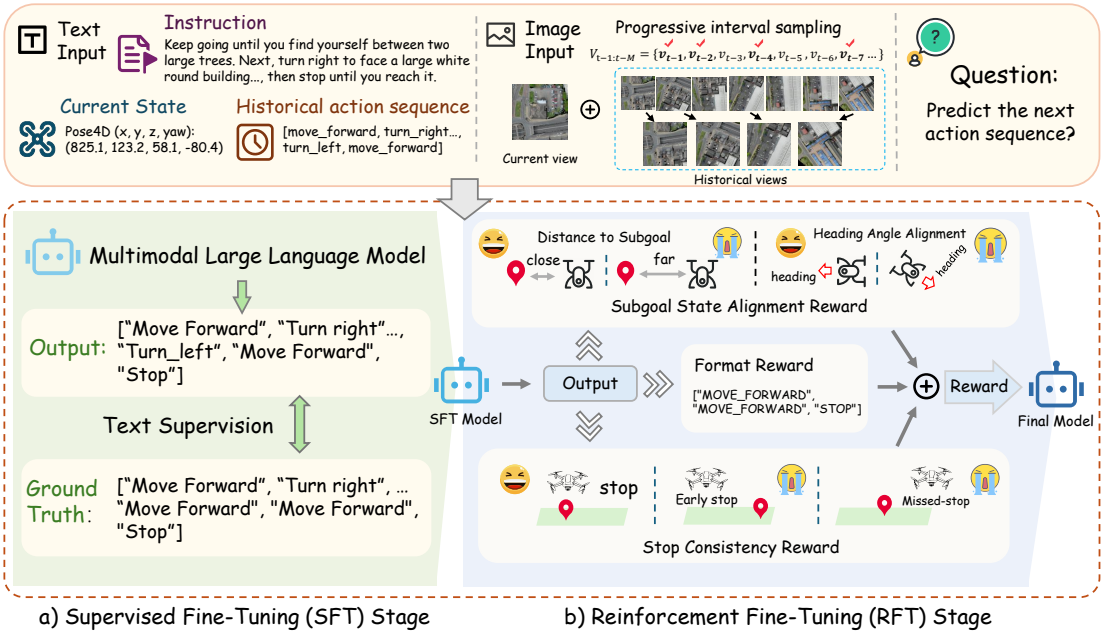


Figure 3: Overview of the AirVLN-R1 architecture. The model receives multimodal inputs and predicts an action sequence to control the UAV. A two-stage training paradigm is used to enhance performance.

4.4.2 Reinforcement Fine-Tuning

From a decision-making perspective, UAV VLN requires the agent to (i) make consistent progress toward intermediate subgoals, (ii) reliably determine when to terminate the episode, and (iii) generate valid outputs. To explicitly capture these requirements, we design a multi-objective reward function, where each reward component corresponds to one of the above decision objectives. We use Group Relative Policy Optimization (GRPO) to optimize the policy.

1. Subgoal State Alignment Reward The UAV state at each intermediate subgoal of the trajectory represents a reasonable position and heading angle achieved by executing a sequence of expert actions. We aim for the model to approach these subgoal states through its own predicted action sequences. To this end, we design two rewards reflecting position proximity and heading angle alignment.

(1) Distance-to-Subgoal This reward encourages the UAV to reduce its distance to the subgoal by executing the predicted action sequence at step t, resulting in a closer state at step $t+1$. The reward is defined as:

$$r_{\text{dis}} = \max\left(\frac{d_t - d_{t+1}}{d_t + \epsilon}, 0\right),$$

where d_t denotes the Euclidean distance between the UAV and the subgoal before executing the predicted action sequence at step t, and d_{t+1} denotes the distance after the execution.

(2) Heading Angle Alignment This reward measures whether the UAV's heading angle after executing the action sequence is aligned with the subgoal's heading angle. Let Δ_{yaw} denote the angular difference between the two headings, normalized to $(-180°, 180°]$. The reward is defined as:

$$r_{\text{yaw}} = \max\left(1 - \frac{|\Delta_{\text{yaw}}|}{\tau_{\text{yaw}}}, 0\right),$$

where τ_{yaw} controls the tolerance range (e.g., $60°$).

2. Stop Consistency Reward. An incorrect **stop** decision—whether premature or missed—directly causes task failure. To mitigate such errors, we define the following reward mechanism: the reward is α if both the predicted and ground-truth action sequences end with **stop**; the reward is β if neither the predicted nor the ground-truth sequence ends with **stop**; otherwise, the reward is 0.

3. Format Reward. To encourage the model to produce outputs with correct structure and valid syntax, we assign a constant reward $r_{\text{format}} = \gamma$ if the output is well-formed; otherwise, we set $r_{\text{format}} = 0$.

Overall Reward Function Definition. The final overall reward is obtained by linearly combining

all the aforementioned reward components:

$$r_{all} = \lambda_1 \cdot r_{\text{dis}} + \lambda_2 \cdot r_{\text{yaw}} + r_{\text{stop}} + r_{\text{format}}.$$

5 Experiments

5.1 Experimental Settings

5.1.1 Baseline Models

We select several representative baseline models, covering traditional methods, zero-shot MLLMs, and recent navigation models. Detailed information about all baseline models can be found in Appendix I.

5.1.2 Implementation Details

AirVLN-R1 is built upon Qwen2.5-VL-7B and trained on an 8×A100 GPU server. The hyperparameter settings for both training stages are provided in Appendix J. To ensure fair comparison, all MLLM baselines are provided with the same inputs as AirVLN-R1, including the navigation instruction, current UAV state, historical action sequence, current view image, and historical view images selected by the same progressive interval sampling strategy, as well as the same prompts. The temperature settings and other hyperparameters for all MLLM baselines are available in our open-source repository.

5.2 Main Results

The comparative results on AirNav are shown in Table 2. Traditional methods perform poorly, with Seq2Seq and CMA achieving SR below 2% and 6% respectively, failing to handle the complexity of UAV VLN. Among zero-shot MLLMs, LLaMA-3.2-11B-Vision yields similarly low SR, while GPT-4o performs better but remains limited on unseen scenarios. The Qwen series achieves stronger results with SR scaling consistently with model size, and Qwen3-VL-235B-A22B reaches the highest SR among all zero-shot baselines.

π_0, fine-tuned on AirNav data, achieves only limited performance (SR of 6.29% on val-seen), suggesting that Vision-Language-Action (VLA) models designed for robot manipulation scenarios do not transfer well to UAV VLN. Uni-NaVid, designed specifically for embodied navigation and fine-tuned on AirNav data, reaches an SR of 20.56% on val-seen and 15.89% on test-unseen, reflecting the advantage of navigation-oriented model design.

AirVLN-R1 achieves the best performance across all metrics in the AirNav Benchmark, with an SR of 51.82% on test-unseen. Notably, AirVLN-R1, built upon Qwen2.5-VL-7B, achieves superior performance compared to larger-scale models such as Qwen2.5-VL-32B and Qwen3-VL-235B-A22B, underscoring the effectiveness of task-specific supervision and RFT.

5.3 Persona-Level Analysis

We evaluate the SR of all models across all 10 user personas on the test-unseen split; complete results are provided in Appendix L. We observe that persona is associated with differences in SR. Among all personas, Teacher/Educator and Advanced Navigation User instructions yield the top two SR values: the former benefit from well-structured, step-by-step organization that facilitates progressive subgoal alignment, while the latter are concise and goal-oriented with minimal linguistic noise, making key navigation cues easy to extract. Conversely, Retired Adult instructions tend to be verbose with repeated contextual descriptions that increase the difficulty of extracting critical navigation cues, while Primary School Student instructions, though concise, lack sufficient spatial detail; both personas consequently achieve comparatively lower SR.

5.4 Ablation Study

Training Paradigm. We compare three paradigms: SFT-only, RFT-only, and SFT+RFT, as shown in Table 2. SFT-only achieves competitive performance on val-seen (SR = 44.27%), but leaves room for further improvement on val-unseen and test-unseen splits, indicating that trajectory imitation alone does not fully generalize. RFT-only struggles due to highly sparse reward signals without SFT initialization, converging to suboptimal policies. SFT+RFT achieves the best and most stable results across all splits, demonstrating that the two stages are complementary.

Historical View Image Selection. We compare four strategies for incorporating historical observations: No-History, Last-K, Uniform-K, and our Progressive Interval Sampling. Progressive Interval Sampling achieves the best performance (SR = 51.82% on test-unseen), outperforming Uniform-K (49.83%) by better balancing fine-grained short-term perception with long-range contextual modeling. Full results are provided in Appendix M.1.

Reward Component Analysis. We ablate each component of the RFT reward function individu-

Method	Validation Seen				Validation Unseen				Test Unseen			
	NE↓	SR↑	OSR↑	SPL↑	NE↓	SR↑	OSR↑	SPL↑	NE↓	SR↑	OSR↑	SPL↑
Random	222.0	0.76	5.68	0.67	223.1	0.76	4.73	0.68	217.2	0.77	5.57	0.69
Seq2Seq*	321.2	1.65	9.48	1.44	343.5	0.92	9.92	0.73	334.5	1.32	10.69	1.13
CMA*	184.7	5.21	15.84	4.80	201.1	3.88	16.07	3.47	189.4	4.63	17.48	4.16
Qwen2.5-VL-7B	182.1	1.96	2.36	1.82	192.5	1.59	1.76	1.41	184.2	1.70	1.95	1.53
Qwen2.5-VL-32B	160.9	3.25	3.66	2.94	170.6	2.68	2.98	2.38	162.4	2.99	3.25	2.68
Qwen3-VL-235B-A22B	157.6	5.58	9.23	5.18	168.5	5.14	8.42	4.63	155.6	5.21	8.44	4.75
LLaMA-3.2-11B-Vision	182.3	1.02	4.85	0.95	193.8	1.57	4.51	1.41	185.5	1.31	4.28	1.12
GPT-4o	154.0	4.71	8.68	4.22	164.3	4.16	7.18	3.73	156.2	4.56	7.88	4.13
π_0*	144.2	6.29	14.27	5.40	163.6	4.47	13.37	3.90	147.5	5.28	13.93	4.60
Uni-NaVid*	112.3	20.56	38.02	18.45	129.3	16.60	33.45	14.71	131.2	15.89	34.25	14.18
Qwen2.5-VL-7B SFT-only*	45.4	44.27	54.96	43.04	48.7	40.98	52.57	39.89	47.5	40.20	53.21	39.15
Qwen2.5-VL-7B RFT-only*	165.1	2.38	4.91	2.14	173.2	2.06	3.90	1.82	164.0	2.41	4.57	2.13
AirVLN-R1* (Ours)	**39.7**	**51.76**	**61.45**	**50.61**	**40.8**	**51.32**	**61.78**	**50.11**	**39.8**	**51.82**	**62.73**	**50.66**

Table 2: Comparison of model performance across evaluation scenarios. * denotes models fine-tuned on AirNav.

ally. Removing the Subgoal State Alignment reward causes the most significant degradation (SR: 51.82% → 46.00%), confirming its central role in guiding path planning. Removing the Stop Consistency reward also leads to a notable drop (SR: 51.82% → 47.65%), highlighting the importance of explicit termination supervision. The Format reward contributes to training stability with a smaller but consistent effect. Detailed results are in Appendix M.2.

Sensitivity Analysis of Reward Weights. We conduct a 3×3 grid search over the key weighting parameters λ_1 and λ_2. Performance remains stable across all tested configurations, with a maximum SR fluctuation of within 1.5 percentage points, indicating that the method is robust to the precise choice of reward weights. See Appendix M.3 for the full sensitivity results.

5.5 Real World Experiments

5.5.1 Setup

To evaluate the performance and deployment capability of AirVLN-R1 in real-world environments, we conduct a series of physical tests without any additional fine-tuning. The experiments cover two typical settings—indoor and outdoor—each containing 100 navigation tasks with varying levels of path complexity. Examples of real-world tasks, along with the details of the experimental setup, are provided in Appendix K.

5.5.2 Results

Quantitative results and overall comparison. We evaluate different methods in real-world test, with detailed results provided in Appendix N. Traditional methods nearly fail to complete any real-world tasks, with Seq2Seq and CMA succeeding on only 1 and 2 out of 200 tasks respectively. In contrast, general MLLMs under zero-shot settings can only solve a very small number of tasks. The stronger model GPT-4o shows a noticeable improvement, yet its overall SR remains limited. Compared with all baselines, AirVLN-R1 reaches SR= $53/200$ and achieves the lowest NE (70.6), maintaining a consistent relative advantage in the real-world environment that aligns with the simulation evaluation.

Resource Cost and Inference Efficiency. Traditional baselines exhibit low computational overhead but completely fail in real-world navigation, while large-scale MLLMs incur substantial latency and hardware requirements, making real-time deployment challenging. In contrast, AirVLN-R1 achieves the best real-world SR with moderate inference latency and GPU memory consumption, representing a more balanced and deployable solution for UAV VLN. A detailed comparison of computational cost and efficiency is provided in Appendix O.

6 Conclusion

This paper presents AirNav, a large-scale UAV VLN benchmark constructed from real urban aerial data, featuring 137K navigation samples with natural and diverse language instructions, along with a comprehensive evaluation of representative approaches under unified metrics and open-source implementations. We further propose AirVLN-R1, a navigation model combining SFT and RFT that achieves state-of-the-art performance on the AirNav benchmark. Real-world experiments validate the feasibility and consistency of sim-to-real transfer on physical UAV platforms. We hope AirNav provides a realistic and comprehensive evaluation foundation for future research on UAV VLN.

7 Limitations

7.1 Limited Data Sources and Scenario Coverage

AirNav is primarily constructed based on SensatUrban and CityRefer. While these data sources provide high-fidelity urban scenes, their geographic coverage, urban styles, and infrastructure layouts are inherently constrained by the regions and annotation schemes of the existing datasets. As a result, the generalization ability of models across different cities, countries, and seasonal conditions remains to be validated with more diverse data. In addition, aerial imagery often exhibits relatively fixed viewpoints and altitude distributions, which may not fully capture the more complex altitude variations, occlusion patterns, and extreme lighting conditions encountered in real-world UAV VLN tasks.

7.2 Gap Between Discrete Action Modeling and Real Flight Control

We adopt a discrete action set and allow the model to output an action sequence of up to eight steps at each step. This design facilitates stable training and fair comparison across methods, but it cannot fully represent the fine-grained motions and dynamic constraints involved in real UAV continuous control. In long-horizon navigation or tasks requiring precise target approach, discrete actions may introduce trajectory approximation errors, thereby limiting the policy's performance in complex maneuvering scenarios.

7.3 Gap Between Simulation Evaluation and Real-World Deployment

Although we conduct preliminary real-world flight experiments across 200 navigation tasks, the environmental diversity of the real-world evaluations remains limited. Moreover, differences between simulation and real UAV deployment persist in terms of perception noise, viewpoint variations, and control uncertainty. Consequently, performance improvements observed on the benchmark do not necessarily translate directly into stable gains in real-world settings. Further reducing the sim-to-real gap remains an important direction for future work.

7.4 Reproducibility of Closed-Source Model Results

Our evaluation includes closed-source models such as GPT-4o. As these models are subject to API access restrictions and potential version updates over time, exact reproduction of the reported results may be limited.

8 Ethical Considerations

8.1 Privacy Risks in Urban Aerial Data

AirNav relies on real-world urban scene data, and aerial imagery may implicitly contain sensitive areas or information related to human activities. Although this work uses datasets intended for research purposes, the release of the benchmark and models should be handled with care to avoid unintended use in unauthorized area localization, surveillance, or privacy inference.

8.2 Reliability in Safety-Critical Applications

UAV VLN is a typical safety-critical application. Even when a model performs well on benchmark evaluations, real-world deployment may still suffer from perception errors or instruction misinterpretation, potentially leading to collisions or entry into hazardous areas. Therefore, the methods proposed in this work should not be regarded as a complete system that can directly replace human control, but rather as components that need to be used in conjunction with engineering-level safety mechanisms, such as human intervention and geofencing.

8.3 Dual-Use Concerns and Potential Misuse

UAV VLN technology exhibits clear dual-use characteristics. While it can support positive applications such as search-and-rescue and infrastructure inspection, it may also be misused for surveillance, tracking, or other inappropriate purposes. We emphasize that the benchmark and models are intended solely for research and lawful applications, and we recommend clearly specifying usage scope and restrictions when releasing code and models to mitigate potential misuse risks.

References

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. 2018. 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.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. 2025a. Qwen3-vl technical report. *Preprint*, arXiv:2511.21631.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. 2025b. Qwen2.5-vl technical report. *Preprint*, arXiv:2502.13923.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. 2024. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Yue Fan, Winson Chen, Tongzhou Jiang, Chun Zhou, Yi Zhang, and Xin Eric Wang. 2022. Aerial vision-and-dialog navigation. *arXiv preprint arXiv:2205.12219*.

Yunpeng Gao, Chenhui Li, Zhongrui You, Junli Liu, Zhen Li, Pengan Chen, Qizhi Chen, Zhonghan Tang, Liansheng Wang, and Penghui Yang. 2025. Openfly: A comprehensive platform for aerial vision-language navigation.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Qingyong Hu, Bo Yang, Sheikh Khalid, W. Xiao, Niki Trigoni, and A. Markham. 2022. Sensaturban: Learning semantics from urban-scale photogrammetric point clouds. *International Journal of Computer Vision*, pages 1–28.

Tao Hu, Qingsen Yan, Yuankai Qi, and Yanning Zhang. 2024. Generating content for hdr deghosting from frequency view. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 25732–25741.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, et al. 2025. Openvla: An open-source vision-language-action model. In *Conference on Robot Learning*, pages 2679–2713. PMLR.

William Labov. 1973. *Sociolinguistic Patterns*. University of Pennsylvania Press.

Jungdae Lee, Taiki Miyanishi, Shuhei Kurita, Koya Sakamoto, Daichi Azuma, Yutaka Matsuo, and Nakamasa Inoue. 2024. Citynav: Language-goal aerial navigation dataset with geographic information. *Preprint*, arXiv:2406.14240.

Shubo Liu, Hongsheng Zhang, Yuankai Qi, Peng Wang, Yaning Zhang, and Qi Wu. 2023. Aerialvln: Vision-and-language navigation for uavs. *Preprint*, arXiv:2308.06735.

Dipendra Misra, Andrew Bennett, Valts Blukis, Eyvind Niklasson, and Yoav Artzi. 2018. Mapping instructions to actions in 3d environments with visual goal prediction.

Taiki Miyanishi, Fumiya Kitamori, Shuhei Kurita, Jungdae Lee, Motoaki Kawanabe, and Nakamasa Inoue. 2023. Cityrefer: Geography-aware 3d visual grounding dataset on city-scale point cloud data.

Don Norman. 2013. *The Design of Everyday Things: Revised and Expanded Edition*. The Design of Everyday Things: Revised and Expanded Edition.

OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino, Alex Renzin, Alex Tachard Passos, Alexander Kirillov, Alexi Christakis, Alexis Conneau, Ali Kamali, Allan Jabri, Allison Moyer, Allison Tam, Amadou Crookes, Amin Tootoochian, Amin Tootoonchian, Ananya Kumar, Andrea Vallone, Andrej Karpathy, Andrew Braunstein, Andrew Cann, Andrew Codispoti, Andrew Galu, Andrew Kondrich, Andrew Tulloch, Andrey Mishchenko, Angela Baek, Angela Jiang, Antoine Pelisse, Antonia Woodford, Anuj Gosalia, Arka Dhar, Ashley Pantuliano, Avi Nayak, Avital Oliver, Barret Zoph, Behrooz Ghorbani, Ben Leimberger, Ben Rossen, Ben Sokolowsky, Ben Wang, Benjamin Zweig, Beth Hoover, Blake Samic, Bob McGrew, Bobby Spero, Bogo Giertler, Bowen Cheng, Brad Lightcap, Brandon Walkin, Brendan Quinn, Brian Guarraci, Brian Hsu, Bright Kellogg, Brydon Eastman, Camillo Lugaresi, Carroll Wainwright, Cary Bassin, Cary Hudson, Casey Chu, Chad Nelson, Chak Li, Chan Jun Shern, Channing Conger, Charlotte Barette, Chelsea Voss, Chen Ding, Cheng Lu, Chong Zhang, Chris Beaumont, Chris Hallacy, Chris Koch, Christian Gibson, Christina Kim, Christine Choi, Christine McLeavey, Christopher Hesse, Claudia Fischer, Clemens Winter, Coley Czarnecki, Colin Jarvis, Colin Wei, Constantin Koumouzelis, Dane Sherburn, Daniel Kappler, Daniel Levin, Daniel Levy, David Carr, David Farhi, David Mely, David Robinson, David Sasaki, Denny Jin, Dev Valladares, Dimitris Tsipras, Doug Li, Duc Phong Nguyen, Duncan Findlay, Edede Oiwoh, Edmund Wong, Ehsan Asdar, Elizabeth Proehl, Elizabeth Yang, Eric Antonow, Eric Kramer, Eric Peterson, Eric Sigler, Eric Wallace, Eugene Brevdo, Evan Mays, Farzad Khorasani, Felipe Petroski Such, Filippo Raso, Francis Zhang, Fred von Lohmann, Freddie Sulit, Gabriel Goh, Gene Oden, Geoff Salmon, Giulio Starace, Greg Brockman, Hadi Salman, Haiming Bao, Haitang Hu, Hannah Wong, Haoyu Wang, Heather Schmidt, Heather Whitney, Heewoo Jun, Hendrik Kirchner, Henrique Ponde de Oliveira Pinto, Hongyu Ren, Huiwen Chang, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian O'Connell, Ian Osband, Ian Silber, Ian Sohl, Ibrahim Okuyucu, Ikai Lan, Ilya Kostrikov, Ilya Sutskever, Ingmar Kanitscheider, Ishaan Gulrajani, Jacob Coxon, Jacob Menick, Jakub Pachocki, James Aung, James Betker, James Crooks, James Lennon, Jamie Kiros, Jan Leike, Jane Park, Jason Kwon, Jason Phang, Jason Teplitz, Jason Wei, Jason Wolfe, Jay Chen, Jeff Harris, Jenia Varavva, Jessica Gan Lee, Jessica Shieh, Ji Lin, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joanne Jang, Joaquin Quinonero Candela, Joe Beutler, Joe Landers, Joel Parish, Johannes Heidecke, John Schulman, Jonathan Lachman, Jonathan McKay, Jonathan Uesato, Jonathan Ward, Jong Wook Kim, Joost Huizinga, Jordan Sitkin, Jos Kraaijeveld, Josh Gross, Josh Kaplan, Josh Snyder, Joshua Achiam, Joy Jiao, Joyce Lee, Juntang Zhuang, Justyn Harriman, Kai Fricke, Kai Hayashi, Karan Singhal, Katy Shi, Kavin Karthik, Kayla Wood, Kendra Rimbach, Kenny Hsu, Kenny Nguyen, Keren Gu-Lemberg, Kevin Button, Kevin Liu, Kiel Howe, Krithika Muthukumar, Kyle Luther, Lama Ahmad, Larry Kai, Lauren Itow, Lauren Workman, Leher Pathak, Leo Chen, Li Jing, Lia Guy, Liam Fedus, Liang Zhou, Lien Mamitsuka, Lilian Weng, Lindsay McCallum, Lindsey Held, Long Ouyang, Louis Feuvrier, Lu Zhang, Lukas Kondraciuk, Lukasz Kaiser, Luke Hewitt, Luke Metz, Lyric Doshi, Mada Aflak, Maddie Simens, Madelaine Boyd, Madeleine Thompson, Marat Dukhan, Mark Chen, Mark Gray, Mark Hudnall, Marvin Zhang, Marwan Aljubeh, Mateusz Litwin, Matthew Zeng, Max Johnson, Maya Shetty, Mayank Gupta, Meghan Shah, Mehmet Yatbaz, Meng Jia Yang, Mengchao Zhong, Mia Glaese, Mianna Chen, Michael Janner, Michael Lampe, Michael Petrov, Michael Wu, Michele Wang, Michelle Fradin, Michelle Pokrass, Miguel Castro, Miguel Oom Temudo de Castro, Mikhail Pavlov, Miles Brundage, Miles Wang, Minal Khan, Mira Murati, Mo Bavarian, Molly Lin, Murat Yesildal, Nacho Soto, Natalia Gimelshein, Natalie Cone, Natalie Staudacher, Natalie Summers, Natan LaFontaine, Neil Chowdhury, Nick Ryder, Nick Stathas, Nick Turley, Nik Tezak, Niko Felix, Nithanth Kudige, Nitish Keskar, Noah Deutsch, Noel Bundick, Nora Puckett, Ofir Nachum, Ola Okelola, Oleg Boiko, Oleg Murk, Oliver Jaffe, Olivia Watkins, Olivier Godement, Owen Campbell-Moore, Patrick

Chao, Paul McMillan, Pavel Belov, Peng Su, Peter Bak, Peter Bakkum, Peter Deng, Peter Dolan, Peter Hoeschele, Peter Welinder, Phil Tillet, Philip Pronin, Philippe Tillet, Prafulla Dhariwal, Qiming Yuan, Rachel Dias, Rachel Lim, Rahul Arora, Rajan Troll, Randall Lin, Rapha Gontijo Lopes, Raul Puri, Reah Miyara, Reimar Leike, Renaud Gaubert, Reza Zamani, Ricky Wang, Rob Donnelly, Rob Honsby, Rocky Smith, Rohan Sahai, Rohit Ramchandani, Romain Huet, Rory Carmichael, Rowan Zellers, Roy Chen, Ruby Chen, Ruslan Nigmatullin, Ryan Cheu, Saachi Jain, Sam Altman, Sam Schoenholz, Sam Toizer, Samuel Miserendino, Sandhini Agarwal, Sara Culver, Scott Ethersmith, Scott Gray, Sean Grove, Sean Metzger, Shamez Hermani, Shantanu Jain, Shengjia Zhao, Sherwin Wu, Shino Jomoto, Shirong Wu, Shuaiqi, Xia, Sonia Phene, Spencer Papay, Srinivas Narayanan, Steve Coffey, Steve Lee, Stewart Hall, Suchir Balaji, Tal Broda, Tal Stramer, Tao Xu, Tarun Gogineni, Taya Christianson, Ted Sanders, Tejal Patwardhan, Thomas Cunninghman, Thomas Degry, Thomas Dimson, Thomas Raoux, Thomas Shadwell, Tianhao Zheng, Todd Underwood, Todor Markov, Toki Sherbakov, Tom Rubin, Tom Stasi, Tomer Kaftan, Tristan Heywood, Troy Peterson, Tyce Walters, Tyna Eloundou, Valerie Qi, Veit Moeller, Vinnie Monaco, Vishal Kuo, Vlad Fomenko, Wayne Chang, Weiyi Zheng, Wenda Zhou, Wesam Manassra, Will Sheu, Wojciech Zaremba, Yash Patil, Yilei Qian, Yongjik Kim, Youlong Cheng, Yu Zhang, Yuchen He, Yuchen Zhang, Yujia Jin, Yunxing Dai, and Yury Malkov. 2024. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

John Pruitt and Jonathan Grudin. 2003. Personas: Practice and theory. In *Proceedings of the 2003 Conference on Designing for User Experiences*.

Sali A. Tagliamonte. 2011. *Variationist Sociolinguistics: Change, Observation, Interpretation*. John Wiley and Sons / Wiley-Blackwell.

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

Jiazhao Zhang, Kunyu Wang, Shaoan Wang, Minghan Li, Haoran Liu, Songlin Wei, Zhongyuan Wang, Zhizheng Zhang, and He Wang. 2024. Uni-navid: A video-based vision-language-action model for unifying embodied navigation tasks. *arXiv preprint arXiv:2412.06224*.

Fengda Zhu, Xiwen Liang, Yi Zhu, Xiaojun Chang, and Xiaodan Liang. 2021. Soon: Scenario oriented object navigation with graph-based exploration.

A Related Work

This section provides a detailed analysis of representative UAV VLN benchmarks, summarizing their design characteristics as well as their strengths and limitations.

LANI (Misra et al., 2018) is one of the earliest UAV VLN benchmarks. It is constructed in an open simulated environment and supports the evaluation of basic path-following capabilities. However, due to its simplified scenes and the lack of explicit spatial structure and photorealism, the overall task difficulty is relatively low, making it insufficient to reflect the complexity of real-world navigation scenarios.

AVDN (Fan et al., 2022) introduces a multi-turn natural language interaction mechanism that emphasizes human-robot collaboration. The dataset is collected in real-world environments, which enhances the realism of language interactions. Nevertheless, AVDN suffers from limited data scale and relatively short navigation paths. Moreover, its reliance on manually designed dialogue procedures makes it difficult to directly apply to large-scale evaluation settings.

AerialVLN (Liu et al., 2023) is constructed using multiple simulated environments, and its instruction design explicitly includes intermediate navigation steps. This formulation facilitates the study of alignment between language and actions. However, as the dataset is entirely based on simulation, it lacks the visual details of real urban environments, making it challenging to evaluate model generalization performance under practical conditions.

CityNav (Lee et al., 2024) is a large-scale dataset sourced from real aerial imagery of urban environments, offering strong realism in terms of visual texture and scene diversity. However, its instructions mainly focus on target descriptions and do not include intermediate navigation processes, which limits its effectiveness in evaluating models' spatial reasoning and step-by-step navigation capabilities.

OpenUAV (Wang et al., 2024) is built on a high-fidelity simulation platform that supports 6-DoF flight and multi-view perception. Its instructions are generated by LLMs and subsequently refined by human annotators, resulting in task settings that are closer to real navigation requirements. Nonetheless, the dataset remains dependent on virtual environments, and its overall scale is relatively limited, restricting coverage of complex and diverse real scenarios.

OpenFly (Gao et al., 2025) leverages multi-source rendering engines to achieve high visual diversity and realism, and employs an automated toolchain to generate large-scale navigation trajectories and instructions, significantly improving data construction efficiency and task complexity. However, its instructions are entirely model-generated, lacking the linguistic habits and stylistic variations of human language. As a result, the overall naturalness and diversity of instruction expressions remain limited.

B Evaluation Metrics

This section provides detailed definitions of the evaluation metrics used in our experiments.

- **Navigation Error (NE)**: the Euclidean distance between the agent's final position and the ground-truth target location. Lower values indicate higher accuracy.
- **Success Rate (SR)**: the proportion of episodes in which the agent terminates within the target distance threshold.
- **Oracle Success Rate (OSR)**: the proportion of trajectories that enter the target threshold at any step, regardless of whether the agent stops correctly.
- **Success weighted by Path Length (SPL)**: a success metric that penalizes redundant paths, encouraging shorter and more efficient trajectories.

C User Personas for Navigation Instruction Generation

The user personas and their corresponding language style preferences are summarized in Table 3.

D Case Study on Persona-Specific Instructions

We further conduct a qualitative analysis through case studies to examine differences in instructions generated under different personas. Table 4 presents several representative instruction examples corresponding to different personas.

Taking the **Courier / Delivery worker** persona as an example, the instructions are clearly oriented toward task execution and road navigation contexts. The language heavily relies on traffic-related facilities and drivable elements as core references, such

ID	Persona	Age Group	Social Role / Background	Preferred Language Style
P1	Primary school student	Child	Student with limited spatial experience	Simple wording, action-oriented, highly explicit instructions
P2	University student	Young adult	Student, frequent user of map-based tools	Concise and direct, efficiency-focused, path-oriented terminology
P3	Office worker	Young to middle-aged	Daily commuter, familiar with urban structure	Structured expression, preference for optimal routes
P4	Community staff	Middle-aged	Property management / security / administrative staff	Stable phrasing, safety reminders, familiarity with local landmarks
P5	Courier / Delivery worker	Young to middle-aged	High-frequency navigation user	Highly efficient instructions, emphasis on path clarity and ordering
P6	Urban tourist	Young to middle-aged	Temporary visitor, unfamiliar with the environment	Landmark-rich descriptions, strong explanations, contextual guidance
P7	Retired adult	Elderly	Non-technical user, slower interaction pace	Redundant explanations, repeated reminders, focus on safety and comfort
P8	Teacher / Educator	Middle-aged	Emphasis on logic and clarity	Formal language, well-structured instructions, pedagogical tone
P9	Community volunteer	Middle-aged to elderly	Active resident, frequent participation in public affairs	Friendly phrasing, landmark-oriented guidance, everyday language
P10	Advanced navigation user	Young to middle-aged	Navigation expert / experienced user	Minimal technical language, clear structure, preference for optimal routes

Table 3: User personas and language style preferences for navigation instruction generation.

as *intersection*, *multi-lane road*, *lane markings*, and *parking area*. Path descriptions are closely organized around intersection choices, lane following, and parking area localization, reflecting a strong preference for efficiency and accessibility.

Instructions from the **Teacher / Educator** persona exhibit a distinct instructional and guidance-oriented characteristic. The expressions are typically organized in a progressive manner, with an emphasis on maintaining directional consistency and intermediate states (e.g., *keep your heading*, *maintain the same heading*) to reduce cognitive load during understanding and execution. In addition, landmarks are described in a more detailed and reproducible manner, such as explicit specifications of track color or lane line styles, ensuring that each step is grounded in stable and clear perceptual cues.

In contrast, instructions from the **Retired adult** persona place greater emphasis on communication comfort and everyday experience. The language style is more conversational, with a gentle and reassuring tone (e.g., *Alright, nice and easy now*, *When you're ready*, *go forward a bit more*). These instructions tend to rely on landmarks closely related to daily life, such as *footbridge*, *creek*, and *sheds*. Precise constraints are intentionally reduced, with greater focus on situational guidance and pace control.

Overall, different personas exhibit systematic differences in information focus, path organization, and interaction style. These differences are closely associated with their respective occupational backgrounds and life experiences, enabling AirNav to generate navigation instructions that better align with real human expression habits and demonstrate higher instruction diversity.

E Instruction Naturalness Evaluation Prompt

Prompt

Role:
You are an expert language evaluation assistant for UAV navigation instructions.

Objective:
Your task is to assess the Naturalness of a navigation instruction as it would be spoken by a real human guiding a UAV in a real-world scenario.

Evaluation Criteria:
Evaluate the instruction based on the following criteria:
1. Naturalness: Whether the instruction sounds like spontaneous human speech rather than a rigid, scripted, or templated command.
2. Practicality: Whether the instruction provides actionable route guidance through landmarks, relative directions, or intermediate cues, rather than low-level action enumeration.
3. Human Alignment: Whether the wording and structure align with how a human would naturally phrase a navigation request in everyday use.

Rating Scale:
Rate it on a scale from 1 to 5:
- 1 = very unnatural (robotic, templated, or action-list-like)
- 2 = somewhat unnatural (syntactically valid but awkward or artificial)
- 3 = neutral (reasonable but not strongly human-like)
- 4 = mostly natural (sounds like something a person might naturally say)
- 5 = very natural (fluent, realistic, and clearly human-like)

Output Requirement:
Your output must be a single integer from 1 to 5 representing the overall naturalness score. Only output the number, and do not provide any explanation.

Task:
Evaluate the following UAV navigation instruction according to the criteria above.

Navigation Instruction:
`{navigation instruction}`

Evaluation Protocol. For each dataset, we randomly sample 2,000 instructions and score them using the prompt above with GPT-4o. To reduce randomness from single-pass evaluation, each instruction is independently scored three times, and the final score is the average of the three results. Note that datasets whose instructions only describe the target object without any intermediate path or process-level guidance are excluded from the comparison to avoid unfair evaluation.

F Human Annotation Setup and Statistical Analysis

To validate the reliability of LLM-based automatic evaluation for instruction naturalness, we introduce

Persona	Instruction Example
Courier / Delivery Worker	Turn left and move forward until the curved road intersection with white painted lines and a central grass island is ahead. At the intersection, turn left and move forward. Move forward, then turn right and move forward until you reach the curved roadway intersection with grassy medians and a large tree in the middle. Move forward until the multi-lane road with white lane markings and a pedestrian crossing appears beside parked cars and industrial buildings. Move forward along this road, then turn left toward the small parking area by the light-gray industrial buildings. Move forward to the bright red car and stop.
Teacher / Educator	Begin by turning right until the tan running track with a solid white lane line near the infield edge is aligned ahead. Move forward toward it, then make a slight right and continue forward along the track edge until you reach the section with the white lane line. From this point, keep your heading and proceed straight toward the curved arc of blue-and-white seating outside the track, continuing forward until you arrive beside the seating. Maintain the same heading and advance straight toward the row of tightly parked white cars along the side of the long gray-roof workshop, continuing until you reach that row. Continue straight along the workshop side, then make a slight left and move forward to the small red car parked in the lot beside the gray-roof warehouses, stop.
Retired Adult	Alright, nice and easy now. Begin by turning right, then move forward at a comfortable pace. When you're ready, turn right again and continue moving forward until you come to the large industrial building with several gray metal roofs and a parking lot that has white trucks beside it. From there, move forward a touch and turn left, then keep moving forward, slow and steady, toward the row of white and blue trucks in the lot. Continue moving forward past those trucks, and when you feel settled, turn left and go forward until the short white footbridge over the narrow dark creek is ahead. At the footbridge, turn right and keep moving forward along the yard until you reach the row of narrow gray-roofed sheds with multiple bays. Go forward a bit more, then turn left and move forward toward the blue car parked among other vehicles in the lot, and stop.

Table 4: Representative Instruction Examples for Different Personas

human annotation as a calibration reference and analyze the consistency between LLM scores and human judgments.

Human Annotation Setup We randomly sample a total of 500 instructions from all datasets involved in the instruction naturalness analysis, with approximately 100 instructions drawn from each dataset, to form the human-annotated subset. Each instruction is independently evaluated by three annotators. All annotators are fluent in English and have a basic understanding of UAV VLN tasks. Human annotation follows the same scoring scheme as the automatic evaluation. Prior to annotation, all annotators are provided with a unified annotation guideline to ensure a consistent understanding of the evaluation dimensions and scoring scale.

Inter-Annotator Agreement We first analyze the inter-annotator agreement to assess the stability of human judgments. Specifically, we adopt Krippendorff's α as the agreement metric. The results show that Krippendorff's α reaches 0.70, indicating moderate to substantial agreement among annotators and a stable consensus in their interpretation of the evaluation criteria.

Human–LLM Correlation We further examine the correlation between human annotations and LLM-based scores. For each instruction, the scores from the three annotators are averaged to obtain a human reference score, which is then compared with the corresponding LLM score using Spearman's rank correlation coefficient (ρ). The results show a strong positive correlation, with Spearman's ρ reaching 0.74, suggesting that LLM-based scoring effectively captures the overall trend of human judgment on instruction naturalness.

Discussion The above results indicate that LLM-based automatic evaluation is highly consistent with human judgments in instruction naturalness assessment, while significantly reducing evaluation costs. Therefore, we adopt LLM-based scoring as the instruction naturalness evaluation tool in the main experiments to enable systematic comparison across large-scale datasets.

G Case Study on Instruction Naturalness

We further conduct a qualitative analysis of instruction naturalness across different datasets through case studies. Specifically, we randomly select two instruction examples from each dataset, as shown in Table 5, and compare them along multiple dimensions related to naturalness.

Taking AerialVLN as an example, its instructions are typically composed of a sequence of low-level actions, with repeated use of expressions such

as *turn around*, *continue straight*, and *go over the buildings*. Although macro-level landmarks such as lakes and fountains are mentioned, these landmarks are not further specified with perceptually distinguishable attributes. In addition, turning descriptions such as *turn up and left* are semantically vague and difficult to precisely associate with concrete spatial changes. This style is closer to programmatic control commands than to natural language instructions used by humans during real navigation.

Instructions in the LANI dataset exhibit clearer executability, as they often constrain motion trajectories using precise turning angles (e.g., *Turn 60 degrees*), thereby reducing ambiguity. However, this highly numerical form of expression does not align well with common human navigation communication habits, as people rarely rely on frequent and exact angle specifications in real-world scenarios. Moreover, the referenced landmarks are typically simple and local small objects (e.g., *green cactus* and *potted plant*), whose visual salience and recognizability are limited from a UAV perspective, weakening the alignment between linguistic descriptions and visual perception.

In contrast, instructions in our dataset demonstrate characteristics that are closer to human navigation language. They tend to rely on landmarks with higher visual salience and discriminability, which are further specified through attributes such as color, shape, and spatial relationships (e.g., *dark-gray rectangular roof*, *curved dirt racetrack*, and *single white car*), thereby reducing potential ambiguity. In terms of action description, our instructions more frequently adopt progressive and relative expressions (e.g., *make a slight left* and *continue past . . . , then turn . . .*), rather than relying heavily on precise angle values, making them more consistent with human navigation language usage.

H Prompt Template for AirVLN-R1

Prompt

Role:
You are an expert navigation assistant for a UAV (Unmanned Aerial Vehicle) flight simulator.

Task Objective:
The UAV operates in an urban environment with visible roads, buildings, and landmarks. Your task is to predict the next sequence of UAV actions based on:
1. A given natural language navigation instruction,
2. The current state of the UAV, including its position and heading angle,
3. The current first-person UAV view image,
4. Up to four historical first-person view images from previous time steps (if available),
5. The previously executed UAV actions (if available).

Text Input:
- Navigation instruction: `{Instruction}`
- Current state of the UAV: `{Current State}`
- Previously executed actions: `{Historical Action sequence}`
(A list of past actions the UAV has taken, in chronological order.)

Image Input:
UAV (Unmanned Aerial Vehicle) View Sequence
- Historical views (from oldest to newest) show the UAV's past observations.
- The last image represents the UAV's current view.
- In all images, the top of the frame corresponds to the UAV's forward direction (its heading).

Step-by-Step Action Planning:
Based on the navigation instruction, the UAV's current state, the previously executed actions (which can help infer the UAV's current orientation and progress), and the provided images, predict how the UAV should move step by step to follow the instruction accurately.

Prediction Rules:
1. Predict no more than 8 future actions for the UAV to execute.
2. If the target location is reachable in fewer than 8 actions, output less than 8 actions sequence and end with "STOP". Otherwise, it clearly requires more than 8 actions to approach the target, output exactly 8 future actions.
3. You must output "STOP" if the UAV has already reached the described target.
4. Output a JSON list of actions, in the exact order they should be executed.
5. Do not include any explanations, reasoning, or additional text — only output the JSON list.

Discrete Action Space:
- MOVE_FORWARD: move straight 5 meters in the current heading
- TURN_LEFT: rotate left 30 degrees
- TURN_RIGHT: rotate right 30 degrees
- STOP: stop the flight

Output Format Examples:
["TURN_RIGHT", "TURN_RIGHT", "MOVE_FORWARD", "MOVE_FORWARD", "MOVE_FORWARD", "MOVE_FORWARD", "MOVE_FORWARD", "MOVE_FORWARD"]
["MOVE_FORWARD", "MOVE_FORWARD", "STOP"]
["STOP"]

I Baseline Model Descriptions

We provide a brief overview of the baseline models evaluated on the AirNav benchmark.

- **Seq2Seq** (Anderson et al., 2018): A classic end-to-end model that directly encodes both language and visual inputs, mapping them to corresponding action sequences. This model serves as the basic comparison method.
- **CMA** (Hu et al., 2024): A model that employs a cross-modal attention mechanism, effectively integrating language and visual information to make navigation decisions.

- **Qwen2.5-VL (7B / 32B)** (Bai et al., 2025b): A MLLM developed and open-sourced by Qwen team, Alibaba Cloud. It supports both image input and language understanding, with different parameter sizes, making it suitable for analyzing the impact of model capacity on navigation tasks.
- **Qwen3-VL-235B-A22B** (Bai et al., 2025a): A large-scale Mixture-of-Experts vision-language model released by the Qwen team, with 235B total parameters and 22B activated parameters per token, making it a strong reference model for evaluating the upper-bound performance of general-purpose MLLMs in navigation tasks.
- **LLaMA-3.2-11B-Vision** (Grattafiori et al., 2024): An open-source MLLM released by Meta, which combines image recognition and language reasoning abilities to evaluate the perceptual and decision-making capabilities of LLM in navigation tasks.
- **GPT-4o** (OpenAI et al., 2024): A powerful multimodal general-purpose model launched by OpenAI, supporting joint text and image input. With its excellent reasoning capabilities, it serves as an important reference for evaluating task performance.
- π_0 (Black et al., 2024): A generalist Vision-Language-Action model pre-trained on large-scale robot manipulation data, built upon PaliGemma with a flow-matching action expert. Since π_0 originally outputs continuous actions, we replace its action head with a discrete action prediction head to adapt it to the UAV VLN task.
- **OpenVLA** (Kim et al., 2025): An open-source 7B-parameter Vision-Language-Action model built upon the Prismatic-7B VLM backbone (Llama 2 7B with fused DINOv2 and SigLIP visual encoders), fine-tuned on 970K real-world robot manipulation trajectories from the Open X-Embodiment dataset.
- **Uni-NaVid** (Zhang et al., 2024): A unified video-based VLA model trained on large-scale multi-task navigation data, covering diverse embodied navigation settings including object navigation, instruction following, and person tracking.
- **OpenFly-Agent** (Gao et al., 2025): A keyframe-aware VLN model specifically designed for aerial navigation, which leverages key visual observations to improve navigation efficiency and accuracy.

J Hyperparameter Settings for SFT and RFT Stages

The hyperparameter settings used in the SFT and RFT stages are summarized in Table 6.

K Real-World Test Setup and Test Cases

The experimental UAV was a DJI Tello TLW004, equipped with a camera, a Vision Positioning System, a barometer, an infrared sensor, and an inertial measurement unit. During testing, the UAV first transmitted its real-time images and state information to a local laptop via a network. The laptop then forwarded the data to a remote server or cloud platform where the model was deployed. Upon receiving all input information, the model generated the corresponding action sequence, which was subsequently executed by the UAV through its built-in motion control API. In both indoor and outdoor settings, the UAV maintains a fixed altitude throughout each navigation task; ascending and descending actions are excluded from the action space. Navigation trajectories are therefore planar, which is consistent with the benchmark design and simplifies control for fair cross-method comparison. Fig. 4 shows two representative real-world navigation tasks conducted in indoor and outdoor environments.

L Persona-Level Evaluation Results

Table 7 reports the SR (%) of all models on the test-unseen split under each of the 10 user personas.

M Additional Ablation Studies

M.1 Historical View Image Selection

To systematically analyze the impact of different historical view image selection strategies on navigation performance, we compare four strategies under the SFT+RFT training paradigm and evaluate them on the test-unseen split. Specifically, the evaluated strategies include:

- **No-History**, which uses only the current view image without incorporating any historical observations;
- **Last-K**, which always selects the most recent K historical view images;
- **Uniform-K**, which uniformly samples K images within a fixed historical window;

(a) Indoor Scene (b) Outdoor Scene

Figure 4: Real-World UAV VLN Deployment in Indoor and Outdoor Scenes.

- **Progressive Interval Sampling (Ours)**, which adopts a non-uniform sampling strategy with progressively increasing temporal intervals.

Table 8 reports the performance of different strategies on the test-unseen split. Several observations can be drawn. First, the **No-History** baseline, which relies solely on the current view image, performs the worst, with NE as high as 121.6 and SR of only 22.96%. This result indicates that single-frame visual observations are insufficient to provide adequate temporal and spatial context for reliable decision-making. Second, the **Last-K** strategy significantly improves performance by retaining the most recent K historical images, enabling the model to capture short-term motion trends and local environmental changes. As a result, the SR increases to 45.1%. However, since this strategy ignores earlier but potentially important observations, the model's ability to reason about long-term navigation progress remains limited. Third, **Uniform-K** further improves performance by uniformly sampling historical images within a fixed window, allowing the model to access observations at multiple temporal scales. This leads to a lower NE of 41.1 and a higher SR of 49.83%, suggesting that covering a longer temporal span of historical information helps the model better understand the global path structure. Nevertheless, due to its fixed sampling interval, this strategy lacks an explicit mechanism to model the temporal importance difference between recent and distant observations. Finally, **Progressive Interval Sampling** achieves the best performance across all metrics, yielding the lowest NE (39.8) and the highest SR (51.82%) on the test-unseen split. Compared to Uniform-K, it further improves both navigation success and path execution efficiency. These results demonstrate that non-uniform sampling with progressively increasing intervals more effectively balances fine-grained short-term perception and long-term contextual modeling.

M.2 Ablation on Reward Components

To further quantify the contribution of each reward component in the RFT stage, we conduct a series of ablation experiments based on the same SFT-initialized model. Specifically, different components of the reward function are removed, and all models are evaluated on the test-unseen split for comparison. The following ablation settings are considered:

- **w/o Subgoal State Alignment**: removing the Distance-to-Subgoal and Heading Angle Alignment reward, while retaining the Stop Consistency and Format Reward;
- **w/o Stop Consistency**: removing the Stop Consistency reward;
- **w/o Format Reward**: removing the reward that enforces output validity.

Table 9 reports the experimental results under different settings on the test-unseen split, where **SFT-only** serves as the baseline without RFT. The results lead to the following observations. First, the Subgoal State Alignment reward is the primary driver of performance improvement. Removing this component results in the most significant degradation across all metrics, with NE increasing from 39.8 to 46.1 and SR dropping sharply from 51.82% to 46.00%. This indicates that Subgoal State Alignment plays a critical role in guiding effective path planning. Second, the Stop Consistency reward affects the reliability of **stop** decisions. Without this reward, the model exhibits a

substantially higher frequency of both early-stop and missed-stop, leading to a noticeable decline in SR (51.82% $\rightarrow$ 47.65%). This result highlights the importance of explicitly supervising **when to stop** in UAV VLN tasks for stable execution and successful task completion. Third, the Format reward has a relatively limited impact on the final performance metrics but contributes positively to training stability and model usability. Since the output structure is already well constrained during the SFT stage, removing the Format reward leads only to a slight performance drop. Nevertheless, this reward helps reduce invalid generations, thereby improving the stability of the RFT and the reliability of practical deployment. Overall, the three reward components exhibit complementary roles: the Subgoal State Alignment reward ensures that the agent **moves in the correct direction**, the Stop Consistency reward encourages the agent to **stop at the appropriate time**, and the Format reward guarantees that the **outputs are valid**. Together, they form a systematic reward design tailored for UAV VLN, resulting in improved performance and stronger generalization.

M.3 Sensitivity Analysis of Reward Weights

The overall reward function involves five weighting parameters: λ_1, λ_2, α, β, and γ. The component-level ablation in Table 9 has already demonstrated that Subgoal State Alignment (governed by λ_1 and λ_2) is the dominant contributor to performance, while Stop Consistency (α) and Format Reward (β, γ) have comparatively smaller and stable effects. We therefore focus the sensitivity analysis on λ_1 and λ_2, as their relative weighting directly controls the balance between distance alignment and heading alignment within the most critical reward component.

Keeping all other RFT settings strictly unchanged (including the initialized model, GRPO configuration, learning rate, training steps, and the stop/format rewards), we conduct a 3×3 grid search by varying λ_1 and λ_2 over $\{0.5, 1.0, 2.0\}$, and evaluate SR on test-unseen.

As shown in Table 10, performance remains stable across the tested range, with the maximum SR fluctuation within 1.5 percentage points. The default setting ($\lambda_1 = 1$, $\lambda_2 = 1$) achieves slightly better results, but multiple neighboring configurations yield comparable performance, indicating that the method does not depend on a precise weight choice.

N Real-World Evaluation Results

Table 11 summarizes the performance, resource consumption, and inference latency of different methods in real-world UAV VLN experiments.

O Analysis of Resource Cost and Inference Efficiency

Table 11 shows the performance and resource usage comparison of different models. The analysis is as follows:

- **Traditional models are lightweight but impractical:** Seq2Seq and CMA exhibit low inference latency and memory consumption; however, they nearly fail to accomplish any navigation tasks in real-world tests, succeeding on only 1–2 out of 200 tasks, resulting in limited practical usability.
- **Deployment cost grows substantially for large models:** Although Qwen2.5-VL-32B achieves better performance than Qwen2.5-VL-7B, it incurs significantly higher computational demands and inference latency, requiring an A100 80GB GPU, which makes it difficult to meet the real-time requirements.
- **Cloud models introduce controllability and real-time risks:** GPT-4o avoids local memory usage via cloud inference, but still has a latency of 3.674 s/step and may impose potential constraints on system controllability and data privacy.
- **AirVLN-R1 balances performance and efficiency:** AirVLN-R1 achieves the highest SR in real-world tests while keeping the computational overhead acceptable, providing a more balanced trade-off between performance and deployment cost.

Dataset	Example 1	Example 2
LANI	Move forward and stay to the right of the red ball. Continue traveling in a straight line until you pass the brown chair, which will be on your left. Just after passing the chair, turn almost 90 degrees to the left, and continue in a straight line until you pass a green cactus on your left. Turn 60 degrees to your left, and continue in a straight line until you pass a potted plant on your left. Turn 45 degrees left just past the potted plant, and then continue in a straight line until a traffic cone is immediately on your right.	Move forward towards the lamp and move past it on its left. Move forward towards the chest and move past it on its left. Stop before hitting the red fence and turn right. Move forward towards the grey object and move past it on its left. Stop before hitting the white fence.
AVDN	"[INS] Destination is a long row of short cargo containers bisecting the island of dirt at your eight o'clock direction..", "[QUE] I am on top of the many small containers, Can I see the destination? Which direction should I go?." "[INS] Northeast, fly to your right and turn around and go the upposit direction just a few feet to your destination." "[QUE] I move to the northeast, Am I near the destination? Which direction should I go?." "[INS] Turn 180 degrees. Go straight until you are over a white container. That is your destination.".	"[INS] Please go to the southeast direction at 5 o'clock. The destination is blue containers." "[QUE] I am on top of many containers, Am I near the destination? Where should I go now?" "[INS] Yes, you are very close to your destination. Turn to your 5 o'clock and go straight forward from there and you will be right on top of your destination."
AerialVLN	Go up the building and fly to the left near the lake. stop at the middle of the lake and turn up and left and prepare for take off. go over the trees and bushes until you see the fountain. go over the fountain and over one building until you see the edge of the ocean. turn around and head towards the spotted lake and land near the trees. turn around and continue straight and over the red tree. turn around and continue straight.	Take off and turn right and move forward and cross the buildings terrace and turn left. now move forward and go over the buildings towards the white building terrace and turn left. now move forward cross the buildings and go over the grass and roads and reach the under the building. now move forward and turn right and go over the buildings and towards the brown building terrace and stay there.
OpenFly	Advance forward to a multi-colored residential area predominantly featuring beige and yellow tones. The area consists of a cluster of low-rise residential buildings with flat rooftops and tree - lined parallel streets. Slightly turn left and proceed straight to reach a yellow medium-sized residential building known for its rectangular structure, repetitive windows, and dark roofs. Shift right to find a medium-sized beige and gray residential building with multiple floors and uniform windows. Move ahead to encounter a medium-sized building with a brown and gray exterior, rooftop garden, and a rectangular shape, surrounded by structures of comparable scale. Slightly turn left again and proceed straight to it.	Proceed to the gray skyscraper building, then turn right towards the green golf course with trees and a pond, an expansive scenic outdoor area. Continue straight ahead, subtly veer left, and proceed straight towards the large green tree with coniferous leaves.
Ours	Turn left to face the grassy park with scattered trees and a curved edge path. Move forward, make a slight left, and reach this park. Continue straight to the next grassy area with scattered trees and a curved path beside the residential streets. Keep straight, then turn right toward the two-lane road with houses on both sides and move to the road. Proceed forward along the road, then turn left toward the dark-gray rectangular roof of the terraced house and continue to it. Continue ahead toward the parking lot, turn right to the black car parked in a row, move to the car, stop.	Move straight ahead toward the wide multi-lane road with white dashed markings beside the unpaved lot, continuing forward until you reach it. Stay on this line, then turn right and proceed straight toward the curved dirt racetrack surrounding the grassy field. Continue past the racetrack, then turn left and head straight toward the narrow waterway bordered by trees. Keep forward, then turn right and continue straight to the large industrial building with a gray roof and adjacent parking lot. From this building, turn left and move straight along the main road until the single white car in the traffic lane is ahead; approach it and stop.

Table 5: Instruction Examples from Different Datasets

Stage	Framework	Batch	LR	Epochs	Steps	#Historical Views	Group Size	KL Coeff	Sampling Temp	λ_1	λ_2	α	β	γ
SFT	LLaMA-Factory	80	1×10^{-4}	2	–	4	–	–	–	–	–	–	–	–
RFT	verl	96	1×10^{-6}	–	1500	4	5	0.001	1.0	1	1	1	0.1	0.1

Table 6: Hyperparameter settings for the SFT and RFT training stages.

Method	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
P1: Primary school student *P2: University student* *P3: Office worker* *P4: Community staff* *P5: Courier/Delivery worker* *P6: Urban tourist* *P7: Retired adult* *P8: Teacher/Educator* *P9: Community volunteer* *P10: Advanced navigation user*										
Random	0.65	0.85	1.05	0.69	0.75	0.52	0.64	0.91	0.72	0.96
Seq2Seq	1.19	1.76	1.05	1.43	1.40	0.92	0.93	1.40	1.60	1.47
CMA	4.46	4.73	4.32	4.47	4.62	3.84	3.50	6.14	5.26	4.93
Qwen2.5-VL-7B	1.55	1.88	1.69	1.83	1.40	2.01	1.14	2.13	2.16	0.96
Qwen2.5-VL-32B	2.14	3.34	2.85	3.55	2.42	3.44	2.72	3.34	3.54	2.43
Qwen3-VL-235B-A22B	4.34	4.79	5.01	5.56	5.00	4.99	4.00	7.84	6.64	3.39
LLaMA-3.2-11B-Vision	2.27	2.44	2.22	0.00	0.00	1.01	1.82	1.30	1.22	1.47
GPT-4o	4.16	4.00	4.00	5.33	4.46	4.76	3.57	5.78	4.76	4.71
π_0	4.76	6.00	4.64	5.21	6.56	5.05	3.57	5.65	4.81	5.80
Uni-NaVid	13.66	16.92	14.01	15.64	16.87	13.47	11.34	17.26	17.21	16.04
Qwen2.5-VL-7B SFT-only	33.79	43.30	39.09	39.26	41.16	38.53	38.10	44.50	39.26	43.23
Qwen2.5-VL-7B RFT-only	2.08	2.18	2.11	2.75	2.63	2.52	1.50	3.22	2.66	2.21
AirVLN-R1 (Ours)	**44.74**	**55.67**	**52.27**	**50.60**	**53.79**	**48.97**	**45.82**	**57.57**	**52.08**	**56.70**

Table 7: SR (%) of all models under each user persona on the test-unseen split.

Method	NE↓	SR↑	OSR↑	SPL↑
No-History	121.6	22.96	34.79	21.84
Last-K	56.8	45.1	56.15	43.96
Uniform-K	41.1	49.83	60.14	48.72
Progressive Interval Sampling (Ours)	**39.8**	**51.82**	**62.73**	**50.66**

Table 8: Ablation Study of Historical View Image Selection Strategies (test-unseen)

Method	NE↓	SR↑	OSR↑	SPL↑
Qwen2.5-VL-7B SFT-only	47.5	40.20	53.21	39.15
SFT + RFT (Full Reward, AirVLN-R1)	39.8	51.82	62.73	50.66
SFT + RFT (w/o Subgoal State Alignment)	46.1	46.00	60.58	44.90
SFT + RFT (w/o Stop Consistency)	44.2	47.65	61.02	46.53
SFT + RFT (w/o Format Reward)	40.7	51.22	62.08	50.06

Table 9: Ablation results of reward components in the RFT Stage (test-unseen)

λ_2 \ λ_1	0.5	1.0	2.0
0.5	50.83	51.46	51.32
1.0	51.20	**51.82**	51.41
2.0	50.42	50.95	50.63

Table 10: Sensitivity analysis of λ_1 and λ_2 (SR% on test-unseen).

Method	NE↓	SR↑	Test Device	GPU Memory Usage (GB)	Inference Latency (s/step)
Seq2Seq	N/A	1/200	RTX 4090 * 1	0.22	0.080
CMA	N/A	2/200	RTX 4090 * 1	0.27	0.075
Qwen2.5-VL-7B	108.4	8/200	RTX 4090 * 1	20.40	0.840
Qwen2.5-VL-32B	88.2	17/200	A100 80GB * 1	67.64	2.511
GPT-4o	73.1	34/200	Cloud	/	3.674
AirVLN-R1 (Ours)	70.6	53/200	RTX 4090 * 1	20.40	0.854

Note. Seq2Seq and CMA fail to generate valid trajectories in most real-world episodes; their NE values are therefore not statistically meaningful and reported as N/A.

Table 11: Comparison of performance, resource usage, and inference latency on real-world test.