REFINEFLY: FAILURE-AWARE POST-TRAINING FOR AERIAL VISION-LANGUAGE NAVIGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Unmanned aerial vehicle vision-language navigation (UAV-VLN) requires agents to translate visual observations and language instructions into reliable flight actions in complex environments. Although recent end-to-end UAV vision-language-action (UAV-VLA) policies reduce reliance on separately designed perception, planning, and control modules, their behavior-cloning objectives provide limited supervision for errors arising during closed-loop execution. Reinforcement learning (RL) offers a promising solution, while limited interaction budgets, long-tailed scene distributions, and policy drift complicate sustained learning. To this end, we propose RefineFly, a failure-aware RL post-training framework for end-to-end UAV-VLA policies. Built on token-level proximal policy optimization (PPO), RefineFly maintains dynamic failure memory within a two-stage scene curriculum to sustain learning from unresolved tasks as training shifts from empirical to balanced scene sampling. Throughout this process, stage-specific reference regularization constrains policy deviation. Experiments on the TravelUAV benchmark demonstrate that RefineFly outperforms all comparison methods across all three splits, improving success rate over AerialVLA by 3.12 to 8.37 percentage points with a total rollout budget of about 30% of the training-set size. Moreover, ablations reveal that the benefits of sustained failure learning vary across evaluation splits, while the two-stage scene curriculum improves overall performance, highlighting the importance of training distributions when learning from failures.

1 INTRODUCTION

Vision-language navigation (VLN) requires embodied agents to associate natural-language instructions with visual observations and execute trajectories toward semantic goals (Anderson et al., 2018; Krantz et al., 2020). Unmanned aerial vehicle (UAV) VLN extends this problem to large-scale three-dimensional (3D) environments, where an aerial agent searches for a language-described object while continuously adjusting forward motion, altitude, yaw, and termination decisions (Liu et al., 2023; Fan et al., 2023; Wang et al., 2025). Recent vision-language-action (VLA) models provide a promising end-to-end interface from onboard observations and instructions to executable controls, reducing reliance on separately engineered perception, planning, and control modules (Zitkovich et al., 2023; Kim et al., 2025; Xu et al., 2026).

However, behavior-cloned UAV-VLA policies receive limited supervision for errors arising during their own closed-loop execution. Existing UAV-VLN systems improve long-horizon navigation through waypoint prediction, hierarchical planners, memory mechanisms, or external perception modules (Wang et al., 2025; Zhang et al., 2025a; Ding et al., 2026; Ning et al., 2026; Jiang et al., 2025). Although end-to-end policies such as AerialVLA (Xu et al., 2026) generate flight actions directly, their training relies on expert demonstrations that provide limited coverage of error-induced states (Ross et al., 2011). Consequently, rapidly changing viewpoints, occlusion, and long trajectories can cause local errors to compound into collisions, navigation drift, premature landing, or timeouts. These failures expose diverse forms of unsuccessful behavior and provide feedback for further policy improvement. Accordingly, online reinforcement learning (RL) post-training offers a way to refine the pretrained policy using rewards from its own navigation outcomes.

Nevertheless, exploiting this feedback effectively requires careful organization of post-training. Specifically, long navigation rollouts involve repeated simulation and VLA inference, making ex-

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.

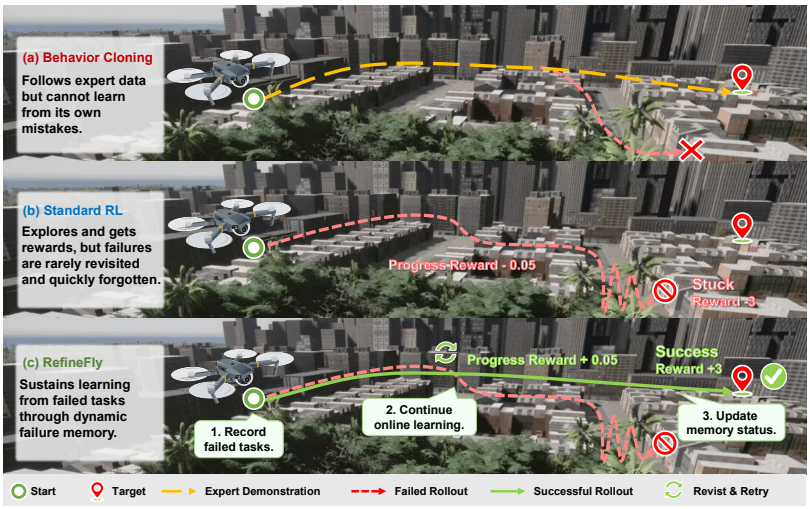


Figure 1: **Comparison of training paradigms for UAV-VLN.** RefineFly sustains online learning from unresolved tasks through dynamic failure memory, with fresh rollouts informing both policy updates and the memory lifecycle.

tensive online interaction costly. Within a limited budget, ordinary task sampling may provide few opportunities to continue learning from unresolved failures. Moreover, uneven scene distributions further restrict these opportunities in environments with relatively few training tasks (Lin et al., 2025; Gao et al., 2026). Consequently, post-training needs to reconcile adaptation under the existing task distribution with broader training coverage of rare scenes. Meanwhile, changes in the selected failure cases and scene distribution can shift the focus of policy updates, potentially weakening previously acquired navigation capabilities. These challenges motivate sustained failure learning together with stage-wise distribution adaptation and constraints on policy deviation.

To address these challenges, we propose RefineFly, a failure-aware online post-training framework for end-to-end UAV-VLA policies, as illustrated in Fig. 1. Built on token-level proximal policy optimization (PPO) (Schulman et al., 2017), RefineFly sustains learning from unresolved tasks through dynamic failure memory within a two-stage scene curriculum. As training shifts from empirical to balanced scene sampling, the framework expands learning opportunities for rare scenes while constraining policy deviation through reference regularization. This design supports continued learning from closed-loop failures and navigation generalization under a finite online interaction budget.

The main contributions of this paper are as follows.

- **RL Post-Training for End-to-End UAV-VLA.** We adapt RL post-training to end-to-end VLA policies for UAV-VLN and empirically study the influence of training distributions and sampling strategies in online post-training, together with the benefits and trade-offs of sustained learning from failure experiences.
- **Failure-Aware Online Post-Training Framework.** We propose RefineFly, which integrates dynamic failure memory with lifecycle management into a two-stage scene curriculum built on token-level PPO. This design sustains learning from failures across stage-specific training distributions while using reference regularization to mitigate policy drift.
- **Navigation Performance and Generalization.** Experiments demonstrate that RefineFly outperforms the compared baselines across all evaluation splits, with substantial gains on challenging long-horizon navigation tasks.

2 Related Work

2.1 VLN for UAVs

Early VLN methods established cross-modal instruction grounding and action prediction (Anderson et al., 2018; Krantz et al., 2020). AerialVLN (Liu et al., 2023) and AVDN (Fan et al., 2023) ex-

tended this setting to outdoor aerial navigation and dialog-conditioned target search, while UAV-ON (Xiao et al., 2025) and OpenFly (Gao et al., 2026) broadened it toward open-world object goals and larger-scale aerial datasets. To execute long-horizon flights, TravelUAV (Wang et al., 2025) predicts continuous waypoints, whereas CityNavAgent (Zhang et al., 2025c), SkyVLN (Li et al., 2025), TypeFly (Chen et al., 2025), and training-free VLM approaches (Hu et al., 2025) combine language reasoning with memory, trajectory generation, model-based control, or program synthesis. Moreover, NavFoM (Zhang et al., 2025a) and LongFly (Jiang et al., 2025) further improve generalization through large-scale pretraining and spatiotemporal modeling. These methods substantially improve semantic planning, while most retain explicit interfaces between high-level reasoning and low-level flight execution, through which prediction and control errors can accumulate.

2.2 VLA Policies for UAV-VLN

RT-2 (Zitkovich et al., 2023) and OpenVLA (Kim et al., 2025) show that robotic actions can be represented in language-compatible output spaces, allowing pretrained vision-language representations to support embodied control. This paradigm has been extended to fine-grained imitation, racing, and cognitive UAV control (Wang et al., 2026; Serpiva et al., 2025; Lykov et al., 2025). Furthermore, AerialVLA (Xu et al., 2026) maps onboard observations and linguistic prompts directly to continuous 3-degree-of-freedom (DoF) controls and an intrinsic landing decision, reducing reliance on oracle waypoints, external detectors, and separate landing modules. However, these UAV-VLA policies are trained mainly by supervised behavior cloning, which provides limited coverage of states induced by autonomous execution errors (Ross et al., 2011). Consequently, closed-loop failures motivate further policy improvement through online feedback.

2.3 RL Post-Training for Aerial VLN

Interactive RL post-training has been explored directly in language-conditioned UAV control. SuReAL (Blukis et al., 2019) combines supervised position prediction with RL-based continuous control and evaluates the resulting policy on a physical quadcopter. More recently, HTNav (Fan et al., 2026) integrates staged imitation and RL with tiered decision making for urban aerial VLN. OpenVLN (Lin et al., 2025) studies policy adaptation under data scarcity using value-based waypoint rewards and KL-regularized updates, while FlightGPT (Cai et al., 2025) combines supervised fine-tuning with reward-based refinement for goal accuracy, reasoning quality, and output compliance. These studies establish the value of interaction feedback, reward design, and staged optimization for improving aerial navigation. Building on this direction, our work studies how sustained learning from failed tasks interacts with scene sampling during end-to-end UAV-VLA post-training. Accordingly, RefineFly maintains dynamic failure memory throughout a two-stage scene curriculum to support continued policy improvement under uneven scene coverage.

3 Method

3.1 Overview

We propose RefineFly, a failure-aware online RL post-training framework for end-to-end VLA policies in UAV-VLN, as illustrated in Fig. 2. Using token-level PPO as the optimization backbone, RefineFly preserves the original autoregressive textual action interface. During online learning, the framework maintains a dynamic failure memory to sustain learning from unresolved tasks while retaining fresh task sampling from the underlying training distribution. Building on this process, a two-stage curriculum shifts training from the original long-tailed distribution toward a more balanced distribution, providing more training opportunities for tasks in rare scenes and enabling further learning from the associated failures. Throughout this process, policy updates are regularized against a reference policy frozen within each stage to mitigate policy drift as the failure memory and training distribution evolve.

3.2 Policy Interface and Optimization Backbone

Textual Action Interface. We consider end-to-end UAV-VLN control and adapt the UAV-VLA policy to RL fine-tuning through its existing autoregressive textual action interface. Specifically, at

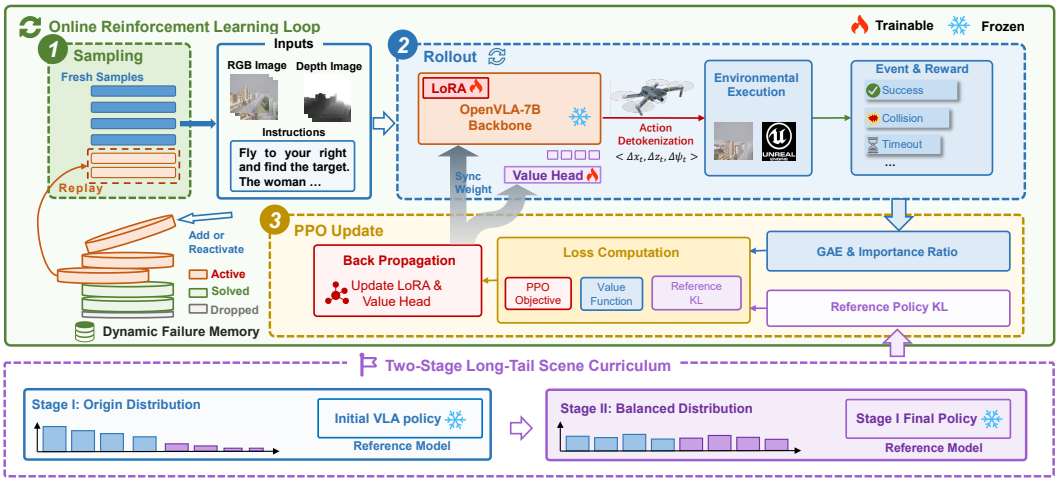


Figure 2: **Overview of RefineFly.** RefineFly sustains learning from unresolved tasks through dynamic failure memory within a two-stage curriculum that rebalances the training distribution across scenes. Token-level PPO updates the policy through the original autoregressive textual action interface, with updates regularized against a reference policy frozen within each stage.

time step t, the instruction x and visual observation o_t form the context $h_t = (x, o_t)$, from which the policy generates an action-token sequence $\mathbf{z}_t = (z_{t,1}, \ldots, z_{t,K_t})$ of length K_t according to

$$\pi_\theta(\mathbf{z}_t \mid h_t) = \prod_{k=1}^{K_t} \pi_\theta(z_{t,k} \mid h_t, z_{t,<k}). \tag{1}$$

Following AerialVLA (Xu et al., 2026), three uniformly quantized control tokens represent forward progression, vertical movement, and yaw adjustment, followed by an optional `LAND` token. A deterministic decoder then maps valid sequences to continuous commands $\mathbf{a}_t = \langle \Delta x_t, \Delta z_t, \Delta \psi_t \rangle$.

Variable-length action-token sequences require PPO updates to account for both valid token positions and executable command formats. Specifically, unmasked padding can contribute to the loss, and unnormalized token sums can overweight longer commands. Accordingly, we pad sequences to a maximum length K, apply a mask $m_{t,k} \in \{0, 1\}$ to select valid action-token positions, and normalize the loss by $M_t = \max(1, \sum_{k=1}^{K} m_{t,k})$. Meanwhile, the same action grammar constrains generation and likelihood computation. Any remaining parsing failure produces a safe no-op and receives an invalid-action penalty.

Token-Level PPO Backbone. Sequence-level PPO applies a shared clipping decision to the joint probability ratio of an autoregressive action sequence. Consequently, a probability change at one token position can activate clipping and suppress policy-gradient contributions from other positions. Accordingly, we adopt token-level PPO as the optimization backbone, applying probability ratios and clipping separately at each valid action-token position. Here, all valid tokens within a command share the action-level advantage $\widehat{A}_t$. For the rollout policy π_{old}, the token probability ratio is

$$\rho_{t,k}(\theta) = \frac{\pi_\theta(z_{t,k} \mid h_t, z_{t,<k})}{\pi_{\text{old}}(z_{t,k} \mid h_t, z_{t,<k})}. \tag{2}$$

To incorporate closed-loop navigation feedback into token-level updates, we compute the shared action-level advantage from rollout rewards and value estimates. During training, a value head $V_\phi(h_t)$ provides these estimates and is optimized using the standard clipped PPO value loss $\mathcal{L}_V$. With $\bar{\rho}_{t,k}(\theta) = \text{clip}(\rho_{t,k}(\theta), 1-\epsilon, 1+\epsilon)$, the masked policy objective is

$$J_{\text{PPO}}^{\text{token}}(\theta) = \mathbb{E}_t \left[\frac{1}{M_t} \sum_{k=1}^{K} m_{t,k} \min\left(\rho_{t,k}(\theta)\widehat{A}_t, \bar{\rho}_{t,k}(\theta)\widehat{A}_t \right) \right], \tag{3}$$

where ϵ is the clipping threshold. Appendix C reports an empirical comparison supporting this backbone choice. These interface and optimization adaptations enable direct RL fine-tuning of the pretrained VLA policy while preserving the original action interface.

3.3 FAILURE-AWARE CURRICULUM UNDER SCENE IMBALANCE

The optimization backbone learns from both successful and failed trajectories, while sustained improvement also depends on the training opportunities available across tasks and scenes. Although event-based penalties provide diverse failure feedback, ordinary task sampling may offer limited opportunities for further training on the associated tasks. Moreover, long-tailed scene distributions make such opportunities particularly scarce in rare scenes. To address these limitations, RefineFly sustains learning from unresolved tasks through a dynamic failure memory within a two-stage curriculum that shifts training toward a more balanced scene distribution.

Dynamic Failure Memory. Compared with the terminal success reward, event-based failure penalties provide feedback on a broader range of navigation outcomes. Specifically, collisions, stalled motion, persistent movement away from the target, premature landing, and timeouts reflect distinct forms of unsuccessful execution. To incorporate this diverse feedback into policy optimization, RefineFly integrates event-specific rewards and penalties with dense distance progress, defining the rollout reward as

$$r_t = \mathrm{clip}(\kappa_p(D_{t-1} - D_t), r_{\min}^{\mathrm{prog}}, r_{\max}^{\mathrm{prog}}) + \sum_{e \in \mathcal{E}} R_e \mathbb{I}[e_t = e], \tag{4}$$

where D_t denotes the Euclidean distance to the target, κ_p scales the progress reward, and $r_{\min}^{\mathrm{prog}}$ and $r_{\max}^{\mathrm{prog}}$ are its clipping bounds. The set $\mathcal{E} = \{\texttt{success}, \texttt{collision}, \texttt{stuck}, \texttt{away}, \texttt{early-stop}, \texttt{timeout}\}$ contains the navigation events, and R_e specifies the reward or penalty associated with event e. Standard generalized advantage estimation (GAE) (Schulman et al., 2016) uses these rewards and the value estimates $V_\phi(h_t)$ to compute the action-level advantages $\widehat{A}_t$ for token-level PPO.

To sustain learning from these diverse failure signals across rollouts, RefineFly maintains a dynamic failure memory $\mathcal{M}_u = \{\xi_i\}$ that evolves with online training. Specifically, each task record is represented as

$$\xi_i = (\omega_i, c_i, f_i, \sigma_i, n_i), \tag{5}$$

where the entries denote the task initialization, scene, failure type, status, and unsuccessful retry count, respectively. Initially, the memory contains failed tasks collected with the initial policy. When an unresolved record is selected, the current policy generates a complete new trajectory from ω_i. After each attempt, a successful task is marked as solved and excluded from further selection, while n_i is incremented after a failed attempt. To limit repeated unsuccessful trials, the record is marked as dropped and excluded when n_i reaches $N_{\max}$. Meanwhile, failures from fresh sampling enter the memory as unresolved records, including previously solved or dropped tasks that fail again.

Two-Stage Scene Curriculum. To sustain learning from failures while increasing training opportunities for rare scenes, we organize online post-training into a two-stage scene curriculum. Specifically, Stage I follows the empirical scene distribution, while Stage II continues from the final Stage I policy with balanced scene sampling. The overall scene allocations are defined as

$$P_1(c) = \frac{N_c}{\sum_{c' \in \mathcal{C}} N_{c'}}, \qquad P_2(c) = \frac{1}{|\mathcal{C}|}, \tag{6}$$

where $\mathcal{C}$ denotes the training scenes, and N_c is the number of tasks in scene c. For scenes with insufficient tasks, repeated sampling fills the Stage II quota.

Both stages reserve a fixed fraction of rollouts for unresolved tasks in the dynamic failure memory while preserving the prescribed scene distribution. Specifically, $q_{s,u}$ denotes the task sampling distribution at iteration u of stage $s \in \{1, 2\}$, with scene distribution P_s. Within this allocation, a fraction η of rollouts is assigned to unresolved tasks in $\mathcal{M}_u$, and the remaining rollouts use fresh task sampling. The current policy executes the selected tasks, and navigation outcomes update the memory through the same lifecycle. Under the rebalanced distribution of Stage II, tasks in rare scenes receive more training opportunities, allowing additional failures to enter the memory and support subsequent learning. Appendix A.1 specifies the scene quotas and sampling procedure.

3.4 Reference-Constrained Policy Updates

Although the failure-aware curriculum strengthens learning from failures and aims to improve generalization under scene imbalance, changes in the training distribution may induce policy drift and weaken previously acquired navigation capabilities. To limit this drift, we adopt reference-policy Kullback-Leibler (KL) regularization. Specifically, the reference policy $\pi_{\text{ref}}^{(s)}$ remains frozen throughout each training stage $s \in \{1, 2\}$. In the two-stage curriculum, Stage I uses the initial VLA policy as $\pi_{\text{ref}}^{(1)}$ and Stage II uses the final Stage I policy as $\pi_{\text{ref}}^{(2)}$ to anchor adaptation to the rebalanced scene distribution.

The reference penalty measures policy deviation at valid action-token positions. Specifically, let $p_{\theta,t,k}(\cdot) = \pi_\theta(\cdot \mid h_t, z_{t,<k})$ and $p_{\text{ref},t,k}^{(s)}(\cdot) = \pi_{\text{ref}}^{(s)}(\cdot \mid h_t, z_{t,<k})$ denote the corresponding token distributions. Using the mask and normalization defined in the optimization backbone, the reference loss is given by

$$\mathcal{L}_{\text{KL}}^{(s)} = \mathbb{E}_t \left[\frac{1}{M_t} \sum_{k=1}^{K} m_{t,k} D_{\text{KL}} \Big(p_{\theta,t,k} \,\|\, p_{\text{ref},t,k}^{(s)} \Big) \right]. \tag{7}$$

The final objective combines the policy, value, and reference terms using trajectories collected under the failure-aware curriculum. Specifically, with task initializations sampled according to $q_{s,u}$, the loss is given by

$$\mathcal{L}(\theta, \phi) = -J_{\text{PPO}}^{\text{token}}(\theta) + c_v \mathcal{L}_V(\phi) + \beta \mathcal{L}_{\text{KL}}^{(s)}, \tag{8}$$

where c_v weights value regression and β controls the reference penalty. Thus, the scene curriculum specifies the overall scene distribution, while token-level PPO and reference regularization govern policy updates at valid action-token positions.

4 Experiments

4.1 Experimental Setup

Dataset. We evaluate RefineFly on the TravelUAV dataset (Wang et al., 2025) under the *UAV-Need-Help* setting adopted by AerialVLA. Specifically, we use 7922 training trajectories and evaluate on 1418 Seen, 958 Unseen Map, and 629 Unseen Object trajectories. Following the benchmark settings, trajectories shorter than 250 meters are classified as easy, and the remainder as hard. We report results on the full test set and both difficulty subsets for each split. The training set exhibits a long-tailed scene distribution, as detailed in Appendix A.1.

Metrics. We evaluate navigation accuracy, task completion, and path efficiency using navigation error (NE), success rate (SR), oracle success rate (OSR), and success weighted by path length (SPL). Specifically, NE measures the final Euclidean distance to the destination. SR measures successful termination within the target region, including a correct `LAND` output or near-zero movement for 10 consecutive steps. OSR records whether the executed trajectory ever enters that region, while SPL accounts for both task completion and path efficiency. Lower NE and higher SR, OSR, and SPL indicate better performance.

Implementation Details. RefineFly post-trains the AerialVLA initialization through online interaction while preserving its action interface. Specifically, AerialVLA combines the OpenVLA-7B (Kim et al., 2025) backbone with a LoRA (Hu et al., 2022) adapter, and post-training optimizes the adapter together with an added value head using simulator-derived rewards. Counting both fresh and memory-selected task executions, the total rollout budget is approximately 30% of the training-set size. Training uses 8 $\times$ NVIDIA A100 (80GB) GPUs and takes about 21 hours. We implement RefineFly using RLinf (Yu et al., 2025), AirSim (Shah et al., 2017), the TravelUAV environment, and the AerialVLA action-token pipeline. Appendix A.2 provides the remaining training configurations.

Baselines. We compare RefineFly with heuristic controls, task-specific UAV-VLN methods, generalist navigation models, and an end-to-end VLA baseline, which either rely on fixed policies, specialized planning modules, or behavior-cloning-based training. For all baselines, we report results from the corresponding publications on the same evaluation splits.

Table 1: Comparison on the Test Seen set. RefineFly reports mean and standard deviation over three evaluation seeds. NE is in meters, and the other metrics are percentages. Bold and underline indicate the best and second-best results, respectively. Shading marks RefineFly. Human performance is for reference only.

Method	Full				Easy				Hard			
	NE↓	SR↑	OSR↑	SPL↑	NE↓	SR↑	OSR↑	SPL↑	NE↓	SR↑	OSR↑	SPL↑
Human	14.15	94.51	94.51	77.84	11.68	95.44	95.44	76.19	17.16	93.37	93.37	79.85
Random Action	222.20	0.14	0.21	0.07	142.07	0.26	0.39	0.13	320.12	0.00	0.00	0.00
Fixed Action	188.61	2.27	8.16	1.40	121.36	3.48	11.48	2.14	270.69	0.79	4.09	0.49
CMA	135.73	8.37	18.72	7.90	84.89	11.48	24.52	10.68	197.77	4.57	11.65	4.51
TravelUAV-DA	98.66	17.45	48.87	15.76	66.40	20.26	51.23	18.10	138.04	14.02	45.98	12.90
NavFoM	93.05	29.17	49.24	25.03	58.98	32.91	53.16	27.87	143.83	23.58	43.40	20.80
LongFly	<u>60.02</u>	36.39	**65.87**	31.07	<u>38.10</u>	38.52	**71.90**	31.24	<u>85.20</u>	33.94	<u>58.94</u>	30.88
AerialVLA	65.88	<u>47.96</u>	57.69	<u>38.54</u>	43.76	<u>49.30</u>	61.30	<u>37.14</u>	93.16	<u>46.30</u>	53.23	<u>40.26</u>
RefineFly (Ours)	**54.96** ± 1.19	**56.33** ± 0.15	<u>65.40</u> ± 0.80	**45.98** ± 1.13	**37.60** ± 1.31	**56.28** ± 0.48	<u>67.39</u> ± 0.98	**42.91** ± 0.58	**76.36** ± 1.07	**56.38** ± 0.88	**62.94** ± 0.59	**49.75** ± 1.80

Table 2: Comparison on the Test Unseen Map set. RefineFly reports mean and standard deviation over three evaluation seeds. NE is in meters, and the other metrics are percentages. Bold and underline indicate the best and second-best results, respectively. Shading marks RefineFly.

Method	Full				Easy				Hard			
	NE↓	SR↑	OSR↑	SPL↑	NE↓	SR↑	OSR↑	SPL↑	NE↓	SR↑	OSR↑	SPL↑
Random Action	202.98	0.00	0.00	0.00	158.46	0.00	0.00	0.00	265.88	0.00	0.00	0.00
Fixed Action	180.47	0.52	2.61	0.39	132.89	0.89	4.28	0.67	247.72	0.00	0.25	0.00
CMA	141.68	2.30	10.02	2.16	102.29	3.57	14.26	3.33	197.35	0.50	4.03	0.50
TravelUAV	138.80	4.18	20.77	3.84	102.94	4.63	22.82	4.24	189.46	3.53	17.88	3.28
NavFoM	125.10	6.30	18.95	5.68	102.41	6.77	20.07	6.04	170.58	5.36	15.71	4.97
LongFly	108.32	11.27	30.27	9.32	78.56	12.96	34.31	10.32	148.10	9.02	24.88	7.98
AerialVLA	<u>67.42</u>	<u>37.58</u>	<u>52.92</u>	<u>28.22</u>	<u>44.99</u>	<u>41.89</u>	<u>58.47</u>	<u>29.72</u>	<u>99.11</u>	<u>31.49</u>	<u>45.09</u>	<u>26.11</u>
RefineFly (Ours)	**58.88** ± 2.09	**42.97** ± 1.76	**60.09** ± 1.01	**31.83** ± 1.25	**44.02** ± 0.99	**46.88** ± 1.24	**63.04** ± 0.37	**32.44** ± 0.72	**79.88** ± 4.95	**37.45** ± 2.65	**55.92** ± 2.27	**30.98** ± 2.05

- **Heuristic Methods.** Random Action samples controls without using visual or language inputs, while Fixed Action executes predefined commands.
- **Task-Specific UAV-VLN Models.** CMA (Anderson et al., 2018) is a recurrent cross-modal navigation baseline, while TravelUAV (Wang et al., 2025) combines multimodal features with hierarchical trajectory decoders and an external target detector. TravelUAV-DA extends TravelUAV by aggregating additional corrective trajectories.
- **Generalist Navigation Models.** NavFoM (Zhang et al., 2025a) uses a navigation foundation model with a dedicated trajectory-planning head, and LongFly (Jiang et al., 2025) introduces explicit spatiotemporal modeling for long-horizon UAV navigation.
- **End-to-End VLA Baseline.** AerialVLA (Xu et al., 2026) autoregressively generates action tokens for UAV control, and its checkpoint initializes RefineFly, making their comparison a direct evaluation of RL post-training.

4.2 Performance Comparison

Performance on Seen Environments. As shown in Table 1, RefineFly improves closed-loop navigation over AerialVLA on the test seen set, with larger gains on Hard trajectories across all four metrics. Specifically, Full-set SR increases by 8.37 percentage points, and the gain reaches 10.08 points on the Hard subset. Moreover, Hard-set SPL improves by 9.49 points. The stronger gains on long trajectories are consistent with dynamic failure memory providing repeated opportunities to refine behavior on failed tasks, where local execution errors can prevent eventual completion.

Table 3: Comparison on the Test Unseen Object set. RefineFly reports mean and standard deviation over three evaluation seeds. NE is in meters, and the other metrics are percentages. Bold and underline indicate the best and second-best results, respectively. Shading marks RefineFly.

Method	Full				Easy				Hard			
	NE↓	SR↑	OSR↑	SPL↑	NE↓	SR↑	OSR↑	SPL↑	NE↓	SR↑	OSR↑	SPL↑
Random Action	260.14	0.16	0.16	0.16	174.10	0.48	0.48	0.48	302.96	0.00	0.00	0.00
Fixed Action	212.84	3.66	9.54	2.16	151.66	6.70	13.88	3.72	243.29	2.14	7.38	1.38
CMA	155.79	9.06	16.06	8.68	102.92	14.83	22.49	13.90	182.09	6.19	12.86	6.08
TravelUAV	118.11	22.42	46.90	20.51	86.12	24.40	49.28	22.03	134.03	21.43	45.71	19.75
NavFoM	108.04	29.83	47.99	27.20	70.51	32.54	50.72	29.54	133.01	28.03	46.18	25.64
LongFly	66.74	43.87	64.56	38.39	54.84	38.01	56.84	31.36	**57.07**	50.25	**74.16**	45.27
AerialVLA	<u>61.45</u>	<u>56.60</u>	<u>64.86</u>	<u>46.61</u>	<u>45.72</u>	<u>56.94</u>	<u>64.11</u>	<u>43.76</u>	69.27	<u>56.43</u>	65.24	<u>48.03</u>
RefineFly (Ours)	**53.42** ± 0.97	**59.72** ± 0.81	**68.31** ± 1.66	**51.34** ± 0.77	**35.49** ± 1.94	**63.80** ± 1.20	**72.09** ± 2.41	**50.97** ± 1.58	<u>62.34</u> ± 0.52	**57.70** ± 1.45	<u>66.43</u> ± 1.95	**51.52** ± 1.02

Table 4: Incremental ablation of RefineFly using evaluation seed 1. ID 1 uses token-level PPO, and ID 4 is the full framework. Checkmarks denote enabled components. Values are full-split SR (%) and its unweighted mean across splits, with absolute changes from AerialVLA in parentheses. Bold and underline mark the best and second-best results. Shading marks RefineFly.

	Framework Components			Success Rate (%)			
ID	Failure Memory	Reference KL	Two-Stage Curriculum	Seen	Unseen Map	Unseen Object	Avg.
–	–	–	–	47.96	37.58	56.60	47.38
1	×	×	×	48.17 (+0.21 ↑)	<u>42.90</u> (+5.32 ↑)	56.44 (−0.16 ↓)	49.17 (+1.79 ↑)
2	✓	×	×	<u>56.21</u> (+8.25 ↑)	31.21 (−6.37 ↓)	59.78 (+3.18 ↑)	49.07 (+1.69 ↑)
3	✓	✓	×	55.01 (+7.05 ↑)	37.47 (−0.11 ↓)	**62.32** (+5.72 ↑)	<u>51.60</u> (+4.22 ↑)
4	✓	✓	✓	**56.28** (+8.32 ↑)	**44.89** (+7.31 ↑)	<u>60.41</u> (+3.81 ↑)	**53.86** (+6.48 ↑)

Generalization to Unseen Maps. As shown in Table 2, RefineFly improves navigation over AerialVLA on unseen maps, increasing Full-set SR by 5.39 percentage points. Unlike the Easy subset, where NE changes only slightly, Hard-set NE decreases by 19.23 m, suggesting improved navigation over long routes, where execution errors can accumulate. Moreover, the concurrent gains in OSR and SR show that RefineFly improves both target-region reachability and successful completion on unseen maps. These gains are consistent with scene rebalancing giving less frequent scenes more training opportunities, while reference regularization limits policy deviation during continued failure learning.

Generalization to Unseen Objects. As shown in Table 3, RefineFly achieves the best Full-set results across all four metrics. Specifically, SR reaches 59.72%, and SPL improves by 4.73 percentage points over AerialVLA. On Hard trajectories, LongFly achieves lower NE and higher OSR, whereas RefineFly obtains higher SR and SPL. The higher completion metrics are consistent with sustained learning from execution feedback, where penalties for collisions and premature landing may help refine navigation and termination behavior for unfamiliar objects.

4.3 Ablation Study

We conduct ablation studies to investigate how sustained failure learning interacts with reference regularization and scene sampling across all test splits. Specifically, all ablations use evaluation seed 1. Table 4 incrementally adds components to the token-level PPO backbone, while Table 5 compares scene sampling schedules. Appendices B and C analyze failure-memory outcomes and compare token-level and sequence-level PPO.

Effect of RL Post-Training and Dynamic Failure Memory. As shown in Table 4, the average performance improvement gained from using only token-level PPO is relatively limited, and the gains are uneven across different evaluation splits. Compared with AerialVLA, the token-level PPO baseline (ID 1) leaves Seen SR nearly unchanged, improves Unseen Map SR by 5.32 percentage

Table 5: Scene sampling comparison using full-split SR (%) with evaluation seed 1. Uniform Sampling uses equal scene quotas in one stage, while Original Distribution retains empirical proportions across two stages. RefineFly shifts from empirical to balanced sampling. Bold and underline mark the best and second-best results. Shading marks RefineFly.

Sampling Strategy	Seen	Unseen Map	Unseen Object	Avg.
Uniform Sampling	50.71	<u>41.23</u>	52.31	48.08
Original Distribution	<u>56.13</u>	40.92	<u>59.14</u>	<u>52.06</u>
Two-Stage Curriculum	**56.28**	**44.89**	**60.41**	**53.86**

points, and reduces Unseen Object SR by 0.16 points. The average gain of 1.79 percentage points supports policy refinement through online feedback, while the uneven changes motivate examining how failure-learning opportunities are distributed during post-training.

Furthermore, dynamic failure memory improves performance on some evaluation splits while exposing a generalization trade-off. Specifically, ID 2 increases Seen and Unseen Object SR by 8.04 and 3.34 points relative to ID 1, whereas Unseen Map SR decreases by 11.69 points. Repeated current-policy rollouts provide fresh feedback on unresolved tasks, supporting further adaptation to previously unsuccessful navigation conditions. However, the unchanged empirical scene allocation does not increase the relative training share of rare scenes. These results suggest that continued task-specific practice can improve navigation in familiar scenes while leaving transfer to unseen maps limited, motivating further reference constraints and scene rebalancing.

Effect of Reference-Policy KL Regularization. Reference-policy KL regularization partially offsets the generalization loss associated with failure-focused training. As can be seen in Table 4, ID 3 improves Unseen Map SR by 6.26 percentage points and average SR by 2.53 points relative to ID 2, while Seen SR decreases by 1.20 points. Anchoring updates to the stage-initial policy discourages large policy deviations during repeated training on unresolved tasks. The improvement on unseen maps and reduction on seen environments are consistent with tempering task-specific adaptation to support transfer. However, Unseen Map SR remains below the token-level PPO baseline, leaving an unresolved trade-off for the scene curriculum to address.

Effect of the Two-Stage Curriculum. The two-stage curriculum further improves average performance by changing the scene allocation under which failure learning continues. As shown in Table 4, ID 4 improves Unseen Map and average SR by 7.42 and 2.26 percentage points relative to ID 3, with a 1.91-point decrease in Unseen Object SR. Balanced sampling in Stage II gives rare scenes more opportunities for both fresh interaction and continued failure learning, offering a plausible explanation for the gain on unseen maps. Furthermore, as illustrated in Table 5, RefineFly outperforms uniform sampling and the two-stage original-distribution baseline on all three splits. Relative to the latter, Unseen Map and average SR increase by 3.97 and 1.80 points, respectively. These results support a strategy of initial optimization based on empirical distributions followed by scene rebalancing, suggesting that sustained failure learning benefits from how training opportunities are allocated across stages.

5 Conclusions

We present RefineFly, a failure-aware post-training framework that sustains learning from closed-loop failures through dynamic failure memory within a two-stage scene curriculum built on token-level PPO. With a total rollout budget of about 30% of the training-set size, RefineFly outperforms all comparison methods across all three splits, improving the SR of AerialVLA by 3.12 to 8.37 percentage points. Beyond these gains, our findings indicate that failure experience provides valuable opportunities for policy improvement, while its benefits vary across evaluation splits under an imbalanced training distribution. Moreover, shifting from empirical to balanced scene sampling improves generalization and outperforms uniform sampling from the outset, highlighting the importance of how training opportunities are distributed when learning from failures. Extending this evaluation to additional benchmarks and real-world flights is an important direction for future work.

AI USE STATEMENT

Generative AI tools were used solely for language polishing of the manuscript.

REPRODUCIBILITY STATEMENT

The method and optimization objectives are described in Section 3, and the experimental setup and evaluation metrics are specified in Section 4.1. Appendix A provides the dataset distribution, training and evaluation parameters, and evaluation seeds. Appendices B and C provide further details on dynamic failure memory and the PPO comparison.

REFERENCES

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 *Proc. CVPR*, pp. 3674–3683, 2018.

Valts Blukis, Yannick Terme, Eyvind Niklasson, Ross A. Knepper, and Yoav Artzi. Learning to map natural language instructions to physical quadcopter control using simulated flight. In *Proc. CoRL*, volume 100, pp. 1415–1438, 2019.

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 *Proc. EMNLP*, pp. 6670–6687, 2025.

Guojun Chen, Xiaojing Yu, Neiwen Ling, and Lin Zhong. Typefly: Low-latency drone planning with large language models. *IEEE Trans. Mob. Comput.*, pp. 9068—-9079, 2025.

Xichen Ding, Jianzhe Gao, Cong Pan, Wenguan Wang, and Jie Qin. History-enhanced two-stage transformer for aerial vision-and-language navigation. In *Proc. AAAI*, volume 40, pp. 18225–18233, 2026.

Chengjie Fan, Cong Pan, Zijian Liu, Ningzhong Liu, and Jie Qin. HTNav: A hybrid navigation framework with tiered structure for urban aerial vision-and-language navigation. In *Proc. CVPR*, pp. 10976–10985, 2026.

Yue Fan, Winson Chen, Tongzhou Jiang, Chun Zhou, Yi Zhang, and Xin Wang. Aerial vision-and-dialog navigation. In *Proc. ACL*, pp. 3043–3061, 2023.

Mohamed Amine Ferrag, Abderrahmane Lakas, and Merouane Debbah. UAVBench: An open benchmark dataset for autonomous and agentic AI UAV systems via LLM-generated flight scenarios, 2025.

Yunpeng Gao, Chenhui Li, Zhongrui You, Junli Liu, Zhen Li, Pengan Chen, Qizhi Chen, Zhonghan Tang, Liansheng Wang, Penghui Yang, et al. Openfly: A comprehensive platform for aerial vision-language navigation. In *Proc. ICLR*, 2026.

Chih Yao Hu, Yang-Sen Lin, Yuna Lee, Chih-Hai Su, Jie-Ying Lee, Shr-Ruei Tsai, Chin-Yang Lin, Kuan-Wen Chen, Tsung-Wei Ke, and Yu-Lun Liu. See, point, fly: A learning-free VLM framework for universal unmanned aerial navigation. In *Proc. CoRL*, pp. 4697–4708, 2025.

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 *Proc. ICLR*, 2022.

Wen Jiang, Li Wang, Kangyao Huang, Wei Fan, Jinyuan Liu, Shaoyu Liu, Hongwei Duan, Bin Xu, and Xiangyang Ji. Longfly: Long-horizon UAV vision-and-language navigation with spatiotemporal context integration, 2025.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. In *Proc. CoRL*, pp. 2679–2713, 2025.

Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (eds.), *Proc. ECCV*, pp. 104–120, 2020.

Tianshun Li, Tianyi Huai, Zhen Li, Yichun Gao, Haoang Li, and Xinhu Zheng. Skyvln: Vision-and-language navigation and NMPC control for UAVs in urban environments. In *Proc. IROS*, pp. 17199–17206, 2025.

Peican Lin, Gan Sun, Chenxi Liu, Fazeng Li, Weihong Ren, and Yang Cong. Openvln: Open-world aerial vision-language navigation, 2025.

Shubo Liu, Hongsheng Zhang, Yuankai Qi, Peng Wang, Yanning Zhang, and Qi Wu. Aerialvln: Vision-and-language navigation for UAVs. In *Proc. ICCV*, pp. 15338–15348, 2023.

Artem Lykov, Valerii Serpiva, Muhammad Haris Khan, Oleg Sautenkov, Artyom Myshlyaev, Grik Tadevosyan, Yasheerah Yaqoot, and Dzmitry Tsetserukou. Cognitivedrone: A VLA model and evaluation benchmark for real-time cognitive task solving and reasoning in UAVs, 2025.

Yuwei Ning, Ganlong Zhao, Yipeng Qin, Si Liu, Yang Liu, Liang Lin, and Guanbin Li. Lookaside-vln: Direction-aware aerial vision-and-language navigation. In *Proc. CVPR*, pp. 32441–32450, 2026.

Stéphane Ross, Geoffrey J. Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proc. AISTATS*, pp. 627–635, 2011.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *Proc. ICLR*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

Valerii Serpiva, Artem Lykov, Artyom Myshlyaev, Muhammad Haris Khan, Ali Alridha Abdulkarim, Oleg Sautenkov, and Dzmitry Tsetserukou. Racevla: VLA-based racing drone navigation with human-like behaviour, 2025.

Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and service robotics: Results of the 11th international conference*, pp. 621–635, 2017.

Xiangyu Wang, Donglin Yang, Hohin Kwan, Jinyu Chen, Hongsheng Li, Yue Liao, Si Liu, et al. Towards realistic UAV vision-language navigation: Platform, benchmark, and methodology. In *Proc. ICLR*, pp. 7292–7310, 2025.

Xiangyu Wang, Donglin Yang, Yue Liao, Wenhao Zheng, Bin Dai, Hongsheng Li, Si Liu, et al. Uav-flow colosseo: A real-world benchmark for flying-on-a-word UAV imitation learning. In *Proc. NeurIPS*, volume 38, 2026.

Jianqiang Xiao, Yuexuan Sun, Yixin Shao, Boxi Gan, Rongqiang Liu, Yanjin Wu, Weili Guan, and Xiang Deng. UAV-ON: A benchmark for open-world object goal navigation with aerial agents. In *Proc. ACM MM*, pp. 13023–13029, 2025.

Peng Xu, Zhengnan Deng, Jiayan Deng, Zonghua Gu, and Shaohua Wan. AerialVLA: A vision-language-action model for UAV navigation via minimalist end-to-end control, 2026.

Chao Yu, Yuanqing Wang, Zhen Guo, Hao Lin, Si Xu, Hongzhi Zang, Quanlu Zhang, Yongji Wu, Chunyang Zhu, Junhao Hu, et al. Rlinf: Flexible and efficient large-scale reinforcement learning via macro-to-micro flow transformation, 2025.

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, 2025a.

Lingfeng Zhang, Yuchen Zhang, Hongsheng Li, Haoxiang Fu, Yingbo Tang, Hangjun Ye, Long Chen, Xiaojun Liang, Xiaoshuai Hao, and Wenbo Ding. Is your VLM sky-ready? A comprehensive spatial intelligence benchmark for UAV navigation, 2025b.

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 *Proc. ACL*, pp. 31292–31309, 2025c.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong T. Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. RT-2: vision-language-action models transfer web knowledge to robotic control. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Proc. CoRL*, pp. 2165–2183, 2023.

APPENDIX

This supplementary material details the scene distribution, training configuration, dynamic failure memory, and additional analyses of RefineFly. Section A specifies the data and optimization settings. Section B describes navigation events, memory outcomes, and paired task-level performance changes. Section C compares token-level and sequence-level PPO, while Section D presents a representative navigation case.

A DETAILED EXPERIMENTAL SETTINGS

This section reports the training-set composition and optimization configuration used for RefineFly.

A.1 DATASET

Benchmarks for Aerial Agents. Existing benchmarks for aerial agents and UAV-VLN cover complementary levels of perception, reasoning, and control. AerialVLN (Liu et al., 2023) and AVDN (Fan et al., 2023) established outdoor instruction-following and dialog-conditioned navigation, while UAV-ON (Xiao et al., 2025) extended evaluation to open-world object-goal search. More recent benchmarks emphasize either data scale or high-level cognition, with OpenFly (Gao et al., 2026) providing diverse trajectories based on predefined flight-action primitives, SpatialSky-Bench (Zhang et al., 2025b) assessing aerial spatial intelligence, and UAVBench (Ferrag et al., 2025) evaluating mission-level reasoning.

Our study requires closed-loop interaction with executable flight actions and explicit navigation feedback. Accordingly, we adopt TravelUAV (Wang et al., 2025), which provides target-oriented navigation with continuous trajectories and evaluation splits covering seen scenes, unseen maps, and unseen objects. Moreover, TravelUAV provides the training and evaluation setting for AerialVLA (Xu et al., 2026), whose end-to-end VLA policy maps onboard observations and linguistic prompts to continuous 3-DoF controls and an intrinsic landing decision. This alignment supports evaluating online post-training in the environment and action space used by the behavior-cloned initialization.

Training-Set Distribution. The TravelUAV training set exhibits a pronounced imbalance across its 19 scenes. Specifically, Fig. 3 reports 7922 training samples, of which 7611 belong to 11 common scenes and account for 96.07% of the total. The remaining 311 samples span 8 rare scenes and account for only 3.93%. Furthermore, the largest scene contains 1545 samples, while the rare-scene counts range from 17 to 59. We define the rare-scene set as

$$\mathcal{R} = \{c \in \mathcal{C} \mid N_c < 100\}, \tag{9}$$

where N_c denotes the number of training samples associated with scene c.

Two-Stage Scene Sampling. The two-stage curriculum changes the overall scene allocation while retaining failure-memory sampling within each stage. Specifically, Stage I follows the empirical scene distribution shown in Fig. 3, while Stage II assigns an equal quota of 100 selections to each scene. Within these scene allocations, a fraction η of rollouts is assigned to unresolved tasks in dynamic failure memory, and the remainder uses fresh task sampling. For fresh sampling, tasks are selected without replacement while unused tasks remain available, followed by sampling with replacement when needed to fill the allocation. Consequently, a complete Stage II quota cycle contains 1900 selections, including 800 from rare scenes. The rare-scene share thus increases from 3.93% under the empirical distribution to 42.11%, providing more opportunities for both fresh interaction and continued learning from failures in these scenes.

A.2 TRAINING AND EVALUATION PARAMETERS

RefineFly uses the same optimizer, rollout, and reward configurations in both curriculum stages. Specifically, online post-training continues to optimize the existing adapter with its original LoRA configuration, while the scene distribution and frozen reference policy change between stages as described in the main text. Table 6 summarizes the training and evaluation parameters.

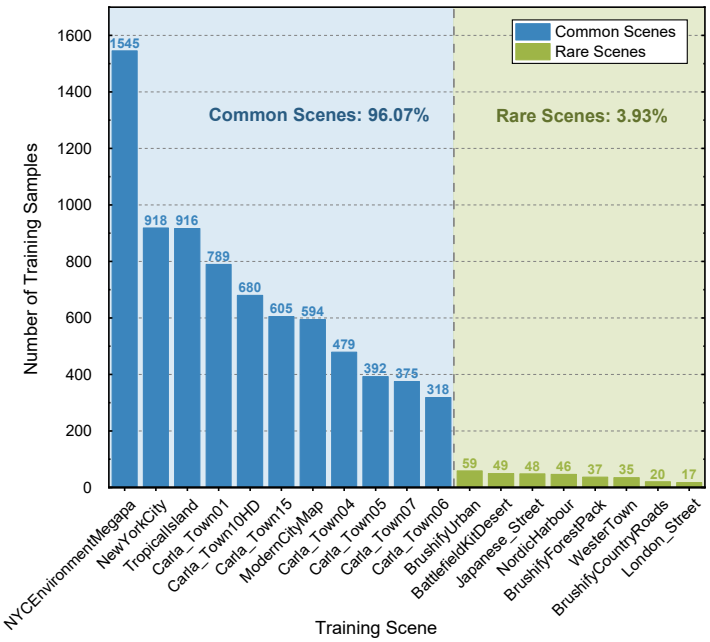


Figure 3: Training sample counts across the 19 TravelUAV scenes used by the *UAV-Need-Help* task.

The online rollout budget counts both fresh task executions and executions selected from dynamic failure memory. Across both stages, fresh task executions amount to approximately 25% of the training-set size. With memory-selected executions accounting for $\eta = 0.2$ of all sampled episodes, the total is $0.25/(1-\eta) = 31.25\%$ of the training-set size, reported as approximately 30%. This ratio measures task executions relative to the number of training tasks, including repeated executions.

The main comparison reports RefineFly performance over three evaluation seeds, while ablations use a single evaluation seed. Specifically, the main results give the mean and standard deviation for seeds 1, 42, and 12345. The evaluated post-training configurations in the ablations and corresponding supplementary analyses use seed 1.

B Dynamic Failure Memory Details

This section describes the navigation feedback and observed outcomes associated with dynamic failure memory. Specifically, Section B.1 defines the events, Section B.2 reports their frequencies, and Section B.3 analyzes the memory lifecycle and task-level changes in evaluation performance.

B.1 Event Types

Navigation events distinguish successful completion from different forms of unsuccessful execution and determine the associated reward feedback.

Success. A success event occurs when the UAV is within 20 meters of the target object and either executes `LAND` or exhibits near-zero movement for 10 consecutive time steps.

Collision. A collision event occurs when the simulator reports a physical collision or depth observations indicate that the UAV is too close to an obstacle.

Stuck. A stuck event occurs when the UAV exhibits near-zero movement for 10 consecutive steps outside the target region.

Continuous away. A continuous-away event occurs when the distance to the target object increases for 10 consecutive steps.

Table 6: Training and evaluation parameters for RefineFly.

Parameter	Value
Model and Optimization	
LoRA rank r	64
LoRA scaling α	128
LoRA dropout	0.05
Policy learning rate	5×10^{-5}
Value head learning rate	1×10^{-4}
Optimizer	AdamW
Weight decay	0.03
Gradient norm clipping	1.0
Learning rate schedule	Cosine
Warm-up steps	10
Schedule decay factor	0.01
Rollout and Sampling	
Stage I rollout epochs	100
Stage II rollout epochs	100
Global batch size	320
Mini-batch size	8
Sampling temperature	0.6
Maximum action steps	200
AirSim `ClockSpeed`	10
PPO and Dynamic Failure Memory	
Discount factor γ	0.995
GAE parameter λ	0.97
PPO clipping threshold ϵ	0.2
Value loss coefficient c_v	0.1
KL coefficient β	0.03
Memory sampling ratio η	0.2
Maximum unsuccessful retries $N_{\max}$	2
Reward Configuration	
Progress reward	0.02 per meter, capped at 0.06 per step
Distance increase penalty	0.02 per meter, capped at 0.06 in magnitude
Successful landing reward	3.0
Collision penalty	-2.0
Timeout penalty	-1.0
Early stop penalty	-0.8
Stuck penalty	-1.2
Continuous away penalty	-1.5
Invalid action penalty	-0.5

Early stop. An early-stop event occurs when the policy issues `LAND` outside the target region.

Timeout. A timeout event occurs when an episode reaches the 200-step horizon without success.

Invalid actions receive a separate format-related penalty. Specifically, an output sequence that cannot be decoded into a valid UAV command produces a safe no-op and receives the penalty in Table 6.

B.2 Failure Distribution

The recorded failures span several navigation difficulties, with collisions accounting for the largest share. Specifically, Fig. 4 summarizes 1488 events from fresh and memory-selected task executions. Collisions account for 1073 events, or 72.1%, followed by continuous-away and timeout events with 183 and 138 cases, or 12.3% and 9.3%. Early-stop and stuck events contribute 48 and 46 cases, together accounting for 6.3%. These events provide feedback on obstacle avoidance, directional progress, and task termination beyond the terminal success signal. To provide unresolved tasks from the start of post-training, we initialize the memory using failures collected with the initial policy and subsequently update its records using navigation outcomes.

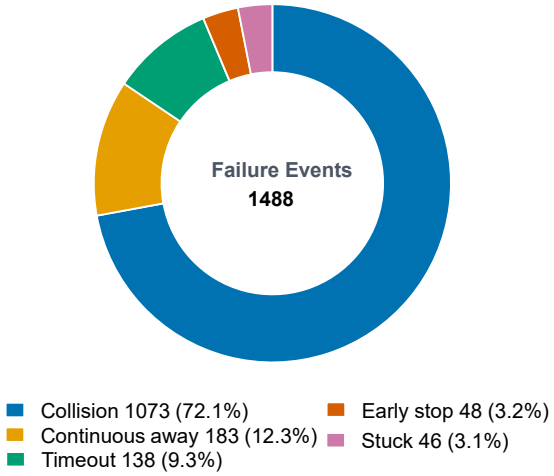


Figure 4: Distribution of 1488 closed-loop navigation failure events recorded during RefineFly training. Collisions account for 1073 events, with 415 events across the other four failure types. Invalid action outputs are excluded.

Table 7: Paired task-level outcomes for token-level PPO (ID 1) and PPO with dynamic failure memory (ID 2) on identical evaluation tasks. S and F denote success and failure. The $F \to S$ rate is $n_{F\to S}/(n_{F\to S} + n_{F\to F})$, and the $S \to F$ rate is $n_{S\to F}/(n_{S\to S} + n_{S\to F})$. The net change is $\Delta\text{SR} = 100(n_{F\to S} - n_{S\to F})/N$. All Splits pools the counts over $N = 3005$ tasks and recomputes each rate using its corresponding denominator.

	Paired Outcome Transitions				Outcome Changes			
Evaluation Split	$S\to S$	$S\to F$	$F\to S$	$F\to F$	Net Successes	$F\to S$ Rate (%)	$S\to F$ Rate (%)	ΔSR (pp)
Seen	493	190	304	431	+114	41.36	27.82	+8.04
Unseen Map	185	226	114	433	−112	20.84	54.99	−11.69
Unseen Object	268	87	108	166	+21	39.42	24.51	+3.34
All Splits	946	503	526	1,030	+23	33.80	34.71	+0.77

B.3 Observed Memory Outcomes

Fig. 5 summarizes the final states of 1488 failure entries. The observed memory states summarize task outcomes from both sampling sources over the recorded interval. Specifically, Fig. 5 reports 720 resolved, 724 active, and 44 dropped records out of 1488. The resolved count includes active tasks completed successfully through fresh sampling as well as memory-selected executions. Active records remain eligible for further training, while dropped records have exhausted the retry limit. Together, these states characterize the memory lifecycle under the available interaction budget.

Dynamic failure memory yields more favorable paired outcomes on Seen and Unseen Object, supporting the value of sustained training on unresolved tasks. Specifically, Table 7 compares the two policies on identical evaluation tasks. On Seen, 304 tasks change from failure under ID 1 to success under ID 2, while 190 change from success to failure, yielding 114 net successes and an 8.04-point SR increase. Similarly, Unseen Object has 108 failure-to-success transitions and 87 success-to-failure transitions, yielding 21 net successes and a 3.34-point SR increase. These gains are consistent with repeated current-policy interaction providing further learning opportunities under previously unsuccessful navigation conditions.

The paired outcomes also reveal a trade-off in transfer to unseen maps under the empirical scene distribution. Specifically, Unseen Map has 114 failure-to-success transitions and 226 success-to-failure transitions, yielding 112 net lost successes and an 11.69-point SR decrease. Across all 3005 tasks, the net gain is 23 successes, or 0.77 percentage points. This pooled change weights splits by task count, whereas the main-paper ablation reports an unweighted mean across splits, which

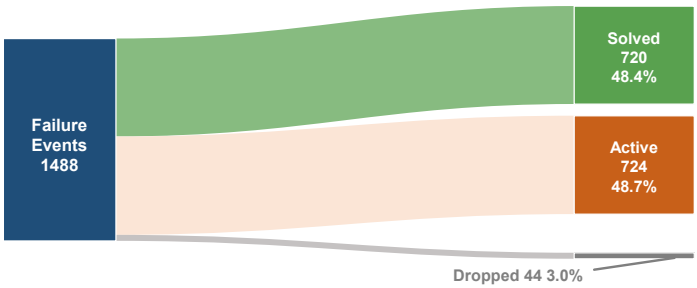


Figure 5: Final states of 1488 failure-memory records over the recorded training interval. The 720 resolved records include successful executions from both memory selection and fresh sampling of active tasks. Another 724 records remain active, while 44 are dropped after two unsuccessful retries.



Figure 6: Average return of token-level PPO and the sequence-level baseline over 100 training steps.

decreases by 0.10 points from ID 1 to ID 2. Together, these results indicate that sustained failure learning can improve some navigation outcomes while weakening transfer to new maps. Since memory sampling preserves the empirical scene allocation in this comparison, rare scenes retain a limited share of training opportunities, motivating reference constraints and the scene rebalancing evaluated in the main text.

C Comparison of Token-Level and Sequence-Level PPO

This section provides empirical support for token-level PPO as the optimization backbone of RefineFly. Specifically, the sequence-level alternative computes one importance ratio from the joint likelihood of the complete action-token sequence. Fig. 6 compares training returns, while Table 8 reports Full-split success rate (SR) for both alternatives. Token-level PPO corresponds to ID 1 in the main-paper component ablation.

Token-level PPO achieves higher training returns than the sequence-level alternative in the recorded run. As shown in Fig. 6, its average return rises rapidly during the first 15 training steps and subsequently remains mostly between 2.0 and 3.0, with a peak near 3.5. In comparison, the sequence-level return mostly stays below 1.0 without comparable growth, and the gap persists through the later training steps.

The higher training returns are accompanied by better navigation performance across all three evaluation splits. Specifically, token-level PPO exceeds the sequence-level alternative by 5.11 to 7.63 percentage points in SR (Table 8). Relative to AerialVLA in Table 4, sequence-level PPO reduces mean SR by 4.53 points, while token-level PPO increases it by 1.79 points. These results support the token-level backbone for the subsequent study of failure memory and scene sampling.

The different treatment of action-token probability ratios offers a possible explanation for the observed performance gap. Specifically, each action contains three control tokens and an optional `LAND` token. Sequence-level PPO couples these probabilities into a joint importance ratio, allow-

Table 8: Full-split SR (%) for sequence-level and token-level PPO on the three TravelUAV evaluation splits. Token-level PPO corresponds to ID 1 in the main-paper component ablation. Bold indicates the higher result in each row.

Evaluation Split	Sequence-Level PPO	Token-Level PPO
Seen	41.96	**48.17**
Unseen Map	37.79	**42.90**
Unseen Object	48.81	**56.44**

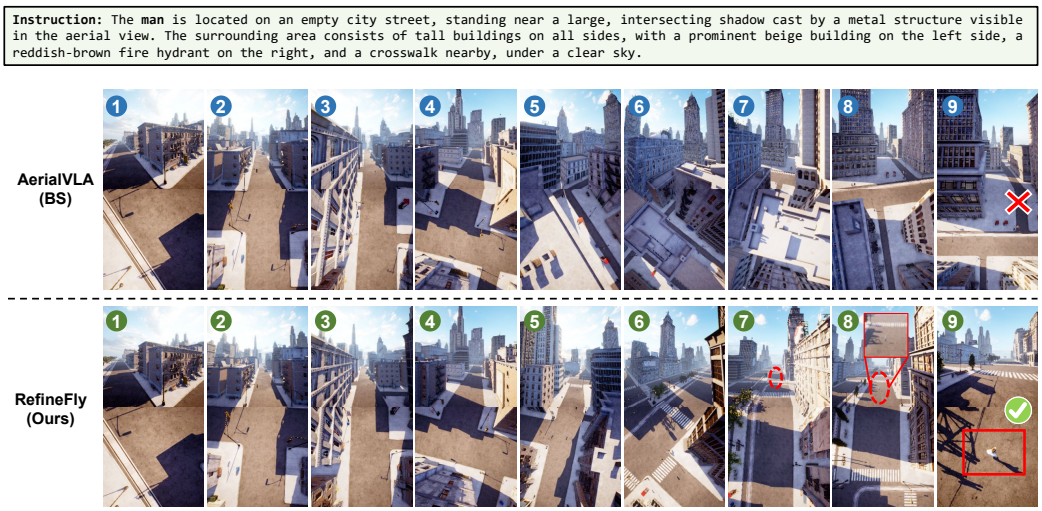


Figure 7: Qualitative navigation comparison under the same instruction and initial observation. The top and bottom rows show key observations from AerialVLA and RefineFly, respectively. Numbers indicate their order within each trajectory.

ing a change at one position to activate clipping of the shared surrogate objective. In comparison, token-level PPO applies clipping separately at each valid position while sharing the action-level advantage across positions. This formulation reduces coupling in the clipping decisions and may retain learning signals at positions whose individual ratios remain within the clipping range.

D Qualitative Analysis

This section illustrates differences in closed-loop navigation between AerialVLA and RefineFly under the same instruction and initial observation. Specifically, Fig. 7 presents nine key observations from each trajectory. Both policies initially traverse similar city blocks, while their behavior diverges later in the route.

As shown in Fig. 7, AerialVLA initially follows a similar route and subsequently moves away from the target area. In comparison, RefineFly proceeds toward the street marked by the crosswalk and intersecting shadow. The target becomes clearly visible in the eighth observation, after which RefineFly continues its descent and reaches the target location in the final observation. This example illustrates improved target approach and task completion following post-training, consistent with the aggregate navigation gains reported in the main text.