SA-GCS: Semantic-Aware Gaussian Curriculum Scheduling for UAV Vision-Language Navigation

Anonymous ACL submission

Abstract

Unmanned Aerial Vehicle (UAV) Vision-Language Navigation (VLN) requires an agent to reach a target in complex environments by interpreting natural language instructions and visual observations. While Reinforcement Learning (RL) post-training of Vision-Language Models (VLMs) has proven effective for improving navigation performance and generalization, the training process still suffers from instability and slow convergence, largely due to the significant variance in sample difficulty. To address this, we propose Semantic-Aware Gaussian Curriculum Scheduling (SA-GCS), a framework that systematically integrates Curriculum Learning (CL) into RL training for UAV VLN. The Semantic-Aware Difficulty Estimator (SADE) leverages cross-modal attention from a VLM to quantify the alignment between the model's semantic focus and the ground-truth target region, yielding a difficulty score that reflects navigation executability. The Gaussian Curriculum Scheduler (GCS) then constructs a Gaussian sampling distribution over these scores and dynamically shifts its mean during training, enabling a smooth easy-to-hard curriculum with continuous exploration. Extensive experiments on the CityNav dataset show that SA-GCS consistently improves success rate and path efficiency across different model scales, while accelerating convergence and producing smoother training curves. The source code of our implementation is publicly available[1].

1 Introduction

Vision-Language Navigation (VLN) for Unmanned Aerial Vehicles (UAVs) aims to guide a UAV to a target location by interpreting natural language instructions and visual observations. It has broad applications in intelligent inspection, aerial search and rescue, and urban monitoring. Early research explored Sequence-to-Sequence (Seq2Seq) (Anderson et al., 2018), Cross-Modal Attention (CMA) (Liu et al., 2023), and Map-based Goal Prediction (MGP) (Lee et al., 2024), which rely on task-specific architectures trained from scratch and often struggle to generalize to diverse instructions and unseen environments.

Recently, Large Language Models (LLMs) and particularly Vision-Language Models (VLMs) have substantially advanced instruction comprehension and visual reasoning (Li et al., 2025; Zhang et al., 2025; Ping et al., 2025), but their generalization to unseen environments remains limited. Since Supervised Fine-Tuning (SFT) alone struggles to cope with dynamic real-world environments, Reinforcement Learning (RL) has become an essential post-training strategy, demonstrating significant improvements in VLM generalization (Saxena et al., 2025; Chu et al., 2025; Tan et al., 2025).

Despite these advances, RL training for UAV VLN still faces a fundamental challenge: the significant variance in sample difficulty. Some samples involve clear instructions and obvious targets, while others combine ambiguous semantics, complex spatial relationships, or densely distracting visual environments. Training with such mixed-difficulty samples without structured guidance causes the model to overfit on easy samples while failing to learn effectively from hard ones, ultimately leading to unstable training, slow convergence, and limited overall performance.

A natural approach to handling such difficulty variance is Curriculum Learning (CL), which organizes training samples from easy to hard and has proven effective in improving convergence and generalization across various domains (Bengio et al., 2009; Li et al., 2023a; Wang et al., 2023). The effectiveness of CL, however, fundamentally depends on how sample difficulty is defined. Unlike tasks with clear difficulty signals, UAV VLN difficulty is jointly determined by spatial distance, en-

[1] https://anonymous.4open.science/r/SA-GCS-4E86

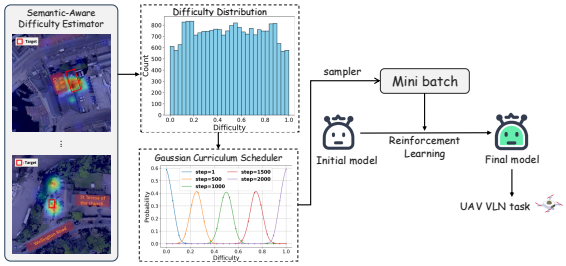

Figure 1: The SA-GCS framework. SADE assigns a difficulty score to each training sample from cross-modal attention, and GCS converts the scores into a time-varying Gaussian sampling distribution that drives RL training. The trained policy is then deployed for UAV VLN.

vironmental complexity, and cross-modal semantic alignment quality, necessitating a difficulty measure that captures the inherent executability of each navigation task—i.e., how reliably the model can ground the target semantics and execute a successful trajectory.

To this end, we propose SA-GCS, a CL framework designed for RL training in UAV VLN. As shown in Figure 1, SA-GCS consists of two core components. The Semantic-Aware Difficulty Estimator (SADE) leverages cross-modal attention from a VLM to measure the alignment between the model's semantic focus and the ground-truth target region, yielding a difficulty score that reflects navigation executability. The Gaussian Curriculum Scheduler (GCS) then constructs a Gaussian sampling distribution over these scores and dynamically shifts its mean during training, enabling a smooth easy-to-hard curriculum with continuous exploration.

The main contributions are as follows: first, we propose SADE, a difficulty estimator that leverages cross-modal attention to quantify sample difficulty in a task-aligned manner for UAV VLN. Second, building upon SADE, we further propose GCS, which together with SADE forms the SA-GCS framework, systematically integrating CL into RL training for UAV VLN. Third, extensive experiments on the CityNav dataset demonstrate that SA-GCS achieves substantial performance gains across different model scales, while accelerating training convergence and producing smoother training curves.

2 Related Work

UAV VLN requires agents to locate targets by interpreting natural language instructions and visual inputs (Wang et al., 2024). Early supervised methods include Seq2Seq (Anderson et al., 2018), CMA (Liu et al., 2023), and MGP (Lee et al., 2024). While recent VLMs demonstrate strong cross-modal reasoning (Saxena et al., 2025; Liu et al., 2024), their generalization to unseen environments remains limited. RL-based post-training has emerged to address this gap (Cai et al., 2025; Chu et al., 2025), but faces challenges including sparse rewards and low sample efficiency.

CL for RL improves training stability by organizing samples from easy to hard (Bengio et al., 2009; Li et al., 2023a). Existing approaches differ in *difficulty modeling*: explicit methods control environment parameters (Tidd et al., 2020), while implicit methods estimate reachability (Florensa et al., 2017). For *curriculum scheduling*, strategies range from adaptive sampling based on success rates (Nesterova et al., 2022) to continuous distribution shifts (Parashar et al., 2025). However, these methods rely on heuristic difficulty measures rather than semantic alignment quality, limiting their effectiveness for multimodal tasks like UAV VLN. See Appendix A for a detailed discussion.

3 Method

3.1 Task Formulation

Each UAV VLN task can be formalized as a triplet (I, D, E), where I denotes the UAV's initial state, including its position and heading; D is a natural language instruction describing the target's characteristics and surrounding landmarks; and E represents the environment. The UAV acquires first-person RGB observations from E and can access a 2D map representation that contains geographic information.

The UAV completes the task by executing a sequence of discrete actions, including **move forward**, **turn left**, **turn right**, and **stop**. When the UAV determines that it is sufficiently close to the target, it may choose the **stop** action. An episode is considered successful if the final stopping position is within a predefined threshold distance (e.g., 20 meters) of the ground-truth target.

3.2 System Framework and Reinforcement Learning

3.2.1 System Pipeline

To address the task in Section 3.1, we follow the standard UAV VLN pipeline adopted in prior studies (Anderson et al., 2018; Liu et al., 2023; Lee et al., 2024; Cai et al., 2025), which consists of four modules: **Perception Acquisition**, **Goal Inference**, **Action Planning**, and **Environment Interaction**. As shown in Figure 2, these modules form a closed-loop system.

1. Perception Acquisition The system collects two types of real-time input: (1) a Semantic Map, a structured representation integrating geographical semantic elements such as roads, buildings, and landmarks, which also encodes the UAV's current position and heading; and (2) a Natural Language Instruction, a textual description outlining the characteristics of the target and its surrounding reference objects.

2. Goal Inference Based on the input, the model generates a reasoning chain that progressively extracts clues related to the target. It then predicts the final coordinates of the target on the map.

3. Action Planning After obtaining the estimated target location, the system adopts a look-ahead mechanism (Liu et al., 2023) to determine the action sequence to be executed at the current time step.

4. Environment Interaction The UAV executes the action sequence, resulting in updates to both the environment and its own state.

This loop is repeated continuously until a **stop** action is triggered or the maximum number of iterations is reached.

3.2.2 Reinforcement Learning Mechanism

To enhance the performance and generalization of the **Goal Inference** module in dynamic environments, we adopt Group Relative Policy Optimization (GRPO) for policy optimization. Following the reward design in FlightGPT (Cai et al., 2025), we employ a composite reward function $R_{\text{total}} = R_{\text{goal}} + R_{\text{IoU}} + R_{\text{format}}$ consisting of three components: (1) a **Goal Accuracy Reward** R_{goal} that measures the Euclidean distance between the predicted and ground-truth target locations; (2) an **Intermediate Reasoning Reward** R_{IoU} based on the IoU between the predicted and actual landmark bounding boxes during the reasoning phase; and (3) a **Format Reward** R_{format} that encourages well-structured output with properly formatted reasoning and answer segments. The complete mathematical formulation is provided in Appendix B.

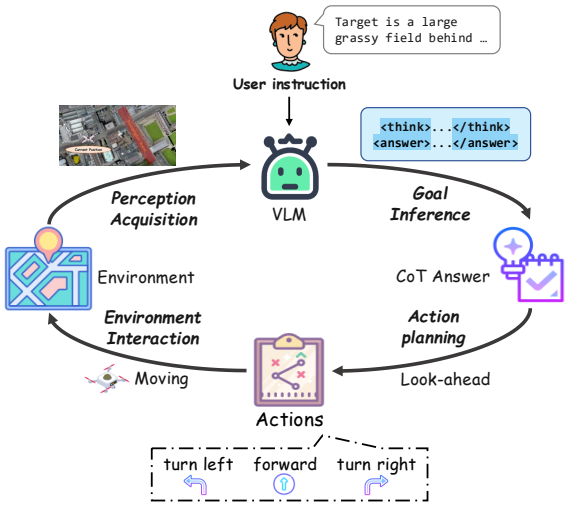

Figure 2: The system pipeline for UAV VLN.

3.3 Difficulty Modeling

When incorporating CL into RL, accurate modeling of sample difficulty is a fundamental prerequisite for curriculum design. Without a reliable difficulty estimation mechanism, CL may fail to align training samples with different learning stages and can even introduce bias, thereby undermining its overall effectiveness.

3.3.1 Understanding Sample Difficulty in UAV VLN

The difficulty of UAV VLN samples is not primarily determined by superficial attributes such as environment scale or instruction length. Instead, it largely depends on the reliability of cross-modal semantic alignment, i.e., whether the model can accurately ground the target semantics described in the language instruction to the correct visual region. When the sample semantics are ambiguous, cross-modal alignment becomes less reliable, making the task harder to execute. Therefore, sample difficulty in UAV VLN should be modeled from the perspective of cross-modal semantic alignment quality, rather than relying solely on heuristic attributes of the instruction or the environment.

3.3.2 From Principle to Metric: Semantic-Aware Difficulty Estimator

Motivated by this insight, we propose SADE as a concrete instantiation for modeling sample difficulty from the perspective of cross-modal semantic

alignment. The core idea of SADE is to leverage the cross-modal attention within VLMs as a signal of semantic alignment. Intuitively, when the model processes instructions and its attention is concentrated on the correct target region, the semantic grounding is clear and the corresponding sample is easier to learn. In contrast, when the attention is dispersed or biased toward incorrect regions, it indicates uncertainty in semantic alignment, rendering the sample more difficult. SADE consists of two main steps: (1) constructing cross-modal attention heatmaps to characterize the model's semantic focus on the target region; and (2) computing a continuous difficulty score based on the alignment between the attention heatmap and the ground-truth target region.

Cross-Modal Attention Heatmap Construction We construct heatmaps for semantic-aware difficulty estimation based on the cross-modal attention mechanisms in VLMs. Prior studies, including BLIP-2 (Li et al., 2023b), GIT (Wang et al., 2022), and Q-GroundCAM (Rajabi and Kosecka, 2024), have demonstrated that cross-modal attention in Transformers provides effective signals for semantic alignment. As illustrated in Figure 3, heatmap construction involves four steps:

1. **Cross-Modal Attention Weight Extraction.** The instruction and image are fed into the VLM, and cross-attention weight matrices are extracted from the decoder to characterize the attention distribution.

2. **Target-Aware Token-to-Patch Attention Retrieval.** The target description provided in the instruction (i.e., the `{target description}` field in the prompt template; see Appendix C) is used to locate the corresponding tokens, and their attention responses to visual patches are extracted.

3. **Head-wise and Layer-wise Aggregation.** Multi-head attention maps are averaged, followed by aggregation across layers, resulting in a final one-dimensional attention vector.

4. **Heatmap Generation.** The attention vector is reshaped into a 2D grid and upsampled to the original image resolution via interpolation. The resulting cross-modal attention heatmap is overlaid on the input image to highlight the regions attended by the model.

The attention heatmap can be viewed as the model's semantic judgment of the instruction-specified target location, which serves as the foundation for subsequent difficulty computation.

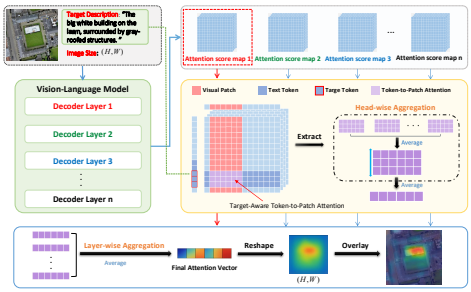

Figure 3: Workflow for constructing the Cross-Modal Attention Heatmap from a Vision-Language Model.

Difficulty Score Computation To transform semantic alignment into a scalar difficulty signal suitable for CL, we use the Soft-IoU metric (Jiang et al., 2018), which has been widely adopted in prior research to measure alignment. Let $M \in \{0,1\}^{H\times W}$ be the binary mask where the target region is 1 and all other regions are 0. Let $A \in [0,1]^{H\times W}$ denote the attention heatmap generated by the model, where higher values indicate that the model pays more attention to those regions. The Soft-IoU is defined as:

$$\text{IoU}_{\text{soft}} = \frac{\sum(A \cdot M)}{\sum A + \sum M - \sum(A \cdot M)}$$

Based on this, we define the Semantic-Aware Difficulty score as:

$$\text{Difficulty} = 1 - \text{IoU}_{\text{soft}}, \quad \text{Difficulty} \in [0,1].$$

This metric quantitatively reflects both the model's localization confidence and its alignment accuracy with the ground-truth target region. A score closer to 1 indicates low confidence and misaligned attention, suggesting the sample is semantically harder for the model. Conversely, a score near 0 implies accurate focus and high confidence, indicating an easier sample. Note that the ground-truth target masks required by SADE are only used during training to compute difficulty scores; they are not needed at inference time.

Figure 4 illustrates two representative cases. The **easy sample** (left) corresponds to the instruction: *"the brown and gray building adjacent to the S & G Cycles building"*. In this case, the target building is located immediately next to a clearly labeled landmark ("S & G Cycles"), offering strong visual and spatial cues. The model's attention is highly concentrated on the correct region, resulting in a heatmap that overlaps significantly with

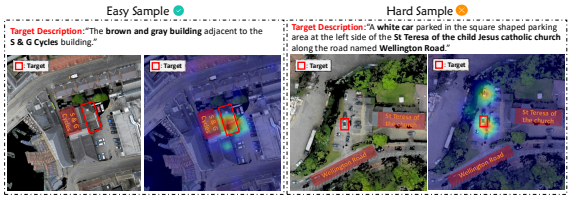

Figure 4: Representative examples of easy and hard samples in difficulty estimation.

the ground-truth mask. Consequently, the Soft-IoU score is high and the computed difficulty is low, indicating that the model successfully focuses on and identifies the target. In contrast, the **hard sample** (right) is associated with a more complex instruction: *"a white car parked in the square-shaped parking area at the left side of the St Teresa of the Child Jesus catholic church along the road named Wellington Road"*. This description chains multiple spatial prepositions ("at the left side of", "along"), requiring the model to resolve nested geometric relationships. Furthermore, the image contains several white vehicles that could serve as potential candidates. As a result, the attention heatmap is dispersed and overlaps poorly with the ground-truth region, leading to a lower Soft-IoU score and a higher difficulty value.

3.4 Curriculum Scheduling Strategies

3.4.1 Linear Curriculum Scheduler

As the most straightforward way to leverage the SADE-based difficulty signal, we first adopt a linear CL strategy that organizes training samples in ascending order of difficulty. Owing to its simplicity and its intuitive embodiment of the "easy-to-hard" CL paradigm, this strategy is used as the baseline curriculum scheduling method.

3.4.2 Gaussian Curriculum Scheduler

Although the linear curriculum scheduler serves as an effective baseline, its sample selection mechanism is relatively rigid, making it difficult to achieve continuous and smooth transitions across samples of varying difficulty levels. To address this limitation, we propose the GCS, which models the sampling probability of training samples with different difficulty levels using a Gaussian distribution. The mean of this distribution is smoothly shifted along the training process, enabling a gradual progression from easy to hard samples while maintaining continuous exposure to samples of other difficulty levels.

Specifically, based on SADE, we assign each training sample x_i (where i indexes the samples in the training set) a difficulty score $d_i \in [0, 1]$. At training step t, the scheduler constructs a Gaussian sampling distribution according to the current training progress and computes the sampling probability $P_i(t)$ for each sample x_i as:

$$P_i(t) = \frac{1}{Z} \exp\left(-\frac{(d_i - \mu(t))^2}{2\sigma^2}\right). \quad (1)$$

Here, $\mu(t) \in [0, 1]$ is the mean of the Gaussian distribution, which gradually shifts from lower to higher difficulty levels as training progresses. The parameter σ controls the spread (standard deviation) of the distribution, and Z is a normalization constant ensuring that all sampling probabilities sum to 1. $\mu(t)$ is updated dynamically via a linear scheduling strategy:

$$\mu(t) = \mu_0 + \frac{t}{T}(1 - \mu_0), \quad (2)$$

where μ_0 denotes the initial mean value, and T is the total number of training steps. We set $\mu_0 = 0$ to start sampling from the easiest samples and progressively cover the full difficulty range, which is a natural default that does not require tuning.

3.4.3 The Complete SA-GCS Framework

Building upon the SADE difficulty scores and the GCS sampling mechanism described above, we integrate both components into a unified training framework. The overall SA-GCS process is summarized in Algorithm 1.

Algorithm 1 Semantic-Aware Gaussian Curriculum Scheduler

Input: Difficulty scores $\{d_i\}_{i=1}^{N}$ for training samples x_i;
Total training steps T;
Initial mean μ_0 and standard deviation σ.
Output: Final trained model.

```
Initialize the RL model
for t = 1 to T do
    Update the current mean μ(t)
    foreach sample x_i do
        Compute sampling probability P_i(t)
    Sample a mini-batch from the dataset based on P_i(t)
    Perform policy optimization using the mini-batch
    Update model parameters
return final trained model
```

4 Experiments

4.1 Datasets and Metrics

We conduct our experiments on the CityNav dataset (Lee et al., 2024), a large-scale simulated benchmark designed for UAV VLN. CityNav reconstructs urban layouts, architectural structures, and semantic landmarks within a simulated environment, providing a realistic testbed for navigation systems. The dataset contains a large number of human-written natural language instructions, covering a diverse range of linguistic styles and navigation objectives. Each instruction is paired with a corresponding ground-truth navigation trajectory, making it well-suited for both training and evaluation in UAV VLN. The dataset also provides ground-truth target region annotations, which can be used by SADE to compute difficulty scores.

For performance evaluation, we follow the official metrics defined in the benchmark: Navigation Error (NE), Success Rate (SR), Oracle Success Rate (OSR), and Success weighted by Path Length (SPL). See Appendix E for definitions.

4.2 Baselines

We select a set of representative baseline methods, covering rule-based strategies (Random), supervised learning approaches (Seq2Seq, CMA, and MGP), general-purpose VLMs (Qwen, LLaMA, and GPT-4o), as well as the latest VLM-based state-of-the-art method (FlightGPT), forming a systematic comparison group. To ensure a fair comparison, each baseline is provided with the same task inputs whenever required by the method, and the VLM-based baselines adopt the same prompt template as in our main setting. More details are provided in Appendix F.

4.3 Implementation Details

The experiment uses Qwen2.5-VL-7B as the base model for both navigation policy training and SADE difficulty estimation. The overall training process follows a two-stage paradigm of "SFT + RL": instruction fine-tuning is first applied to obtain an initialization policy with basic task capabilities, followed by RL to further improve the model's generalization in unknown environments. This training paradigm has been demonstrated to be effective in related studies (Cai et al., 2025). A customized prompt template is designed to guide the model in producing structured reasoning and accurate target predictions, with its complete format detailed in Appendix C. The computational cost of SADE is analyzed in Appendix D. The hyperparameters for SFT, RL, and CL are detailed in Appendix G.

5 Results and Analysis

5.1 Overall Performance

Table 1 presents a comparison of our method against various baseline models on the CityNav dataset. All results are expressed as the mean value ± standard deviation, based on 5 independent runs. Across three data splits and four evaluation metrics, SA-GCS consistently achieves the best or highly competitive results. In particular, on the more challenging Test-Unseen split, SA-GCS significantly outperforms all baseline methods in terms of SR and SPL, demonstrating strong generalization capability in unseen environments. Furthermore, we conduct the same experiments on Qwen2.5-VL-3B (results in Appendix H), where a consistent performance improvement trend is observed, further validating the effectiveness of SA-GCS across different model scales.

5.2 Ablation Study

5.2.1 Difficulty Estimation Strategy

To validate the effectiveness of different difficulty modeling strategies in the UAV VLN task, we conduct an ablation study on the difficulty estimation module. We adopt **Random sampling** as the no-curriculum baseline, where training samples are uniformly sampled from the dataset at each step. On this basis, we introduce CL by ranking samples with a difficulty score and gradually shifting training from easy to hard. Specifically, we compare three difficulty signals: (1) **Distance-based Difficulty**, which uses the Euclidean distance between the starting position and the target location, where greater distance indicates higher difficulty; (2) **CLIP-based Difficulty** (Radford et al., 2021), which computes cross-modal similarity between the instruction and visual observation using a CLIP model, where lower similarity indicates higher difficulty; and (3) **SADE**, the proposed difficulty estimation method in this work.

Table 1 presents the experimental results under identical scheduling strategies (Linear and Gaussian) with different difficulty signals. The distance-based difficulty signal yields only limited improvements over the Random baseline, suggesting that spatial distance alone is insufficient to character-

Table 1: Comparison of Model Performance Across Different Data Splits.

Method	Validation Seen				Validation Unseen				Test Unseen			
	NE↓	SR↑	OSR↑	SPL↑	NE↓	SR↑	OSR↑	SPL↑	NE↓	SR↑	OSR↑	SPL↑
Random	$221.8_{\pm 1.6}$	$0.0_{\pm 0.0}$	$1.2_{\pm 0.1}$	$0.0_{\pm 0.0}$	$224.3_{\pm 0.9}$	$0.0_{\pm 0.0}$	$0.9_{\pm 0.1}$	$0.0_{\pm 0.0}$	$209.5_{\pm 1.2}$	$0.0_{\pm 0.0}$	$1.4_{\pm 0.1}$	$0.0_{\pm 0.0}$
Seq2Seq	$149.1_{\pm 0.6}$	$4.5_{\pm 0.2}$	$10.7_{\pm 0.5}$	$4.4_{\pm 0.3}$	$200.8_{\pm 1.2}$	$1.0_{\pm 0.1}$	$7.8_{\pm 0.2}$	$1.0_{\pm 0.1}$	$174.8_{\pm 0.9}$	$1.7_{\pm 0.1}$	$8.5_{\pm 0.4}$	$1.6_{\pm 0.1}$
CMA	$151.2_{\pm 0.9}$	$3.8_{\pm 0.1}$	$10.7_{\pm 0.4}$	$3.7_{\pm 0.1}$	$206.1_{\pm 0.8}$	$1.1_{\pm 0.1}$	$8.0_{\pm 0.3}$	$1.1_{\pm 0.1}$	$178.4_{\pm 1.1}$	$1.6_{\pm 0.1}$	$10.2_{\pm 0.3}$	$1.6_{\pm 0.1}$
MGP	$59.3_{\pm 0.4}$	$8.8_{\pm 0.4}$	$\mathbf{35.3_{\pm 1.1}}$	$8.2_{\pm 0.2}$	$75.6_{\pm 0.3}$	$5.7_{\pm 0.3}$	$22.4_{\pm 0.7}$	$5.5_{\pm 0.2}$	$93.2_{\pm 0.6}$	$6.5_{\pm 0.3}$	$25.8_{\pm 0.5}$	$6.0_{\pm 0.3}$
Qwen2.5-VL-32B	$84.2_{\pm 0.5}$	$12.8_{\pm 0.3}$	$24.4_{\pm 0.7}$	$11.2_{\pm 0.4}$	$92.4_{\pm 0.4}$	$10.2_{\pm 0.4}$	$20.6_{\pm 0.4}$	$8.9_{\pm 0.3}$	$83.7_{\pm 0.3}$	$11.9_{\pm 0.2}$	$23.7_{\pm 0.6}$	$10.7_{\pm 0.2}$
LLaMA-3.2-11B-Vision	$199.6_{\pm 1.1}$	$1.2_{\pm 0.1}$	$5.1_{\pm 0.2}$	$1.1_{\pm 0.1}$	$214.4_{\pm 0.7}$	$0.5_{\pm 0.1}$	$4.4_{\pm 0.2}$	$0.5_{\pm 0.1}$	$190.3_{\pm 0.8}$	$1.3_{\pm 0.1}$	$4.5_{\pm 0.1}$	$1.2_{\pm 0.1}$
GPT-4o	$149.0_{\pm 1.0}$	$2.3_{\pm 0.1}$	$7.1_{\pm 0.4}$	$2.1_{\pm 0.1}$	$160.8_{\pm 0.6}$	$2.5_{\pm 0.2}$	$6.7_{\pm 0.4}$	$2.3_{\pm 0.2}$	$145.2_{\pm 0.5}$	$3.7_{\pm 0.2}$	$8.7_{\pm 0.3}$	$3.3_{\pm 0.2}$
FlightGPT	$65.8_{\pm 0.3}$	$17.8_{\pm 0.6}$	$30.1_{\pm 0.8}$	$15.7_{\pm 0.4}$	$68.5_{\pm 0.4}$	$14.6_{\pm 0.5}$	$29.5_{\pm 0.6}$	$13.3_{\pm 0.5}$	$76.8_{\pm 0.5}$	$21.4_{\pm 0.7}$	$35.2_{\pm 1.0}$	$19.1_{\pm 0.6}$
Qwen2.5-VL-7B	$115.5_{\pm 0.4}$	$4.9_{\pm 0.3}$	$12.9_{\pm 0.2}$	$4.3_{\pm 0.3}$	$123.7_{\pm 0.5}$	$5.7_{\pm 0.1}$	$14.3_{\pm 0.2}$	$5.0_{\pm 0.1}$	$119.0_{\pm 0.2}$	$6.7_{\pm 0.1}$	$15.2_{\pm 0.2}$	$5.9_{\pm 0.1}$
Qwen2.5-VL-7B-SFT	$97.2_{\pm 0.4}$	$10.4_{\pm 0.4}$	$18.3_{\pm 0.4}$	$9.6_{\pm 0.3}$	$102.3_{\pm 0.6}$	$10.4_{\pm 0.2}$	$18.6_{\pm 0.5}$	$9.6_{\pm 0.2}$	$117.9_{\pm 0.5}$	$11.1_{\pm 0.3}$	$21.4_{\pm 0.6}$	$10.7_{\pm 0.4}$
Qwen2.5-VL-7B-SFT + RL (Random)	$68.3_{\pm 0.3}$	$15.4_{\pm 0.6}$	$27.5_{\pm 0.6}$	$13.9_{\pm 0.5}$	$67.2_{\pm 0.4}$	$15.6_{\pm 0.4}$	$28.8_{\pm 0.8}$	$13.8_{\pm 0.3}$	$75.8_{\pm 0.5}$	$21.2_{\pm 0.7}$	$34.9_{\pm 0.8}$	$19.1_{\pm 0.6}$
Qwen2.5-VL-7B-SFT + RL (Dist. + Linear)	$67.9_{\pm 0.4}$	$15.5_{\pm 0.4}$	$27.6_{\pm 0.8}$	$14.0_{\pm 0.3}$	$66.8_{\pm 0.3}$	$15.7_{\pm 0.5}$	$29.2_{\pm 0.6}$	$13.8_{\pm 0.5}$	$74.7_{\pm 0.3}$	$21.3_{\pm 0.5}$	$35.4_{\pm 0.9}$	$19.2_{\pm 0.4}$
Qwen2.5-VL-7B-SFT + RL (Dist. + GCS)	$66.5_{\pm 0.2}$	$15.7_{\pm 0.5}$	$27.8_{\pm 0.7}$	$14.4_{\pm 0.4}$	$66.4_{\pm 0.4}$	$15.8_{\pm 0.3}$	$29.4_{\pm 0.7}$	$14.2_{\pm 0.3}$	$74.2_{\pm 0.3}$	$21.5_{\pm 0.6}$	$35.5_{\pm 0.7}$	$19.4_{\pm 0.5}$
Qwen2.5-VL-7B-SFT + RL (CLIP + Linear)	$65.8_{\pm 0.4}$	$16.1_{\pm 0.3}$	$28.3_{\pm 0.7}$	$14.5_{\pm 0.3}$	$65.7_{\pm 0.3}$	$16.0_{\pm 0.5}$	$29.6_{\pm 0.5}$	$14.2_{\pm 0.5}$	$73.5_{\pm 0.4}$	$21.9_{\pm 0.5}$	$35.7_{\pm 0.9}$	$19.9_{\pm 0.4}$
Qwen2.5-VL-7B-SFT + RL (CLIP + GCS)	$64.7_{\pm 0.3}$	$16.6_{\pm 0.5}$	$28.7_{\pm 0.5}$	$15.1_{\pm 0.5}$	$64.9_{\pm 0.2}$	$16.2_{\pm 0.4}$	$30.1_{\pm 0.7}$	$14.6_{\pm 0.3}$	$72.4_{\pm 0.4}$	$22.4_{\pm 0.6}$	$36.1_{\pm 0.7}$	$20.5_{\pm 0.6}$
Qwen2.5-VL-7B-SFT + RL (SADE + Linear)	$63.1_{\pm 0.2}$	$17.3_{\pm 0.4}$	$29.6_{\pm 0.7}$	$15.8_{\pm 0.5}$	$64.2_{\pm 0.4}$	$16.5_{\pm 0.5}$	$30.7_{\pm 0.6}$	$15.1_{\pm 0.3}$	$71.0_{\pm 0.3}$	$23.2_{\pm 0.5}$	$36.4_{\pm 0.9}$	$21.4_{\pm 0.6}$
Qwen2.5-VL-7B-SFT + RL (SADE + GCS)	$\mathbf{59.2_{\pm 0.2}}$	$\mathbf{18.8_{\pm 0.4}}$	$31.8_{\pm 0.6}$	$\mathbf{17.4_{\pm 0.5}}$	$\mathbf{63.4_{\pm 0.2}}$	$\mathbf{17.6_{\pm 0.4}}$	$\mathbf{31.8_{\pm 0.5}}$	$\mathbf{15.8_{\pm 0.3}}$	$\mathbf{68.1_{\pm 0.4}}$	$\mathbf{24.7_{\pm 0.6}}$	$\mathbf{37.5_{\pm 0.9}}$	$\mathbf{22.8_{\pm 0.6}}$

ize the true difficulty of UAV VLN samples. The CLIP-based difficulty signal also shows limited effectiveness, as CLIP primarily captures the global semantic relevance between language instructions and visual observations. However, in UAV VLN, even when the visual input contains semantic elements highly correlated with the instruction, navigation may still fail due to ambiguous target localization, dispersed attention, or semantic interference from visually similar regions. In contrast, when SADE is employed as the difficulty signal, the model achieves consistent and substantial improvements across all evaluation metrics. SADE estimates difficulty from the model's internal perspective, aligning the difficulty definition closely with the executability of navigation decisions, thereby providing a more reliable and task-aligned sample ranking for CL.

To further validate the effectiveness of SADE as a difficulty measure, we conduct a quantitative analysis of the discriminative ability of the three difficulty signals. Specifically, we use FlightGPT, the strongest baseline, to perform full inference on the validation set, recording for each sample whether the task succeeds and the corresponding three difficulty scores. We then sort the samples from easy to hard by each difficulty signal and equally divide them into 10 bins, compute the average SR within each bin, and calculate the Pearson correlation coefficient r based on the average difficulty score and average success rate of the 10 bins.

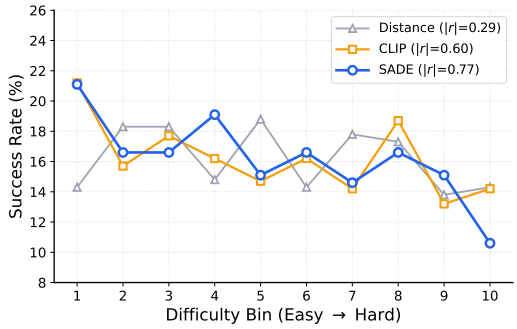

Figure 5: Correlation analysis between difficulty scores and task SR. Samples are divided into 10 bins by each difficulty signal, with average SR per bin shown. Pearson r is reported for each signal.

As shown in Figure 5, SADE exhibits the most pronounced decreasing trend in success rate as difficulty increases ($|r|$ is the largest), indicating that SADE scores are highly correlated with the actual task difficulty. The CLIP curve shows a certain downward trend but with substantial fluctuations, as it fails to capture local alignment failures. Path distance demonstrates the weakest predictive ability, further confirming that physical distance is not the determining factor of navigation difficulty in UAV VLN.

Notably, SADE + Linear outperforms CLIP + GCS across all three splits (e.g., Test-Unseen SR: 23.2 vs 22.4), suggesting that the quality of the difficulty signal has a greater impact on CL effectiveness than the choice of scheduling strategy.

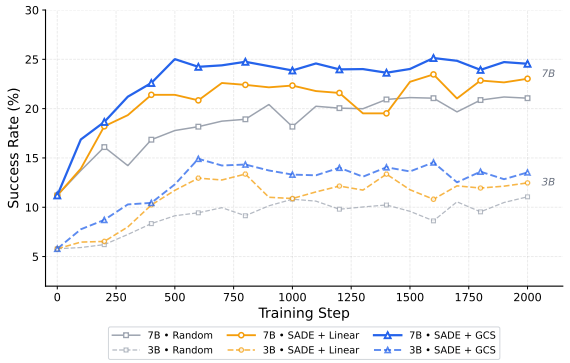

Figure 6: SR progression on the Test-Unseen split using different sampling strategies (3B and 7B).

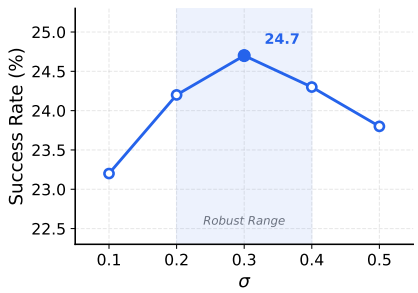

Figure 7: Sensitivity of GCS to σ on Test-Unseen.

5.2.2 Curriculum Scheduling Strategy

With SADE fixed as the difficulty signal, we further compare three sampling scheduling strategies: Random sampling, Linear scheduling, and GCS. As shown in Table 1, SADE + Linear outperforms the Random baseline across all splits, indicating that CL facilitates the model in learning higher-quality navigation strategies. Building upon this, SADE + GCS further surpasses Linear scheduling in both SR and SPL, suggesting that the GCS, by maintaining continuous coverage over samples of varying difficulty, can more effectively balance curriculum progression and sample diversity.

From the training curves (Figure 6), GCS achieves both faster convergence and smoother training curves. For the 7B model, SADE + GCS approaches convergence after approximately 500 steps, whereas SADE + Linear requires roughly 700 steps. Moreover, in the later training phase (700–2000 steps), the Linear scheduler exhibits noticeable SR fluctuations, with the trough at step 1300 (19.52%) and the peak at step 1600 (23.47%) differing by 3.95%, while the GCS training curve remains relatively smooth. Figure 6 also shows the SR progression of the 3B model on the Test-Unseen split. It can be observed that regardless of the model scale, SA-GCS consistently accelerates performance improvement in the early training stages, enabling the model to reach effective navigation strategies more quickly.

5.2.3 Sensitivity to Hyperparameters

In SA-GCS, the core hyperparameter of GCS is the standard deviation σ of the Gaussian distribution, which controls the concentration of sampling at each training stage. To evaluate the sensitivity of the framework to this hyperparameter, we train models with $\sigma \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$ and report SR on the Test-Unseen split, as shown in Figure 7. SA-GCS maintains stable performance across a wide range of $\sigma \in [0.2, 0.4]$, with SR varying within 0.5 percentage point. The best performance is achieved at $\sigma = 0.3$, which is the default setting used throughout all experiments in this paper.

6 Conclusion

This paper proposes SA-GCS, a framework that integrates CL into RL training for UAV VLN. The method leverages SADE to extract difficulty signals aligned with navigation executability from the model's cross-modal attention, and combines it with GCS to achieve a smooth easy-to-hard training process. Experiments on the CityNav dataset demonstrate that SA-GCS significantly improves navigation performance while accelerating convergence, with consistent effectiveness across both 3B and 7B models. Ablation studies further verify that SADE outperforms alternative difficulty signals such as distance and CLIP, and that GCS surpasses linear scheduling in both convergence speed and training-curve smoothness.

7 Limitations

7.1 Gap Between Simulation and Real-World Deployment

All experiments in this work are conducted in the simulation environment. However, real-world UAV deployment involves additional challenges such as perception noise, lighting variations, and GPS positioning errors, meaning that performance improvements observed in simulation may not directly transfer to real environments. Bridging the sim-to-real gap remains an important direction for future work.

7.2 Scope of Curriculum Learning Applicability

SA-GCS is currently designed and validated for city-scale target localization tasks in UAV VLN. Different types of VLN tasks (e.g., indoor navigation or multi-turn continuous instruction following) involve different difficulty compositions and task characteristics, and the curriculum scheduling strategy may require corresponding adjustments. Extending SA-GCS to a broader range of navigation scenarios and verifying its generalizability is a direction worth further exploration.

7.3 Dependence on Ground-Truth Target Annotations

SADE relies on ground-truth target masks to compute difficulty scores, which may not be available in all VLN datasets or real-world scenarios. Future work could explore using pseudo-labels generated by object detectors or segmentation models to approximate SADE in weakly supervised settings.

8 Ethical Considerations

8.1 Safety in Deployment Scenarios

UAV navigation is inherently safety-critical. Despite strong benchmark performance, deployed systems may encounter perception failures or misinterpret ambiguous instructions, which could result in unsafe flight behavior. We therefore stress that the proposed method serves as a research contribution rather than a deployment-ready solution, and should be integrated with appropriate safety mechanisms such as geofencing and human override before any real-world application.

8.2 Potential for Dual Use

VLN capabilities for UAVs can be applied to beneficial tasks such as disaster response and infrastructure monitoring, but may also be repurposed for unauthorized tracking or surveillance. We intend this work solely for academic research and encourage future users to establish clear usage guidelines and access restrictions when releasing associated code and models.

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, 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, and 8 others. 2025. Qwen2.5-vl technical report. *Preprint*, arXiv:2502.13923.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Hengxing Cai, Jinhan Dong, Jingjun Tan, Jingcheng Deng, Sihang Li, Zhifeng Gao, Haidong Wang, Zicheng Su, Agachai Sumalee, and Renxin Zhong. 2025. Flightgpt: Towards generalizable and interpretable uav vision-and-language navigation with vision-language models. *arXiv preprint arXiv:2505.12835*.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. 2025. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*.

Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. 2017. Reverse curriculum generation for reinforcement learning. In *Conference on robot learning*, pages 482–495. PMLR.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Borui Jiang, Ruixuan Luo, Jiayuan Mao, Tete Xiao, and Yuning Jiang. 2018. Acquisition of localization confidence for accurate object detection. In *Proceedings of the European conference on computer vision (ECCV)*, pages 784–799.

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. *arXiv preprint arXiv:2406.14240*.

Chengxi Li, Pai Zheng, Yue Yin, Baicun Wang, and Lihui Wang. 2023a. Deep reinforcement learning in smart manufacturing: A review and prospects. *CIRP Journal of Manufacturing Science and Technology*, 40:75–101.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023b. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR.

Tianshun Li, Tianyi Huai, Zhen Li, Yichun Gao, Haoang Li, and Xinhu Zheng. 2025. Skyvln: Vision-and-language navigation and nmpc control for uavs in urban environments. *arXiv preprint arXiv:2507.06564*.

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

Youzhi Liu, Fanglong Yao, Yuanchang Yue, Guangluan Xu, Xian Sun, and Kun Fu. 2024. Navagent: Multi-scale urban street view fusion for uav embodied vision-and-language navigation. *arXiv preprint arXiv:2411.08579*.

Maria Nesterova, Alexey Skrynnik, and Aleksandr Panov. 2022. Reinforcement learning with success induced task prioritization. In *Mexican International Conference on Artificial Intelligence*, pages 97–107. Springer.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Shubham Parashar, Shurui Gui, Xiner Li, Hongyi Ling, Sushil Vemuri, Blake Olson, Eric Li, Yu Zhang, James Caverlee, Dileep Kalathil, and 1 others. 2025. Curriculum reinforcement learning from easy to hard tasks improves llm reasoning. *arXiv preprint arXiv:2506.06632*.

Yuqi Ping, Tianhao Liang, Huahao Ding, Guangyu Lei, Junwei Wu, Xuan Zou, Kuan Shi, Rui Shao, Chiya Zhang, Weizheng Zhang, and 1 others. 2025. Multimodal large language models-enabled uav swarm: Towards efficient and intelligent autonomous aerial systems. *arXiv preprint arXiv:2506.12710*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and 1 others. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR.

Navid Rajabi and Jana Kosecka. 2024. Q-groundcam: Quantifying grounding in vision language models via gradcam. *arXiv preprint arXiv:2404.19128*.

Pranav Saxena, Nishant Raghuvanshi, and Neena Goveas. 2025. Uav-vln: End-to-end vision language guided navigation for uavs. *arXiv preprint arXiv:2504.21432*.

Huajie Tan, Yuheng Ji, Xiaoshuai Hao, Minglan Lin, Pengwei Wang, Zhongyuan Wang, and Shanghang Zhang. 2025. Reason-rft: Reinforcement fine-tuning for visual reasoning. *arXiv preprint arXiv:2503.20752*.

Brendan Tidd, Nicolas Hudson, and Akansel Cosgun. 2020. Guided curriculum learning for walking over complex terrain. *arXiv preprint arXiv:2010.03848*.

Hsueh-Cheng Wang, Siao-Cing Huang, Po-Jui Huang, Kuo-Lun Wang, Yi-Chen Teng, Yu-Ting Ko, Dongsuk Jeon, and I-Chen Wu. 2023. Curriculum reinforcement learning from avoiding collisions to navigating among movable obstacles in diverse environments. *IEEE Robotics and Automation Letters*, 8(5):2740–2747.

Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. 2022. Git: A generative image-to-text transformer for vision and language. *arXiv preprint arXiv:2205.14100*.

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*.

Xinyuan Zhang, Yonglin Tian, Fei Lin, Yue Liu, Jing Ma, Kornélia Sára Szatmáry, and Fei-Yue Wang. 2025. Logisticsvln: Vision-language navigation for low-altitude terminal delivery based on agentic uavs. *arXiv preprint arXiv:2505.03460*.

A Detailed Related Work

A.1 Vision-Language Navigation: Supervised and Reinforcement Learning Approaches

UAV VLN tasks require an agent to navigate in a complex environment by interpreting natural language instructions and visual inputs. The objective is to perform target localization and flight path planning, making it a representative task that combines cross-modal understanding with autonomous decision-making (Wang et al., 2024).

Early works in UAV VLN predominantly adopted supervised learning frameworks to optimize performance. For instance, Seq2Seq models encode image and instruction sequences to produce outputs, marking an initial exploration of UAV VLN (Anderson et al., 2018). Building upon this, CMA mechanisms were introduced to enhance perception–language alignment(Liu et al., 2023). Further advancements such as the MGP integrated language understanding and visual perception to construct navigation maps and perform goal prediction, demonstrating improved structural reasoning and planning abilities (Lee et al., 2024).

With the recent advances in LLMs and VLMs, researchers have started leveraging general-purpose VLMs for UAV VLN. Benefiting from their strong cross-modal perception capabilities, these models can achieve performance comparable to task-specific systems even with limited training data (Saxena et al., 2025; Liu et al., 2024). However, their generalization capability in complex and unseen environments remains limited, hindering practical deployment.

To address this, RL has been explored as a post-training strategy to improve generalization. For example, FlightGPT proposed a two-stage training pipeline consisting of SFT followed by RL, and demonstrated its effectiveness and generalization on UAV VLN (Cai et al., 2025). Recent studies, including those from the Google team, further validated that RL-based post-training outperforms pure SFT in enhancing model generalizability (Chu et al., 2025; Tan et al., 2025).

Nevertheless, RL approaches still face challenges such as unstable reward signals, slow convergence, and low sample efficiency. Designing robust and efficient training mechanisms remains an open and urgent problem in this area.

A.2 Curriculum Learning in Reinforcement Learning

CL aims to organize training samples in an "easy-to-hard" manner, enabling models to receive more stable and dense learning signals at early stages and gradually acquire more complex capabilities and policies (Bengio et al., 2009). In RL, the training process often suffers from sparse rewards and low sample efficiency. Introducing CL into RL is commonly regarded as an effective approach to improving training efficiency and stability (Li et al., 2023a; Wang et al., 2023). The key challenges lie in how to characterize task difficulty and how to schedule the training distribution.

Existing CL methods for RL can be broadly categorized into two types in terms of **task difficulty modeling**: explicit and implicit. Explicit difficulty modeling defines difficulty levels through controllable environment parameters or task configurations. For instance, Guided Curriculum Learning (GCL) designs a three-stage curriculum with increasing terrain complexity and external perturbations, guiding robots to learn progressively from flat ground to complex obstacles, which significantly shortens convergence time and improves traversal success rates in locomotion tasks (Tidd et al., 2020). In contrast, implicit difficulty modeling emphasizes task difficulty from the perspective of *task reachability* or *success probability*. For example, Reverse Curriculum Generation starts from goal states and gradually expands to more distant initial states, guiding agents to learn to approach the goal from easy to hard, thereby accelerating the learning process for tasks such as grasping and navigation (Florensa et al., 2017).

Regarding curriculum scheduling strategies, some studies adopt adaptive sampling mechanisms based on performance feedback. For example, Success-Induced Task Prioritization (SITP) dynamically adjusts the sampling distribution according to the agent's success rates across tasks, prioritizing tasks with intermediate success rates to balance exploration and exploitation, and achieves higher sample efficiency on multi-task control benchmarks (Nesterova et al., 2022). Another line of work emphasizes stable curriculum progression through continuous and smooth distribution evolution. For instance, the Easy-to-Hard (E2H) Scheduler models the sampling probability of tasks with different difficulty levels as a continuous distribution that shifts along training steps during RL fine-

tuning, significantly improving the accuracy and sample efficiency of small-scale models (Parashar et al., 2025).

Although curriculum-based RL has achieved notable progress across various domains, its application in UAV VLN remains relatively limited. Unlike other tasks, the difficulty of UAV VLN is jointly influenced by multiple factors, including spatial distance, environmental complexity, and the quality of cross-modal semantic alignment, while its RL training process often relies more on stable and progressive task distribution scheduling. Therefore, how to construct a reasonable difficulty measure and further design stable and effective curriculum scheduling strategies remains a key open challenge in this direction.

B Reward Function Details

Following FlightGPT (Cai et al., 2025), we employ a composite reward function consisting of three components to guide policy optimization from multiple perspectives.

Goal Accuracy Reward. The accuracy of the predicted target location is a key indicator of the system's effectiveness. Let the UAV's predicted position be $\hat{\mathbf{p}} = (\hat{p}_x, \hat{p}_y)$ and the ground truth be $\mathbf{p}^* = (p_x^*, p_y^*)$. We define the reward based on their Euclidean distance $d(\hat{\mathbf{p}}, \mathbf{p}^*) = \sqrt{(\hat{p}_x - p_x^*)^2 + (\hat{p}_y - p_y^*)^2}$:

$$R_{\text{goal}} = \begin{cases} 1, & \text{if } d(\hat{\mathbf{p}}, \mathbf{p}^*) \leq d_{\text{success}} \\ \exp\left(-\dfrac{d(\hat{\mathbf{p}}, \mathbf{p}^*) - d_{\text{success}}}{\tau}\right), & \text{if } d_{\text{success}} < d(\hat{\mathbf{p}}, \mathbf{p}^*) \leq d_{\text{cutoff}} \\ 0, & \text{otherwise} \end{cases}$$

where $d_{\text{success}} = 20$ meters is the threshold for task success, $d_{\text{cutoff}} = 80$ meters is the upper limit beyond which no reward is given, and $\tau = 100$ is the decay temperature controlling the sharpness of the exponential drop-off.

Intermediate Reasoning Reward. To provide guidance for intermediate reasoning steps, we introduce a reward based on the Intersection over Union (IoU) between the predicted landmark bounding box $\hat{B}$ and the ground-truth bounding box B:

$$R_{\text{IoU}} = \frac{\text{Area}(B \cap \hat{B})}{\text{Area}(B \cup \hat{B})}$$

If the model fails to output a valid bounding box, we set $R_{\text{IoU}} = 0$. This mechanism incentivizes spatial reasoning before location prediction.

Format Reward. To ensure the model generates structured outputs, we introduce a format compliance reward. If the output includes both `<think>` and `<answer>` tags properly enclosing the reasoning and answer segments, a reward of $+0.5$ is given. If a `"landmark_bbox"` field in the format `[x1, y1, x2, y2]` is successfully extracted within the `<think>` tag, an additional $+0.25$ is granted. If a `"target_location"` field in the format `[x, y]` is successfully extracted within the `<answer>` tag, another $+0.25$ is granted.

Overall Reward. The total reward used for policy optimization is the sum of the three components:

$$R_{\text{total}} = R_{\text{goal}} + R_{\text{IoU}} + R_{\text{format}}$$

C Prompts Template

Prompt 1: UAV VLN task

System Message:
You are an intelligent autonomous aerial vehicle (UAV) capable of real-world navigation and visual target localization.

Mission Objective:
Your mission is to locate a specific target described in natural language instructions.

Details of the Target:
`{target description}`

Environmental Perception:
- The UAV's current position is indicated by the starting point of an arrow in the image, with its heading angle represented by the arrow's direction.
- The yellow box outlines the UAV's current camera field of view on the map, centered at pixel coordinates: `cur_pose = {UAV current position}`.
- Landmark regions are highlighted with red masks.

Operational Guidance:
- The target is usually located near a red-masked landmark.
- Use both the target description and the visual input to identify the most relevant red-masked landmark region.
- Infer the relative position of the target with respect to that landmark.

Output Format Specification:
- Present your reasoning process within <think> and </think> tags.
Your reasoning may include:
- A semantic interpretation of the target description.
- Identification of the correct landmark region.
- The bounding box of that region in the following format:
{"landmark_bbox": [x1, y1, x2, y2]}
- Provide your final answer within <answer> and </answer> tags in the following format:
{"target_location": [x, y]}

D Computational Overhead Analysis

The SADE difficulty estimation is a one-time preprocessing step performed before RL training begins. Using 4 × A100 GPUs, computing SADE scores for all 22k training samples takes approximately 6 hours, which accounts for 10% of the total RL training time (approximately 60 hours). Since SADE scores are precomputed once and reused throughout training, no additional per-step overhead is introduced.

E Evaluation Metrics

- **Navigation Err (NE):** The Euclidean distance (in meters) between the UAV's final position and the target location. A lower NE indicates higher accuracy.
- **Success Rate (SR):** The percentage of episodes where the UAV ends within 20 meters of the target location. This reflects the overall task completion rate.
- **Oracle Success Rate (OSR):** The proportion of episodes where any point along the UAV's trajectory comes within 20 meters of the target. This captures whether the target was ever approached during navigation.
- **Success weighted by Path Length (SPL):** Among successful episodes, SPL evaluates the efficiency of the navigation path. It penalizes unnecessary detours and rewards trajectories close to the shortest path.

F Baseline Method Details

- **Random**
This baseline randomly selects an action at each time step, without relying on any perceptual input or language instruction. It serves as a reference for strategy-free control performance.
- **Seq2Seq** (Anderson et al., 2018)
This method encodes perceptual inputs and language instructions using a recurrent sequence model to produce navigation outputs. It represents an early end-to-end supervised learning approach for navigation tasks.
- **CMA** (Liu et al., 2023)
Built upon the Seq2Seq framework, CMA introduces a cross-modal attention mechanism to enhance the alignment and fusion of visual and linguistic features, improving the model's ability to handle multimodal information.
- **MGP** (Lee et al., 2024)
MGP combines a language parser and an object detection module to extract semantic information about goals and landmarks. It then constructs a semantic navigation map to predict the target location.
- **Qwen2.5-VL-7B** (Bai et al., 2025)
A multimodal large language model based on the Qwen architecture with 7 billion parameters. It supports joint understanding and reasoning over visual and linguistic inputs.
- **Qwen2.5-VL-32B** (Bai et al., 2025)
A larger-scale variant in the Qwen2.5 series, equipped with 32 billion parameters and enhanced capability for complex multimodal reasoning.
- **LLaMA-3.2-11B-Vision** (Grattafiori et al., 2024)
A vision-language model based on the LLaMA architecture with 11 billion parameters. It effectively handles visual and textual inputs across diverse multimodal tasks.
- **GPT-4o** (OpenAI et al., 2024)
A state-of-the-art multimodal large language model that accepts image and text inputs, and demonstrates strong performance in cross-modal understanding and instruction following.
- **FlightGPT** (Cai et al., 2025)
A state-of-the-art method based on VLMs,

which enables interpretable navigation decisions via structured reasoning and policy optimization, and substantially improves generalization to unseen environments.

G Training Hyperparameters

Table 2 lists the hyperparameters for SFT and RL training, and Table 3 summarizes the hyperparameters for CL.

Table 2: Hyperparameters for SFT and RL

Stage	Batch Size	Learning Rate	#Epochs	Rollout
SFT	16	2e-5	2	–
RL	1	1e-5	1	4

Table 3: Hyperparameters for CL

μ_0	σ	#Steps	Samples/Step
0	0.3	2000	2

H Results on Qwen2.5-VL-3B

To evaluate the scalability of SA-GCS across different model sizes, we conduct the same set of experiments using Qwen2.5-VL-3B as the base model. As shown in Table 4, SA-GCS consistently improves performance over all baselines across all splits, with the SADE + GCS configuration achieving the best results, demonstrating that the proposed framework generalizes effectively to smaller model scales.

Table 4: Comparison of Model Performance Across Different Data Splits (3B).

Method	**Validation Seen**				**Validation Unseen**				**Test Unseen**			
	NE↓	SR↑	OSR↑	SPL↑	NE↓	SR↑	OSR↑	SPL↑	NE↓	SR↑	OSR↑	SPL↑
Qwen2.5-VL-3B	$170.5_{\pm 0.8}$	$1.5_{\pm 0.1}$	$3.5_{\pm 0.1}$	$1.4_{\pm 0.1}$	$182.4_{\pm 1.0}$	$1.2_{\pm 0.1}$	$3.3_{\pm 0.2}$	$1.1_{\pm 0.1}$	$166.3_{\pm 0.7}$	$1.4_{\pm 0.1}$	$2.8_{\pm 0.1}$	$1.3_{\pm 0.1}$
Qwen2.5-VL-3B-SFT	$121.4_{\pm 0.5}$	$4.7_{\pm 0.2}$	$13.9_{\pm 0.4}$	$4.3_{\pm 0.2}$	$121.5_{\pm 0.7}$	$5.5_{\pm 0.2}$	$14.4_{\pm 0.3}$	$4.9_{\pm 0.3}$	$124.2_{\pm 0.6}$	$5.7_{\pm 0.3}$	$16.3_{\pm 0.5}$	$5.2_{\pm 0.2}$
Qwen2.5-VL-3B-SFT + RL (Random)	$88.8_{\pm 0.5}$	$10.3_{\pm 0.4}$	$22.4_{\pm 0.7}$	$9.1_{\pm 0.3}$	$90.7_{\pm 0.4}$	$8.5_{\pm 0.3}$	$21.7_{\pm 0.5}$	$7.4_{\pm 0.3}$	$93.4_{\pm 0.6}$	$11.0_{\pm 0.5}$	$24.3_{\pm 0.6}$	$10.2_{\pm 0.4}$
Qwen2.5-VL-3B-SFT + RL (SADE + Linear)	$85.2_{\pm 0.4}$	$11.4_{\pm 0.3}$	$24.5_{\pm 0.6}$	$10.3_{\pm 0.4}$	$89.2_{\pm 0.3}$	$9.4_{\pm 0.4}$	$23.3_{\pm 0.7}$	$8.5_{\pm 0.2}$	$89.8_{\pm 0.5}$	$12.4_{\pm 0.4}$	$26.3_{\pm 0.5}$	$11.5_{\pm 0.3}$
Qwen2.5-VL-3B-SFT + RL (SADE + GCS)	$\mathbf{82.7}_{\pm 0.3}$	$\mathbf{12.8}_{\pm 0.4}$	$\mathbf{26.3}_{\pm 0.5}$	$\mathbf{11.6}_{\pm 0.3}$	$\mathbf{85.2}_{\pm 0.5}$	$\mathbf{10.8}_{\pm 0.3}$	$\mathbf{24.6}_{\pm 0.6}$	$\mathbf{9.8}_{\pm 0.4}$	$\mathbf{86.1}_{\pm 0.4}$	$\mathbf{13.6}_{\pm 0.5}$	$\mathbf{27.6}_{\pm 0.7}$	$\mathbf{12.5}_{\pm 0.4}$