Ro-SLM: Onboard Small Language Models for Robot Task Planning and Operation Code Generation

Wenhao Wang[1], Yanyan Li[2], Long Jiao[1], Jiawei Yuan[1]

[1]Department of Computer & Information Science, University of Massachusetts Dartmouth
[2]Department of Computer Science & Engineering, California State University San Marcos
{wwang5, ljiao, jyuan}@umassd.edu, yali@csusm.edu

Abstract

Recent advances in large language models (LLMs) provide robots with contextual reasoning abilities to comprehend human instructions. Yet, current LLM-enabled robots typically depend on cloud-based models or high-performance computing infrastructure, which limit their deployment on robots under unreliable internet environments or with constrained computational resources, such as UAVs and small ground vehicles. Thus, deploying fine-tuned small language models (SLMs) that support onboard deployment offers a promising alternative. This paper introduces Ro-SLM, a framework that enables reliable SLM-driven robot operation by distilling LLMs' knowledge and reasoning. Ro-SLM starts from dataset synthesis by leveraging LLMs to generate diverse task instructions, produce corresponding ground truth code with minimal human assistance, and augment instructions into real-world application scenarios. Ro-SLM is then fine-tuned with the dataset, in which LLM serves as a reward function to guide the training. Extensive experiments on UAV operation tasks demonstrate that Ro-SLM improves the performance of SLM from being incapable of supporting robotic task planning and code generation to achieving performance that approaches LLM[1].

1 Introduction

Recent advances in LLMs (Achiam et al., 2023; Team et al., 2023) have demonstrated remarkable proficiency in robotic areas such as control (Tagliabue et al., 2024), planning (Mohanan and Salgoankar, 2018), and navigation (Yu et al., 2023). Trained on internet-scale data and comprising billions of parameters, LLMs encode rich knowledge and exhibit strong semantic understanding and context-generation capabilities, enabling robots to reliably comprehend human instructions for planning and generate corresponding robot operation code (Vemprala et al., 2024; Liang et al., 2023; Team et al., 2025). As a result, increasing efforts have focused on designing structured prompting strategies (Marvin et al., 2023) to enhance foundational LLMs (e.g., OpenAI GPT and Gemini) in supporting robotic tasks reliably (Arenas et al., 2024; Liang et al., 2023; Wang et al., 2025a).

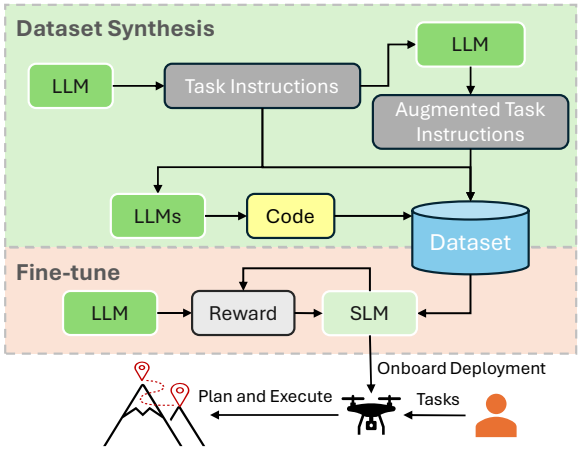

Figure 1: Ro-SLM framework for enabling SLM-driven robot operations.

Despite LLMs' strong reasoning capabilities, the scale of LLMs brings significant overhead and limits their applications on resource-constrained robotic platforms. On the one hand, cloud-based LLMs such as OpenAI GPT4 (Achiam et al., 2023) require a stable internet connection to access remote inference services, which is infeasible for robots operating in environments with unreliable connectivity, such as mountainous regions or disaster response. On the other hand, locally deploying LLMs (e.g., Llama-3.1-405B (Grattafiori et al., 2024)) demands significant computational and energy resources, making them impractical for robots with limited payload and battery capacity, such as small ground vehicles or UAVs. Leveraging

[1]Project is available at https://github.com/ai-uavsec/Ro-SLM

Findings of the Association for Computational Linguistics: ACL 2026, pages 34436–34460

SLMs for resource-constrained robotic platforms provides another promising alternative (Lu et al., 2024). While the deployment of SLMs has been practically demonstrated, they exhibit weaker reasoning capabilities due to their reduced parameter scale (Chen et al., 2025c), which hence limits their ability to extract task-relevant information from prompts and generalize to novel robotic tasks like LLMs.

To overcome these limitations, we propose Ro-SLM (Robot operation with Small Language Model), a framework that enables reliable SLM-driven robot operations by distilling the knowledge and reasoning from LLMs. As illustrated in Figure 1, Ro-SLM consists of two stages:

Dataset Synthesis A critical component of effective fine-tuning of Ro-SLM is constructing a high-quality dataset that consists of varying task complexity scenarios and corresponding execution plans and operation code. Given the challenge that the generation of such robot task and execution data is time-consuming and requires domain experts, this paper configured LLMs with expert knowledge to facilitate the construction and validation of a high-quality synthetic fine-tune dataset. Specifically, an instruction set is first generated by configuring an LLM agent to automatically generate task instructions with varying complexities and clearly specified task objectives, thereby reducing the risk of misinterpretations during code generation. Then, a corrective code generation approach (Wang et al., 2025b) accurately generates corresponding ground truth robot operation code with human assistance introduced only for high-complexity tasks. To align SLM with real human-like task descriptions, we further leverage the world knowledge of LLM to augment the instruction set with real-world robotic application scenarios. In addition, to prevent the SLM from overfitting to syntactic patterns in code comments, we remove all comments from the ground truth code set.

Fine-tune After constructing the dataset, the SLM is first trained using supervised fine-tuning (SFT). To ensure the model is getting the best result, the LLM then serves as a reward function to guide the model optimization process.

We conducted extensive experiments on UAV operation tasks with varying levels of complexity to evaluate the effectiveness of Ro-SLM. Our evaluation shows that the Ro-SLM with Llama-3.1-8B achieves 97.7% Success Rate and 98.9% Completeness on simple tasks, i.e., 98.9% of required actions in all evaluated tasks are completed correctly and 97.7% of tasks are entirely completed without any error. For complex tasks, Ro-SLM achieves a 70% Success Rate and 83.7% Completeness. Compared with the performance of OpenAI o4-mini LLM, Ro-SLM achieves a comparable performance. Thus, Ro-SLM is promising to act as an intelligent agent for resource-constrained robot platforms for task planning and operation code generation.

2 Related Work

2.1 LLM-driven Robot Code Generation

Compared with traditional pipelines in which human experts manually craft robot operation code from task instructions, recent research has increasingly leveraged LLMs' natural language understanding and reasoning capabilities to automatically infer tasks and generate corresponding robot-operation code or plans (Wang et al., 2025a; Vemprala et al., 2024; Chen et al., 2025a; Liang et al., 2023). To enhance reasoning performance and ensure the correctness of generated code for robotic tasks that require complex reasoning, prior works have employed prompt engineering (Marvin et al., 2023) to shape LLM behavior through structured system prompts, such as constraints and examples (Arenas et al., 2024; Wang et al., 2025a). More recent approaches further improve reliability by adopting corrective code-generation frameworks, in which errors and mismatches in generated code are identified and iteratively refined (Chen et al., 2024; Joublin et al., 2024; Wang et al., 2026). Notably, most existing methods rely on LLMs (e.g., OpenAI GPT) due to their remarkable reasoning capabilities; however, these methods depend on cloud-based computation and internet connectivity and therefore cannot be deployed onboard robots. In contrast, SLMs have significantly fewer parameters and are more suitable for onboard deployment, but they struggle to achieve comparable performance as the LLMs (Hu et al., 2025).

2.2 Dataset Generation

Fine-tuned SLMs can achieve performance comparable to that of LLMs on domain-specific tasks (Zhan et al., 2025). However, unlike domains such as computer vision, robot tasks are customized and robot capabilities-dependent, resulting in fine-tuning dataset construction being labor-intensive and difficult to scale. Recent approaches have leveraged LLMs' world knowledge

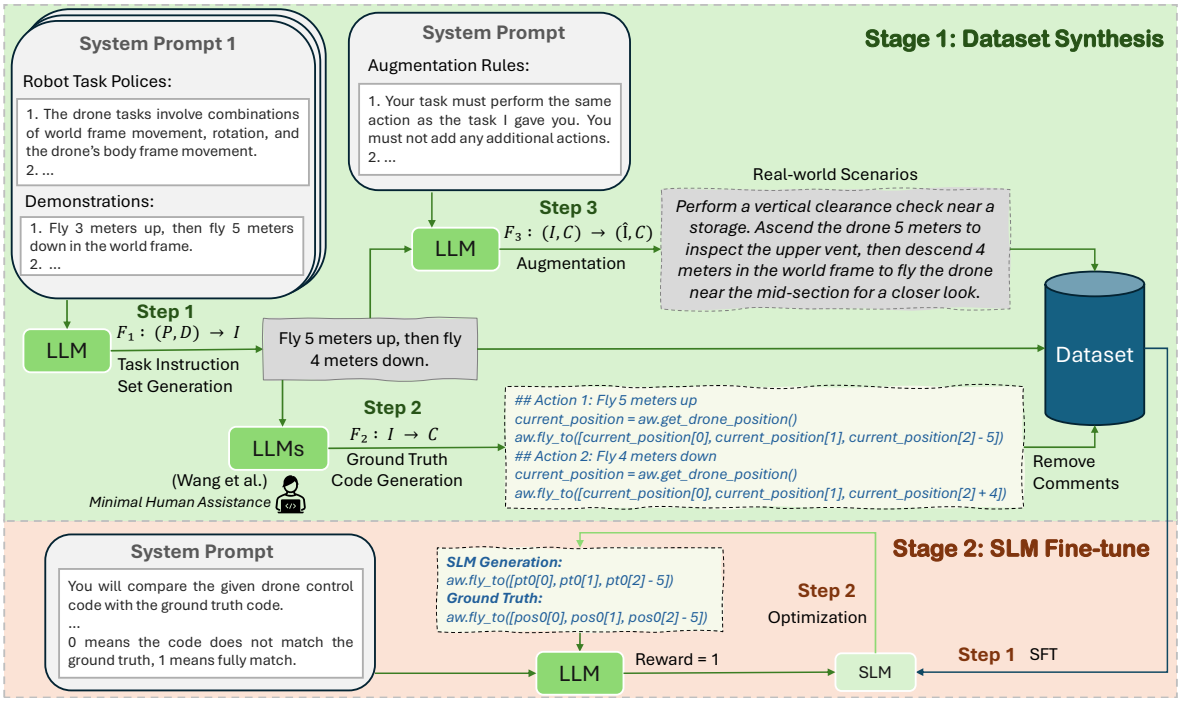

Figure 2: Ro-SLM overview: dataset synthesis and SLM fine-tuning. The LLMs are configured with different system prompts to support instruction generation, augmentation, ground truth code mapping, and the reward function. The detailed system prompts are provided in Appendix A, B, and C.

acquired from the large-scale pretraining to synthesize a dataset for fine-tuning. For example, LLM-Trainer uses LLMs to create and annotate robot trajectories based on the demonstrations (George and Farimani, 2025). PRISM distills an LLM-enabled planner to train an SLM that rivals the planning performance of LLMs (Ravichandran et al., 2025). Other work (Wen et al., 2025; Chen et al., 2025b; Zhu et al., 2024) incorporates human-in-the-loop strategies for synthesizing task instructions and robot operation plans. Moreover, LLMs are prone to hallucinations and SLMs are sensitive to data contamination (Scaria et al., 2025). Therefore, ensuring the quality of dataset generation is crucial for the reliability of fine-tuned SLMs.

2.3 SLM Fine-tune

Fine-tuning adapts pretrained language models to specific tasks by training on labeled input-output pairs so that the model learns to follow desired behaviors or instructions (Minaee et al., 2024). Several techniques have been proposed to improve the efficiency of fine-tuning. To reduce the GPU memory usage, LoRA introduces low-rank trainable updates to pretrained model weights, which significantly reduces the number of trainable parameters and the size of optimizer states (Hu et al., 2022). Moreover, to align model outputs with human preferences, reinforcement learning from human feedback (RLHF) optimizes the model using rewards derived from human feedback (Ouyang et al., 2022). To further improve the efficiency of RLHF, group relative policy optimization (GRPO) (Shao et al., 2024) removes the need for training a reward model as used in PPO-based (Schulman et al., 2017) approaches, which also reduces memory usage and accelerates training.

3 Method

3.1 Overview

Figure 2 presents the workflow of our Ro-SLM framework. In the *Stage 1*, we synthesize the fine-tune dataset, which consists of three key steps: (1) **Task Instruction Set Generation**: an LLM-based agent generates a set of concise and executable task instructions for robotic operations of varying complexity; (2) **Ground Truth Code Generation**: LLM agents with enhanced reasoning produce corresponding ground truth code with minimal human assistance; (3) **Augmentation**: an LLM agent augments the dataset such that the task instructions are aligned to real-world applications. In *Stage 2*, with the dataset prepared, the SLM is initially trained using SFT and further optimized via GRPO with

rewards provided by the LLM reward function.

3.2 Task Instruction Set Generation

The goal of task instruction generation is to create a set of executable task instructions for robot operations. To achieve this, we leverage reasoning and the context generation capabilities of LLM to infer feasible robot operations and generate a set of task instructions. Formally, we view the LLM agent as implementing a function $\mathcal{F}_1 : (P, D) \rightarrow I$, where P denotes the robot operational tasks policies and D is task instruction demonstrations. Given (P, D), the LLM generates a set of task instructions I.

Task Generation: To prevent LLM hallucination that could produce non-executable tasks, we employ in-context learning to guide the LLM toward reliably producing valid task instructions. Specifically, we construct a system prompt to enhance a foundational LLM. First, we define the LLM's role of generating robot operational task instructions. Next, we provide the policies P for robot operations and task instruction formats. These policies are essential for the LLM to understand the robot's capabilities and operational constraints. Moreover, the policies enforce the LLM to produce clear and concise instructions, thereby minimizing ambiguity that could lead to misinterpretation in the subsequent code generation. Finally, we construct k demonstrations of task instructions, which cover the fundamental robot operation patterns and standardized instruction formats. The LLM learns to imitate and recombine these patterns while generating new task instructions.

Task Distribution: After configuring the foundational LLM with the above structured system prompt, we prompt the LLM agent to generate n task instructions. To increase diversity, we further design multiple system prompt configurations for varying task complexity distributions. As a result, the instruction set I spans multiple levels of complexity, ranging from few-stage planning and basic spatial reasoning (one or two actions) to multi-stage complex reasoning (five or more actions require complex reasoning of the robot's dynamics). The diversity ensures that the SLM is fine-tuned on a broad task distribution, which enables the SLM to develop robust and generalizable reasoning capabilities across varying task complexity scenarios.

3.3 Ground Truth Code Generation

To reduce human effort, we leverage the reasoning and planning capabilities of LLMs to ground each instruction in I into code. Specifically, the LLM agents implement a grounding function $\mathcal{F}_2 : I \rightarrow C$, which maps each task instruction in $I_i \in I$ to its corresponding ground truth code C_i.

LLM Code Generation: To ensure reliable instruction grounding, we adopt the corrective code generation framework proposed in (Wang et al., 2025b), which identifies and corrects errors in the generated code before producing the final version C_i for each task instruction I_i.

Code Correctness Control: While LLMs can reliably generate correct code for simple tasks, their performance degrades on tasks requiring complex spatial reasoning or long-horizon planning (Wang et al., 2026, 2025b). Therefore, fully LLM-based grounding may introduce errors in C, leading to the SLM being fine-tuned on incorrect supervision. To ensure the correctness of C, we adopt a hybrid code generation strategy, in which human oversight is used for code generation for tasks that involve five or more steps of action. In these tasks, a human expert reviews the generated code and corrects any errors with respect to the task instructions.

3.4 Augmentation

As discussed in Section 3.2, the instruction set I must be clear and concise to prevent misinterpretation during subsequent code generation, which makes the task instructions expressed in high-level operational plans (e.g., *"Fly 5 meters up, then fly 4 meters down."*). However, such high-level plans usually differ from the way humans typically describe tasks in real-world settings, such as the real-world scenario example presented in Figure 2. To bridge this gap, we augment the dataset by mapping the task instructions into robot application scenarios. Specifically, we leverage the world knowledge in LLMs to rephrase each instruction to a description of a corresponding real-world robot deployment. The augmentation function is defined as $\mathcal{F}_3 : (I, C) \rightarrow (\hat{I}, C)$, where $(\hat{I}, C)$ represent the augmented instruction and corresponding code pairs.

Augmentation Quality Control: The primary challenge in dataset augmentation is to ensure that each augmented instruction $\hat{I}_i$ describes the same robot operations as the original instruction I_i, such that the code C_i matches with the augmented tasks

during SLM fine-tuning. To achieve this, we leverage in-context learning of LLMs again. We construct a system prompt that imposes rules on the task-mapping process to prevent introducing additional robot actions, and the $\hat{I}$ should be feasible real-world robot operations. Moreover, to avoid overly verbose and unnatural generations (Zhang et al., 2025), LLM is prompted to use common vocabulary, one paragraph description, and human-like tones in its augmentation.

3.5 Dataset Construction

We construct the final fine-tune dataset by applying $\mathcal{F}_1$, $\mathcal{F}_2$, $\mathcal{F}_3$ in sequence. The final dataset S consists of paired instructions and code, denoted as $S = (I \cup \hat{I}, C)$. Furthermore, since the SLMs are sensitive to the syntactic patterns of fine-tuning data and their performance degrades under syntactic contamination (Scaria et al., 2025), we remove all comments from the code in the dataset. This data preprocessing strategy reduces the risk that the SLM overfits to linguistic patterns rather than the code, which ensures the fine-tuned SLM focuses on producing robot operational code when given a task instruction.

3.6 SLM Fine-tune

Given the dataset S, the fine-tune process trains an SLM M to implement $\mathcal{F} : (M, I') \rightarrow C'$, such that M generates robot operation code C' when provided with a task instruction I'. The fine-tuning procedure consists of two steps:

Supervised Fine-Tuning: The M is fine-tuned on the dataset S to align the SLM with the desired code generation style and enables it to learn robot actions (i.e., APIs) and their mappings from task instructions to executable code. Additionally, we employ LoRA (Hu et al., 2022) to improve fine-tuning efficiency and reduce memory usage.

Optimization with LLM Feedback: In the second stage, we further optimize M to enhance step-level reasoning by GRPO (Shao et al., 2024) with LLM as reward function. Unlike text-generation settings, where being semantically similar to the ground truth is often sufficient for a high reward, in robotic code generation, even semantically similar code may implement different robot actions. Conversely, semantically dissimilar expressions may correspond to the same underlying operation (e.g., variable "position" and "base" both referring to the robot's location). To address these issues, we configure an LLM to interpret the robot behavior implied by the code generated by the SLM and compare it with the ground truth. The LLM then outputs feedback indicating whether the generated code executes the same robot actions as the ground truth. This feedback is then constructed as a reward for GRPO to optimize the SLM's weights.

4 Experiments

4.1 Experimental Setup

4.1.1 Implementation

We implement the dataset synthesis in our proposed Ro-SLM framework using OpenAI GPT models. Specifically, task instruction generation and augmentation are performed with GPT-5.1 (OpenAI Models, 2025a), while ground truth code generation is conducted using o4-mini (OpenAI Models, 2025b). For the training set, we synthesize 203 task instructions, of which 150 are automatically grounded into code, and 53 require human assistance for grounding. Dataset augmentation results in a final dataset that consists of 598 instruction-code pairs. The dataset is then split into 492 pairs for training T and 106 pairs for evaluation E.

The fine-tuning of Ro-SLM is implemented with Llama-3.1-8B (Grattafiori et al., 2024) from Hugging Face (Jain, 2022), which can be deployed for inference on small robots with their onboard processing units (e.g., Jetson AGX Orin). In the optimization, the reward function is implemented by GPT-5.1 (OpenAI Models, 2025a). Both training and optimization of Ro-SLM are conducted on a server with two Nvidia A100 GPUs.

We use the pre-built "block" (Research, 2025) environment of the AirSim simulator (Shah et al., 2018). All results are averaged over three runs to mitigate the randomness of LLM generation (Atil et al., 2024).

4.1.2 Baselines

Three methods are selected as the baselines for comparison:

GSCE (Wang et al., 2025a): This method configured a foundational o4-mini with a structured system prompt, enabling the LLM to generate UAV operation code according to human instructions.

Corrective GSCE (Wang et al., 2025b): The state-of-the-art (SOTA) method for robot operation code generation. Build upon GSCE; the code is iteratively refined towards achieving the task objectives, which is also the implementation of ground

truth code generation in section 3.3. Notably, all the LLMs in this method use o4-mini.

Llama-3.1-8B: This baseline uses the original SLM without fine-tuning. The original Llama-3.1-8B (Grattafiori et al., 2024) downloaded from Huggingface is configured with the same structured system prompt as GSCE (Wang et al., 2025a) to act as an agent for UAV operation code generation, which has been proven effective on LLMs.

4.1.3 Test Datasets

The Basic and Advanced datasets from (Wang et al., 2026) are selected as benchmark datasets for a comprehensive evaluation of the model's reasoning capabilities across varying levels of task complexity. The Basic dataset consists of 44 intuitive UAV operation task instructions and each involves one or two actions. They are designed to assess a few-stage planning and basic logical reasoning. The Advanced dataset contains 20 long-horizon UAV operation tasks with four levels of complexity ("*Low*", "*Medium*", "*High*", and "*Very High*"). Each advanced task requires complex reasoning about UAV dynamics in the geometric world in order to make correct decisions. Notably, all tasks in both datasets are manually validated to prevent potential simulation-induced errors that could affect the experiment result.

The above tasks evaluate the model's ability to reason about task logic and follow instructions. To assess robustness, we further extend both datasets by mapping the original instructions to real-world application scenarios, which assesses whether the models are able to reliably respond to task instructions that are more natural and practical. The extended datasets are provided in Appendix D.

4.1.4 Evaluation Metrics

We evaluate performance using **Completeness** and **Success Rate (SR)** (Wang et al., 2026). The ground truth is represented by a sequence of UAV state transitions. Each state transition is a vector of four elements: $[x, y, z, \theta]$, where x, y, and z denote the robot's position transitions in the North, East, and Down axes, and θ represents the yaw rotation.

Completeness measures the proportion of actions that are executed correctly in a given task. Given a task i, its Completeness is defined as Eq. 1, i.e., the ratio between the number of correctly executed actions (a_i^{correct}) and the total number of ground truth actions (l_i^{gt}) in the task. This metric provides insight into reasoning performance throughout the intermediate code execution process.

$$\text{Completeness}_i = \frac{|a_i^{\text{correct}}|}{|l_i^{\text{gt}}|} \tag{1}$$

SR captures task-level performance by measuring whether the robot successfully reaches the final goal state while correctly following intermediate actions. Given a task i, it is considered successful (SR_i=1) if and only if the entire action sequence is executed without error, i.e., $Completeness_i = 1$; otherwise, SR_i=0.

4.2 Results

4.2.1 Overall Result

The overall results presented in Table 1 indicate that the SLM fine-tuned by Ro-SLM achieves comparable performance to the LLM. Specifically, Ro-SLM obtains 98.9% Completeness and 97.7% SR for the Basic tasks; 83.7% Completeness and 70.0% SR for the Advanced tasks, which are close to the performance achieved by GSCE. In contrast, applying the same prompting strategies as GSCE to the original Llama-3.1-8B is incapable of producing accurate robot operation code; in our experiment, the model either repeats the task instructions in its generation or produces incorrectly reasoned code.

Moving on to the real-world instruction mapping results in Table 2. On the Basic tasks, the Ro-SLM achieves 95.5% Completeness and 93.2% SR. For the Advanced tasks, Ro-SLM shows improvement; it obtains 87% Completeness and 75% SR, which reduces the performance gap with the LLM on long-horizon tasks. This improvement is attributed to the dataset augmentation process, which exposes the SLM to a diverse range of real-world phrasing and thus enhances its robustness to human tone task instructions.

Compared with the original Llama-3.1-8B, Ro-SLM significantly improves it from being incapable to LLM-like performance in all settings. While Ro-SLM has a relatively lower performance compared with the Corrective GSCE on the Advanced tasks, it is designed to balance its reasoning capability and onboard deployability for resource-constrained environments. The SOTA performance of Corrective GSCE is achieved by its iterative refinement strategy that requires three LLMs. Integrating such a strategy, even with SLMs, will exceed the capability of onboard units of most small robot platforms.

	Basic		Advanced	
Method	SR	Completeness	SR	Completeness
Llama-3.1-8B (Grattafiori et al., 2024)	9.1%	9.1%	5%	9%
GSCE (Wang et al., 2025a)	100%	100%	75.0%	91.5%
Corrective GSCE (Wang et al., 2025b)	100%	100%	90.0%	97.8%
Ro-SLM with Comments	95.5%	96.6%	50.0%	76.2%
Ro-SLM without Augmentation	95.5%	96.6%	55.0%	74.8%
Ro-SLM Simple Task Only	90.9%	93.2%	0%	1.9%
Ro-SLM Complex Task Only	0%	2.3%	75%	83.8%
Ro-SLM without GRPO Optimization	97.7%	98.9%	70.0%	83.1%
Ro-SLM	97.7%	98.9%	70.0%	83.7%

Table 1: Results on the Basic and Advanced datasets.

	Basic		Advanced	
Method	SR	Completeness	SR	Completeness
Llama-3.1-8B (Grattafiori et al., 2024)	11.4%	26.1%	0%	8.4%
GSCE (Wang et al., 2025a)	100%	100%	75.0%	94.8%
Corrective GSCE (Wang et al., 2025b)	100%	100%	81.7%	96.3%
Ro-SLM with Comments	90.9%	93.2%	55.0%	75.2%
Ro-SLM without Augmentation	88.6%	92.0%	70.0%	81.3%
Ro-SLM Simple Task Only	90.9%	94.3%	0%	3.6%
Ro-SLM Complex Task Only	0%	2.3%	70%	83.0%
Ro-SLM without GRPO Optimization	93.2%	95.5%	75%	86.4%
Ro-SLM	93.2%	95.5%	75%	87.0%

Table 2: Results on the real-world mapping of Basic and Advanced datasets.

Overall, these results demonstrate that prompting strategies effective for LLMs cannot transfer to SLMs. By distilling LLM's reasoning and knowledge into SLM, Ro-SLM can substantially improve the reasoning capability and robustness of SLMs and enable SLM to approach the performance of LLM on robotic task planning and code generation.

4.2.2 Ablation Study

An ablation study is performed to analyze the contribution of each component in our Ro-SLM framework. Specifically, we construct variants of Ro-SLM that retain code comments *(Ro-SLM with Comments)*, omit augmentation *(Ro-SLM without Augmentation)*, reduced task diversity *(Ro-SLM Simple Task Only or Complex Task Only)*, and remove GRPO optimization *(Ro-SLM)*. The results are presented in Tables 1 and 2.

Comments Although the SLM fine-tuned on datasets that retain code comments achieves competitive performance on the Basic tasks, its SR is reduced by 20% on the Advanced tasks. While comments generated by LLMs are useful for human interpretation, they introduce diverse linguistic patterns. For the basic tasks, where the code is short, the comments did not affect. However, Advanced tasks involve longer action sequences, and code comes with more extensive comments. Since SLMs are sensitive to the syntactic patterns of the training data (Scaria et al., 2025), this variability introduces noise that degrades performance. Accordingly, we remove all code comments in our synthetic dataset to improve the robustness of the fine-tuned SLM.

Augmentation Augmentation improves both SR and Completeness by at least 2% on Basic tasks, and more than 5% on Advanced tasks. These gains indicate that exposing the SLM to more diverse and naturalistic task instructions during training enhances its robustness in both few-stage and long-horizon reasoning. Therefore, we incorporate dataset augmentation to improve the SLM's generalization across varying instruction styles.

	SR		Completeness	
Advanced	GSCE (Wang et al., 2025a)	Ro-SLM	GSCE (Wang et al., 2025a)	Ro-SLM
Low	100%	66.7%	100%	87.9%
Medium	88.9%	77.8%	97.8%	86.9%
High	75.0%	75.0%	88.9%	75.6%
Very High	25.0%	50.0%	73.7%	81.7%
Real-world Mapping				
Low	100%	66.7%	100%	87.9%
Medium	88.9%	88.9%	98.9%	95.6%
High	75.0%	75.0%	91.7%	72.6%
Very High	25.0%	50.0%	84.8%	81.7%

Table 3: Comparison of SR & Completeness on different complexities: advanced tasks and their real-world mapping.

Task Diversity As shown in the results, the SLM performs well on Basic tasks but poorly on Advanced tasks when trained exclusively on a simple-task-only dataset, and exhibits the opposite behavior when trained only on complex tasks. This pattern indicates that SLMs adapt to the distribution of the fine-tuning data and tend to overfit to a narrow range of task distribution. In contrast, SLMs trained on a diverse distribution of tasks exhibit more robust performance across varying task types. These results demonstrate the importance of incorporating multiple task-complexity distributions in the dataset synthesis process to ensure the generalization and robustness of fine-tuned SLMs.

Optimization Removing the optimization step does not affect performance on the Basic tasks, but it reduces the Completeness on the Advanced tasks by 0.6%. This suggests that GRPO improves step-level reasoning in long-horizon tasks by refining intermediate decision-making, as the reward function encourages step-wise equivalence with the ground-truth code during optimization. Consequently, we incorporate optimization in our framework to enhance SLM's intermediate step reasoning.

4.3 Analysis over Task Complexity

We next analyze the performance of the SLM and the LLM on the Advanced tasks across the four complexity levels (Wang et al., 2026), as shown in Table 3.

SR As task complexity increases, the SR of the LLM decreases with the increment of complexity, whereas SLM exhibits a different trend. On the original Advanced tasks, the SLM underperforms the LLM at the *Low* level, performs comparably at *Medium* and *High* levels. Notably, the SLM outperforms the LLM under *Very High* level. The improvement is attributed to the availability of the accurate ground-truth code provided through human assistance during dataset construction, which enables the SLM to learn correct complex long-horizon behaviors. These results indicate that the SLM effectively learns mapping tasks to code from the dataset and generalizes to varying task complexities.

Completeness The SLM approaches the performance of LLM across the *Low*, *Medium*, and *High* levels. Although the SLM substantially outperforms the LLM at the *Very High* level in SR, its Completeness remains similar to the LLM. This suggests that while the LLM fails to correctly reason about critical steps in highly complex tasks, the SLM is able to handle these steps more correctly, thereby improving task success.

Overall, although SLMs have inherently more limited reasoning capabilities (Chen et al., 2025c), these results demonstrate that Ro-SLM substantially enhances the ability of SLMs to perform complex and long-horizon reasoning robotic tasks at a level comparable to LLM.

4.4 Experiment on Ground Vehicle

To further evaluate the generalization capability of our framework, we extend Ro-SLM to a different robotic platform, a ground vehicle with different APIs and constraints for robot operations. Following the same pipeline, the framework first synthesizes task instructions, generates corresponding ground-truth code, and performs dataset augmentation with respect to the ground vehicle's operations. The trained SLM is then evaluated on eight ground-vehicle operation tasks adopted from (Wang et al.,

	SR	Completeness
Corrective GSCE (Wang et al., 2025b)	87.5%	97.2%
Ro-SLM	75%	81.1%

Table 4: Results on ground vehicle.

2025b), which control a ROSMASTER X3 to perform operations requiring complex reasoning. As shown in the Table 4, by leveraging Ro-SLM, the SLM approaches the LLM performance on ground vehicle operations. This suggests that Ro-SLM captures robot task planning rather than overfitting to platform-specific APIs, and demonstrates its potential to generalize across heterogeneous robotic systems.

5 Conclusion

In this paper, we present Ro-SLM, a framework that distills LLMs' reasoning and knowledge into SLM to enable reliable task planning and code generation on resource-constrained robot platforms. Our framework first synthesizes a high-quality instruction-code dataset by utilizing LLMs in task instruction generation, corrective code grounding with minimal human assistance, and augmentation. Then the SLM is fine-tuned on the synthetic dataset and further optimized to the best result with LLM-aided optimization. Experiments on UAV and ground vehicle operation tasks show that the SLM in our framework substantially outperforms the original SLM and approaches the LLM-level performance across varying task complexities.

Limitations

Despite the promising results demonstrated by our framework, several limitations could be addressed in future research:

Generalization and Real-World Deployment The dataset synthesis is highly task-specific, and transferring our framework across different domains remains challenging. Moreover, although Ro-SLM performs well in a simulated environment, its robustness in real-world robotic systems with unpredictable environmental variables requires further validation.

Computational Scalability Although Ro-SLM substantially improves the performance of Llama-3.1-8B, the model still requires approximately 16GB of memory, which limits deployment on compact edge devices with limited memory capacity. While smaller SLMs (e.g., 3B or 1B models) would fit on a broader range of devices, their performance within the Ro-SLM framework remains to be explored.

Human Involvement in Data Generation While code generation for simple tasks can be automated, minimal human assistance is still needed to verify and correct ground-truth code for high-complexity tasks. This process requires domain expertise and introduces manual overhead, which limits the scalability of dataset construction to new domains and to larger or more complex datasets.

Lack of Iterative Correction. Unlike LLM-based approaches such as Corrective GSCE (Wang et al., 2025b) that employ multi-round refinement, Ro-SLM relies on a single-pass generation at inference time. While this design improves efficiency and deployability, it limits the model's ability to correct intermediate reasoning errors, particularly in complex multi-step tasks.

In summary, these limitations suggest that while Ro-SLM provides a strong foundation for enabling onboard language-driven robot control, further work will focus on improving scalability, robustness, and adaptability across diverse real-world robotic platforms.

Ethical Considerations

As language model-driven robotic systems become increasingly integrated into real-world deployment, the proposed framework also raises ethical considerations in safety. The generated code controls robotic operation and therefore carries risks of physical harm if improperly deployed. Errors in execution could result in collisions, property damage, or injury to humans. Transferring to real-world robots should include additional safeguards, such as an emergency stop.

Acknowledgments

This work was supported by the US National Science Foundation awards 2318710 and 2318711, and the UMass Dartmouth Internal Research Seed Funding Program.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. [gpt]-4 technical report. *arXiv preprint arXiv:2303.08774*.

Montserrat Gonzalez Arenas, Ted Xiao, Sumeet Singh, Vidhi Jain, Allen Ren, Quan Vuong, Jake Varley, Alexander Herzog, Isabel Leal, Sean Kirmani, Mario Prats, Dorsa Sadigh, Vikas Sindhwani, Kanishka Rao, Jacky Liang, and Andy Zeng. 2024. How to prompt your robot: A promptbook for manipulation skills with code as policies. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4340–4348.

Berk Atil, Alexa Chittams, Liseng Fu, Ferhan Ture, Lixinyu Xu, and Breck Baldwin. 2024. Llm stability: A detailed analysis with some surprises. *arXiv preprint arXiv:2408.04667*.

Guojun Chen, Xiaojing Yu, Neiwen Ling, and Lin Zhong. 2025a. Typefly: Low-latency drone planning with large language models. *IEEE Transactions on Mobile Computing*, 24(9):9068–9079.

Yongchao Chen, Jacob Arkin, Charles Dawson, Yang Zhang, Nicholas Roy, and Chuchu Fan. 2024. Autotamp: Autoregressive task and motion planning with llms as translators and checkers. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6695–6702.

Yuxuan Chen, Yixin Han, and Xiao Li. 2025b. Fastnav: Fine-tuned adaptive small-language- models trained for multi-point robot navigation. *IEEE Robotics and Automation Letters*, 10(1):390–397.

Zihan Chen, Song Wang, Zhen Tan, Xingbo Fu, Zhenyu Lei, Peng Wang, Huan Liu, Cong Shen, and Jundong Li. 2025c. A survey of scaling in large language model reasoning. *arXiv preprint arXiv:2504.02181*.

Abraham George and Amir Barati Farimani. 2025. Llm trainer: Automated robotic data generating via demonstration augmentation using llms. *arXiv preprint arXiv:2509.20070*.

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*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Zichao Hu, Junyi Jessy Li, Arjun Guha, and Joydeep Biswas. 2025. Robo-instruct: Simulator-augmented instruction alignment for finetuning code LLMs. In *Second Conference on Language Modeling*.

Shashank Mohan Jain. 2022. Hugging face. In *Introduction to transformers for NLP: With the hugging face library and models to solve problems*, pages 51–67. Springer.

Frank Joublin, Antonello Ceravola, Pavel Smirnov, Felix Ocker, Joerg Deigmoeller, Anna Belardinelli, Chao Wang, Stephan Hasler, Daniel Tanneberg, and Michael Gienger. 2024. Copal: Corrective planning of robot actions with large language models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8664–8670.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. 2023. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500.

Zhenyan Lu, Xiang Li, Dongqi Cai, Rongjie Yi, Fangming Liu, Xiwen Zhang, Nicholas D Lane, and Mengwei Xu. 2024. Small language models: Survey, measurements, and insights. *arXiv preprint arXiv:2409.15790*.

Ggaliwango Marvin, Nakayiza Hellen, Daudi Jjingo, and Joyce Nakatumba-Nabende. 2023. Prompt engineering in large language models. In *International conference on data intelligence and cognitive informatics*, pages 387–402. Springer.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

MG Mohanan and Ambuja Salgoankar. 2018. A survey of robotic motion planning in dynamic environments. *Robotics and Autonomous Systems*, 100:171–185.

OpenAI Models. 2025a. GPT-5.1. Accessed: 25-Dec-2025.

OpenAI Models. 2025b. o4-mini. Accessed: 25-Dec-2025.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Zachary Ravichandran, Ignacio Hounie, Fernando Cladera, Alejandro Ribeiro, George J Pappas, and Vijay Kumar. 2025. Distilling on-device language models for robot planning with minimal human intervention. *arXiv preprint arXiv:2506.17486*.

Microsoft Research. 2025. Setup blocks environment for airsim. `https://microsoft.github.io/AirSim/unreal_blocks/`. Accessed: 2-Feb-2025.

Nicy Scaria, Silvester John Joseph Kennedy, and Deepak Subramani. 2025. Sensitivity of small language models to fine-tuning data contamination. *arXiv preprint arXiv:2511.06763*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. 2018. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics: Results of the 11th International Conference*, pages 621–635. Springer.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Andrea Tagliabue, Kota Kondo, Tong Zhao, Mason Peterson, Claudius T Tewari, and Jonathan P How. 2024. Real: Resilience and adaptation using large language models on autonomous aerial robots. In *2024 IEEE 63rd Conference on Decision and Control (CDC)*, pages 1539–1546. IEEE.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, and 1 others. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, and 1 others. 2025. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*.

Sai H. Vemprala, Rogerio Bonatti, Arthur Bucker, and Ashish Kapoor. 2024. Chatgpt for robotics: Design principles and model abilities. *IEEE Access*, 12:55682–55696.

Wenhao Wang, Yanyan Li, Long Jiao, and Jiawei Yuan. 2025a. Gsce: a prompt framework with enhanced reasoning for reliable llm-driven drone control. In *2025 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 441–448.

Wenhao Wang, Yanyan Li, Long Jiao, and Jiawei Yuan. 2025b. Llm-driven corrective robot operation code generation with static text-based simulation. *arXiv preprint arXiv:2512.02002*.

Wenhao Wang, Yanyan Li, Long Jiao, and Jiawei Yuan. 2026. Large language model-driven closed-loop uav operation with semantic observations. *IEEE Internet of Things Journal*, 13(7):14465–14476.

Junjie Wen, Yichen Zhu, Jinming Li, Minjie Zhu, Zhibin Tang, Kun Wu, Zhiyuan Xu, Ning Liu, Ran Cheng, Chaomin Shen, Yaxin Peng, Feifei Feng, and Jian Tang. 2025. Tinyvla: Toward fast, data-efficient vision-language-action models for robotic manipulation. *IEEE Robotics and Automation Letters*, 10(4):3988–3995.

Bangguo Yu, Hamidreza Kasaei, and Ming Cao. 2023. L3mvn: Leveraging large language models for visual target navigation. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3554–3560.

Xianyang Zhan, Agam Goyal, Yilun Chen, Eshwar Chandrasekharan, and Koustuv Saha. 2025. Slm-mod: Small language models surpass llms at content moderation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8774–8790.

Yusen Zhang, Sarkar Snigdha Sarathi Das, and Rui Zhang. 2025. Demystify verbosity compensation behavior of large language models. In *Proceedings of the 2nd Workshop on Uncertainty-Aware NLP (UncertaiNLP 2025)*, pages 160–178.

Jinbiao Zhu, Lishan Wang, Yi Guc, and Fei Wang. 2024. Ofslms: Offline fine-tuned small language models based on hybrid synthetic knowledge for robot introspective decision-making. In *2024 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 1839–1844.

A Task Instruction Generation and System Prompts

For the training set, we design two distinct system prompts for task-instruction generation with different task complexities. The first prompt induces the LLM to generate 122 tasks containing fewer than five action steps. These tasks are relatively simple and intuitive, and are intended to establish the SLM's basic UAV operation capabilities and short-horizon reasoning skills. The second prompt induces the LLM to generate 42 tasks containing complex, long-horizon UAV operations, such as flying in a series of square patterns. These tasks require multi-stage planning and complex spatial reasoning, and are therefore intended to develop the SLM's long-horizon and logical reasoning capabilities.

For the evaluation set, we modify the number of tasks specified in the system prompts. We generate 28 tasks using the first system prompt and 11 tasks using the second system prompt.

In total, we construct 203 tasks spanning varying lengths, complexity levels, and task types for the fine-tune dataset.

System Prompt 1:

You are going to generate drone control tasks for me.

Rules for the task generation:
1. The drone tasks involve combinations of world frame movement, rotation, and the drone's body frame movement.
2. The task description must be concise. Do not add additional explanation with "()" in the task.
3. The task description must clearly state the coordinate system (world frame or drone's body frame) for each movement action.
4. The task description must clearly state the rotation direction, and the rotation angle must be divisible by 30.
5. Move and rotate are the only two actions available for the drone.
6. Movement distance should be an integer, the number should be larger than 2 meters and smaller than 10 meters.
7. No more than 5 steps of actions in a task.
8. The task description should be in a human tone.

Here are four example tasks:
1. Fly 3 meters up, then fly 5 meters down in the world frame.
2. Rotate 180 degrees, then fly 5 meters forward in the drone's body frame.
3. Turn to face the local south, then fly 6 meters forward in the drone's body frame.
4. Fly the drone in the top-right direction at an angle of 60 degrees from the horizontal axis, in the YZ plane of the drone's body frame for a distance of 5 meters.

Your output should be tasks only.

Please generate 110 tasks like examples and 12 tasks that fly the drone in XZ or YZ plane like example 4 in the drone's body frame.

System Prompt 2 Part A:

You are going to generate drone control tasks for me.

Rules for the task generation:
1. The drone tasks involve combinations of movement and rotation.
2. The task description must be concise and clear. Do not add additional explanation with "()" in the task.
3. The drone is going to fly a series of square patterns. The patterns involve flying forward, right, backward, and left; flying forward, left, forward, and right; symmetric; reverse; two or more squares; figure of 8.
4. State the purpose, if the task requires a specified facing direction (align, opposite, perpendicular).
5. Move and rotate are the only two actions available for the drone.
6. Movement distance should be an integer, the number should be larger than 2 meters and smaller than 10 meters.
7. The task description should be in a human tone.

Here are four example tasks:
1. Take off and fly up 5 meters. You should fly in a square pattern with 5-meter sides by moving north, east, south, and west in the world axis. 2. Take off and fly up 5 meters. You will examine a square area. You should fly in a square pattern with 5-meter sides by moving forward, left, backward, and right. To examine the square area, the drone should orientate perpendicular to the moving direction on each side of the square.
3. Take off and fly up 5 meters. You will examine a square area. You should fly in a square pattern with 5-meter sides by moving forward, right, backward, and left. To examine the square area, the drone should orientate perpendicular to the moving direction on each side of the square. Next, ascend another 5 meters and fly the square pattern in reverse order to examine the same area.

System Prompt 2 Part B:

4. Take off and fly up 5 meters. Fly in a square with 5-meter sides. The movement pattern should follow this sequence: forward, right, backward, and left in the world axis. Next, fly a second square that is symmetric with respect to the X-axis in the XY plane. To examine the two square areas, the drone should orientate perpendicular to the moving direction on each side of the squares.
5. Take off and fly up 5 meters. Fly a figure of 8 on a flat, horizontal plane with each side of 5 meters. The left square is on your left-rear side, and the right square is on your right-front side. You should begin with the left square by flying left. The right square should start from moving north. Additionally, for the left square, the drone is oriented in the flying direction; for the right square, the drone should examine the area while flying the right square pattern. A good strategy for examining is the drone orientate perpendicular to the moving direction on each side of the right square.

Your output should be tasks only.

Generate 42 tasks according to the flight path and pattern from the examples I gave you. You should cover all the patterns in the examples.

B Task Augmentation System Prompt

System Prompt:

You are an assistant for thinking up a real-world drone operation task that implements the same drone moving actions as the given task, such that the task description is more realistic for real-world applications.

Here are some rules for you:
1. Make sure to state the coordinate system (world frame or drone's body frame) for each movement action in your modified task.
2. Your task must perform the same action as the task I gave you. You must not add any additional actions.
3. You must not introduce misleading descriptions that could be interpreted as additional actions.
4. Your output should be human tone-like and should not use uncommon words.
5. Must output your modified task in one paragraph.

Here is an example:
Query: "Fly 5 meters up, then fly 4 meters down."
Answer: "Perform a vertical clearance check near a storage. In the world frame, ascend 5 meters to inspect the upper vent, then descend 4 meters in the world frame to position the drone near the mid-section for a closer look."

C Reward Function and System Prompt

Because code generation for the same task may differ syntactically, direct comparison based on textual similarity is impractical. For example, one generation uses "position" as a variable name and another generation uses "base", although they both refer to the drone's location, their textual representations differ. We therefore adopt the code interpretation strategy from (Wang et al., 2025b) to assess whether the SLM-generated code and the ground truth code produce the same operational outcome. If the SLM-generated code fully matches the ground truth behavior, the reward is set to 1; otherwise, it is set to 0.

System Prompt:

You will compare the intentions of the given drone control code with the ground truth code.

You should focus on the code, not the comments. Because code will be the actual actions of the drone.

Here are the available functions for the drone when you interpret the code.
aw.takeoff() - takes off the drone.
aw.land() - lands the drone.
aw.fly_to([x, y, z]) - flies the drone to the position specified as a list of three arguments corresponding to world XYZ coordinates. The flying speed is 2 meters per second.
aw.get_yaw() - returns the current yaw of the drone in degrees.
aw.set_yaw(yaw) - sets the yaw of the drone to the specified value in degrees.
aw.get_drone_position() - returns the current position of the drone as a list of 3 floats corresponding to world XYZ coordinates.

Your answer must be 0 or 1, where 0 means the code does not match the ground truth, and 1 means it fully matches.

D Task Dataset Real-World Mapping

Task	Scenario
Fly 5 meters up, then fly 4 meters down.	Please perform a vertical inspection of the warehouse wall. In the world frame, ascend 5 meters to check the condition of an overhead ventilation duct, then descend 4 meters in the world frame to inspect a mid-height access panel more closely.
Fly 5 meters down, then fly 4 meters up.	You are flying under a bridge to inspect structural damage. In the world frame, descend 5 meters to move from the initial hover position down toward the lower girder area for a detailed scan, then ascend 4 meters in the world frame to bring the drone closer to the mid-level of the bridge structure for follow-up imaging.
Fly 5 meters up, then fly 4 meters up.	You are going to conduct a rooftop antenna inspection at a warehouse. In the world frame, ascend 5 meters to rise above nearby obstacles and reach the general rooftop level, then ascend another 4 meters in the world frame to perform a visual check of the antenna assembly.
Fly 5 meters down, then fly 4 meters down.	The drone is flying around a tall warehouse. In the world frame, descend 5 meters to move from the overhead inspection position down toward the shelf, then continue descending 4 meters in the world frame to inspect the lower shelves and base of the rack more closely.
Rotate 180 degrees, then fly 4 meters forward in the drone's body frame.	Perform a corridor inspection. First, in the drone's body frame, rotate 180 degrees to face the opposite direction down the hallway. Then, still in the drone's body frame, fly 4 meters forward to continue the inspection along the newly faced segment of the corridor.
Rotate 180 degrees, then fly 4 meters backward in the drone's body frame.	The drone is going to conduct a reverse-approach inspection. First, in the drone's body frame, rotate 180 degrees to face the support beams behind the drone, then in the drone's body frame, fly 4 meters backward to carefully reposition closer to the rear side of the structure for detailed imaging.
Rotate 180 degrees, then fly 4 meters right in the drone's body frame.	The drone is going to fly over the fence of the construction site perimeter. In the drone's body frame, first rotate 180 degrees to turn from facing the inner side to facing the outer fence line, then fly 4 meters to the right in the drone's body frame to laterally reposition along the fence for a side-on visual inspection.
Rotate 180 degrees, then fly 4 meters left in the drone's body frame.	You are going to inspect the side of the warehouse aisle. First, in the drone's body frame, rotate 180 degrees to face the shelving on the opposite side of the aisle. Then, still in the drone's body frame, fly 4 meters to the left to capture images of the adjacent shelf segment.

Turn to face the local south, then fly 4 meters forward in the drone's body frame.	The drone is going to perform an inspection of rooftop solar installation along a building's southern edge. First, in the world frame, turn the drone until its front faces local south to align with the panel row, then in the drone's body frame, fly 4 meters forward to approach the center of the southern panel array for closer inspection.
Turn to face the local south, then fly 4 meters backward in the drone's body frame.	Now, perform an indoor aisle inspection in a warehouse. First, in the world frame, yaw the drone to face local south to align with the aisle direction. Then, in the drone's body frame, fly 4 meters backward to slowly increase the distance from a tall storage rack while maintaining the camera's view of its shelves for documentation.
Turn to face the local south, then fly 4 meters right in the drone's body frame.	Fly along a rectangular crop field boundary. First, in the world frame, rotate the drone until its front aligns with south to match the survey heading. Then, in the drone's body frame, translate 4 meters to the drone's right side to position it laterally over the field's inner edge for side-view imaging of the boundary.
Turn to face the local south, then fly 4 meters left in the drone's body frame.	Perform a short side-clearance survey along a building. First, in the world frame, rotate the drone so that its forward-facing camera points toward the local south. Then, move 4 meters to the left in the drone's body frame to inspect the adjacent wall section while keeping the same south-facing view.
Turn 90 degrees clockwise, then fly 4 meters forward in the drone's body frame.	You are going to inspect the side of a building. First, in the world frame, rotate the drone 90 degrees clockwise to face the adjacent wall. Then, in the drone's body frame, fly 4 meters forward to approach the wall and capture detailed images of the wall.
Turn 90 degrees clockwise, then fly 4 meters backward in the drone's body frame.	Conduct an indoor aisle inspection in a warehouse. First, in the world frame, rotate the drone 90 degrees clockwise to align it with a side aisle. Then, in the drone's body frame, fly 4 meters backward to safely reverse along that aisle while keeping the camera oriented toward the shelves being inspected.
Turn 90 degrees clockwise, then fly 4 meters right in the drone's body frame.	The drone is inspecting the side of a house. First, in the world frame, yaw the drone 90 degrees clockwise to align its front camera with the side wall. Then, in the drone's body frame, fly 4 meters to the right to scan along the wall section while maintaining the same distance from the structure.
Turn 90 degrees clockwise, then fly 4 meters left in the drone's body frame.	The drone is inspecting the side of a wall while maintaining a fixed position relative to the structure. First, in the world frame, rotate the drone 90 degrees clockwise so its camera faces along the wall. Then, in the drone's body frame, translate 4 meters to the left along the wall and capture images of the adjacent wall section.

Turn to face the local east, then fly 4 meters forward in the drone's body frame.	Conduct a short inspection along a known east-west power line. In the world frame, rotate the drone points toward the local east to align with the line's direction, then in the drone's body frame, fly 4 meters forward along its nose direction to advance to the next inspection point.
Turn to face the local east, then fly 4 meters backward in the drone's body frame.	Perform an alignment maneuver during a perimeter survey of a construction site. In the world frame, yaw the drone until its nose points toward local east to match the planned heading, then in the drone's body frame, fly 4 meters backward to fly away.
Turn to face the local east, then fly 4 meters right in the drone's body frame.	Perform a short inspection along a house wall. First, in the world frame, yaw the drone to face local east so its camera aligns with the wall's length, then, maintaining the drone's heading, move 4 meters to the right in the drone's body frame to scan along the wall while keeping its orientation fixed.
Turn to face the local east, then fly 4 meters left in the drone's body frame.	Inspect the exterior of a wall. First, in the world frame, yaw to align the drone so its nose points toward the local east to face the wall directly. Then, maintaining altitude and orientation, fly 4 meters to the left in the drone's body frame to laterally scan along the wall for structural damage.
Turn 90 degrees counterclockwise, then fly 4 meters forward in the drone's body frame.	Perform a corridor inspection inside a warehouse. In the world frame, first yaw 90 degrees counterclockwise to align the drone with the inspection aisle. Then, in the drone's body frame, fly 4 meters forward along the aisle to approach the designated inspection point.
Turn 90 degrees counterclockwise, then fly 4 meters backward in the drone's body frame.	Conduct an indoor inspection for a bookshelf. First, in the world frame, turn the drone 90 degrees counterclockwise to align its camera with a side shelf row. Then, in the drone's body frame, fly 4 meters backward along its longitudinal axis to slowly increase the distance from the shelf while maintaining the same viewing angle for documenting the entire rack.
Turn 90 degrees counterclockwise, then fly 4 meters right in the drone's body frame.	Perform an indoor aisle inspection in a warehouse. First, in the world frame, rotate the drone 90 degrees counterclockwise to align its forward-facing camera with a perpendicular storage aisle. Then, in the drone's body frame, translate 4 meters to the right to laterally shift along the aisle and inspect the shelf sections on that side.
Turn 90 degrees counterclockwise, then fly 4 meters left in the drone's body frame.	Perform indoor flying in a shop. First, in the world frame, yaw the drone 90 degrees counterclockwise to align its front with the target shelving row. Then, in the drone's body frame, translate 4 meters to the left to move laterally along the aisle and capture side-view images of the shelf contents.

Turn to face the local west, then fly 4 meters forward in the drone's body frame.	Inspect a solar panel on a rooftop. In the world frame, rotate the drone's nose points toward the local west to align with the panel row, then, in the drone's body frame, fly 4 meters forward along its current heading to move along the row for a closer inspection.
Turn to face the local west, then fly 4 meters backward in the drone's body frame.	You are flying around a barn. First, in the world frame, yaw the drone to face local west so its camera can align with the target wall. Then, in the drone's body frame, fly 4 meters backward to increase standoff distance from the wall while keeping the same heading for stable visual inspection.
Turn to face the local west, then fly 4 meters right in the drone's body frame.	Inspect a solar farm row alignment. First, in the world frame, rotate the drone until it faces local west to align with the panel rows. Then, in the drone's body frame, fly 4 meters to the right to shift over to an adjacent lane while maintaining the same heading.
Turn to face the local west, then fly 4 meters left in the drone's body frame.	Inspect the side of a wall. In the world frame, yaw the drone to face local west to align its camera with the wall, then in the drone's body frame, translate 4 meters to the left to scan along the wall while maintaining the same orientation.
Fly the drone in the top-right direction at an angle of 30 degrees from the horizontal axis, in the YZ plane of the drone's body frame for a distance of 10 meters.	Inspect the structural integrity of a roof corner joint. In the drone's body frame, in the YZ plane, move the drone 10 meters in the top-right direction at an angle of 30 degrees from the horizontal axis (within that YZ plane) so the onboard camera can capture the upper-right edge of the wall-ceiling junction for detailed imaging.
Fly the drone in the top-left direction at an angle of 30 degrees from the horizontal axis, in the YZ plane of the drone's body frame for a distance of 10 meters.	Survey the upper-left corner of a warehouse shelf for inventory inspection. In the drone's body frame, move in the YZ plane by flying 10 meters in a direction 30 degrees above the horizontal axis toward the top-left relative to the drone's forward-facing orientation, I would like to see what on the path.
Fly the drone in the bottom-right direction at an angle of 30 degrees from the horizontal axis, in the YZ plane of the drone's body frame for a distance of 10 meters.	Inspect a vertical cable fastened on a building corner. In the drone's body frame, move 10 meters in the bottom-right direction within its YZ plane at an angle of 30 degrees from the horizontal axis to follow the slanted cable.
Fly the drone in the bottom-left direction at an angle of 30 degrees from the horizontal axis, in the YZ plane of the drone's body frame for a distance of 10 meters.	Inspect a shelf corner for structural damage: starting from a hover near the aisle, fly the drone in the drone's body frame YZ plane by moving 10 meters in the bottom-left direction relative to its forward-facing orientation, specifically at an angle of 30 degrees downward from the horizontal axis in that YZ plane, so that it can scan along the lower-left edge of the shelving for cracks or deformation.

Rotate 180 degrees, then fly the drone in the top-right direction at an angle of 30 degrees from the horizontal axis, in the YZ plane of the drone's body frame for a distance of 10 meters.	Inspect the rear side of a cell-tower panel. First, in the drone's body frame, rotate 180 degrees to face the opposite side of the tower. Then, still in the drone's body frame, fly 10 meters in the YZ plane in the top-right direction, maintaining a 30-degree angle from the horizontal axis, to position the drone optimally for capturing detailed images of the rear cabling and connectors.
Rotate 180 degrees, then fly the drone in the top-left direction at an angle of 30 degrees from the horizontal axis, in the YZ plane of the drone's body frame for a distance of 10 meters.	Conduct an indoor inspection along a warehouse aisle. First, in the drone's body frame, rotate 180 degrees to face the opposite direction down the aisle. Also, in the drone's body frame, fly 10 meters in the YZ plane in a top-left direction, maintaining a 30-degree angle from the horizontal axis to approach and examine elevated shelving and overhead fixtures on the upper-left side of the aisle.
Rotate 180 degrees, then fly the drone in the bottom-right direction at an angle of 30 degrees from the horizontal axis, in the YZ plane of the drone's body frame for a distance of 10 meters.	Perform an inspection of the rear-lower corner of a building. First, in the drone's body frame, rotate 180 degrees to face the opposite direction along the original heading. Then, still in the drone's body frame within its YZ plane, fly 10 meters in the bottom-right direction at an angle of 30 degrees from the horizontal axis to reach and survey the lower-right rear junction of the wall and support beam.
Rotate 180 degrees, then fly the drone in the bottom-left direction at an angle of 30 degrees from the horizontal axis, in the YZ plane of the drone's body frame for a distance of 10 meters.	Perform an internal pipe inspection where the drone needs to reorient and then follow a specific diagonal path. First, in the drone's body frame, rotate 180 degrees to face the opposite direction along the pipe. Then, still in the drone's body frame, fly in the bottom-left direction within the YZ plane at an angle of 30 degrees from the horizontal axis for a distance of 10 meters to inspect a lower-side junction of the pipe.
Turn 60 degrees clockwise, then fly 10 meters forward in the drone's body frame.	Inspect a communication antenna on a building rooftop. First, in the world frame, rotate the drone 60 degrees clockwise to align its front camera with the antenna array, then, in the drone's body frame, fly 10 meters forward to approach the structure for a detailed visual inspection.
Turn 60 degrees clockwise, then fly 10 meters backward in the drone's body frame.	Inspect a billboard along a roadside. In the world frame, yaw the drone 60 degrees clockwise to align its camera with the billboard surface, then, in the drone's body frame, fly 10 meters backward to increase distance for a wider inspection shot while keeping the same orientation.
Turn 60 degrees clockwise, then fly 10 meters right in the drone's body frame.	Perform a perimeter scan around a farm corner. In the world frame, yaw the drone 60 degrees clockwise to align its camera with the side fence, then in the drone's body frame fly 10 meters to the right to fly along the fence.
Turn 60 degrees clockwise, then fly 10 meters left in the drone's body frame.	Conduct an indoor aisle inspection in abandoned storage. First, in the world frame, rotate the drone 60 degrees clockwise to align its forward direction with the target aisle. Then, in the drone's body frame, translate 10 meters to the left to move laterally along the side of a storage rack while keeping its camera oriented down the aisle.

Turn 60 degrees counterclockwise, then fly 10 meters forward in the drone's body frame.	I need to inspect a building corner for damage. First, in the world frame, rotate the drone 60 degrees counterclockwise to align its camera with the next inspection segment. Then, in the drone's body frame, fly 10 meters forward along its current heading so I can see the corner damage.
Turn 60 degrees counterclockwise, then fly 10 meters backward in the drone's body frame.	I need to inspect a building for heat leaks. First, in the world frame, rotate the drone 60 degrees counterclockwise to align its thermal camera with the target wall section. Then, in the drone's body frame, fly 10 meters backward to increase the inspection distance while keeping the camera pointed at the same area for a wider thermal overview.
Turn 60 degrees counterclockwise, then fly 10 meters right in the drone's body frame.	You are a drone flying inside a warehouse. In the world frame, first yaw the drone 60 degrees counterclockwise to align its forward-facing camera toward the new shelf row, then in the drone's body frame, translate 10 meters to the right to move laterally along the shelf for a continuous side-view inspection of stored items.
Turn 60 degrees counterclockwise, then fly 10 meters left in the drone's body frame.	I need to see the status of my roof panel. First, in the world frame, yaw the drone 60 degrees counterclockwise to face along the panel row, then in the drone's body frame, fly 10 meters to the left to position the drone over the adjacent panel for next panel imaging.

Table 5: Application Scenarios in Basic Task Dataset

Task	Scenario
Take off and fly up 5 meters. You should fly in a square pattern with 5-meter sides by moving forward, right, backward, and left along the world axis.	The drone is going to carry out an inspection flight above an empty parking lot. First, take off and ascend 5 meters in the world frame to reach a safe inspection height. Then, in the world frame, fly in a square pattern with 5-meter sides: move 5 meters forward to scan the area ahead, 5 meters to the right to check the neighboring section, 5 meters backward to pass over the starting area again, and finally 5 meters to the left to return to the original position at the same altitude.
Take off and fly up 5 meters. You should fly in a square pattern with 5-meter sides by moving forward in the drone's body frame and turning right at each corner.	Control the drone to conduct an automated perimeter check around a small farm. First, take off and climb vertically 5 meters in the world frame to reach the inspection altitude. Then, in the drone's body frame, fly forward 5 meters to begin the pass along one side, make a right turn to face the next side, fly forward another 5 meters, and repeat this pattern of turning right and flying forward 5 meters at each corner until completing a square path with 5-meter sides.
Take off and fly up 5 meters. You should fly in a square pattern with 5-meter sides by moving forward, right, backward, and left along the world axis. Make sure the drone is oriented in the flying direction.	Carry out a simple perimeter check around a small equipment zone. First, take off and climb vertically 5 meters in the world frame. Then, keeping the drone at this height and always yawed so its front faces the direction of travel, fly a square path with 5-meter sides in the world frame by moving 5 meters forward, then 5 meters to the right, then 5 meters backward, and finally 5 meters to the left to return to the starting point.
Take off and fly up 5 meters. You should fly in a square pattern with 5-meter sides by moving forward, right, backward, and left along the world axis. Make sure the drone is oriented opposite to the flying direction.	I need the drone to fly over a rooftop area. First, take off and ascend 5 meters in the world frame to reach a safe inspection height. Then, fly a square survey pattern with 5-meter sides in the world frame: move forward 5 meters, then right 5 meters, then backward 5 meters, and then left 5 meters to return to the starting horizontal position. Throughout the entire square flight, keep the drone's heading opposite to its current direction of travel so that its camera is always facing the area already inspected.
Take off and fly up 5 meters, then turn to face west. Next, turn to face east. From your current position, fly in a square with 5-meter sides while continuously facing east. The movement pattern should follow this sequence: north, east, south, and west along the world axis.	Please conduct a short positioning and orientation test near a building. First, take off and ascend 5 meters in the world frame, then yaw to face west to verify the heading, followed by a yaw to face east to set the final inspection direction. From this fixed altitude and always keeping the drone's nose facing east, fly a 5-meter square path in the world frame, moving north 5 meters, then east 5 meters, then south 5 meters, and finally west 5 meters to return to the starting horizontal position.

Take off and fly up 5 meters. You will examine a square area. You should fly in a square pattern with 5-meter sides by moving forward, right, backward, and left. To examine the square area, the drone should orientate perpendicular to the moving direction on each side of the square.	Perform a low-altitude inspection of a construction site. First, take off and ascend vertically 5 meters in the world frame to reach the inspection height. Then, survey a square 5-meter-by-5-meter area by flying in a square path: move 5 meters forward, then 5 meters right, then 5 meters backward, and finally 5 meters left, all in the same world frame. While flying along each side of the square, keep the drone's body orientation perpendicular to its current direction of motion so its camera faces sideways relative to the flight path during the entire inspection.
Take off and fly up 5 meters. You will examine a square area. You should fly in a square pattern with 5-meter sides by moving forward, right, backward, and left. To examine the square area, the drone should orientate perpendicular to the moving direction on each side of the square. Next, ascend another 5 meters and fly the square pattern in reverse order to examine the same area.	The drone carries out a two-level aerial inspection of warehouse shelves. First, take off and climb 5 meters in the world frame. At this height, survey a 5-by-5-meter square by flying in a square pattern: move 5 meters forward, then 5 meters right, then 5 meters backward, then 5 meters left, keeping the drone's heading perpendicular to its direction of motion for each leg of the square. After completing this loop, ascend another 5 meters in the world frame and repeat the same square pattern in the reverse order to inspect the same area from a higher altitude.
Take off and fly up 5 meters. Fly in a square with 5-meter sides. The movement pattern should follow this sequence: forward, right, backward, and left along the world axis. Next, fly a second square that is symmetric with respect to the X-axis in the XY plane. To examine the two square areas, the drone should orientate perpendicular to the moving direction on each side of the squares.	Conduct an aerial calibration around a building rooftop. First, take off and ascend 5 meters in the world frame. At that altitude, survey a square inspection area with 5-meter sides by flying straight forward 5 meters, then right 5 meters, then backward 5 meters, and then left 5 meters, all defined in the world frame. Next, inspect a second square area that is the mirror image of the first one across the world X-axis in the XY plane, following the same square path. Throughout both square inspections, keep the drone's heading perpendicular to its current flight direction along each side, so the camera consistently views sideways relative to the motion.
Take off and fly up 5 meters. Fly a figure of 8 on a flat, horizontal plane with each side of 5 meters. The left square is on your left-rear side, and the right square is on your right-front side. You should begin with the left square by flying left. The right square should start from moving forward. Make sure the drone is oriented in the flying direction for all squares.	The drone is a trainer; it is going to fly a training flight for inspecting two adjacent rooftop sections. First, take off and climb 5 meters in the world frame to reach a safe inspection height. Then, at this constant altitude, trace a flat "figure-8" path made of two 5-meter squares on the horizontal plane: the first square is located to your left rear side and is flown by moving left first, and the second square is located to your right front side and is flown by moving forward first. Throughout the maneuver, keep the drone's nose pointed in the direction of motion along each segment of both squares.

Take off and fly up 5 meters. Fly a figure of 8 on a flat, horizontal plane with each side of 5 meters. The left square is on your left-rear side, and the right square is on your right-front side. You should begin with the left square by flying left. The right square should start from moving north. Additionally, for the left square, the drone is oriented in the flying direction; for the right square, the drone should examine the area while flying the right square pattern. A good strategy for examining is the drone orientate perpendicular to the moving direction on each side of the right square.	Conduct an aerial inspection around two adjacent houses. First, take off and ascend vertically 5 meters in the world frame to reach the inspection height. Then, at this constant altitude, fly a flat figure-of-eight pattern made of two adjacent 5-meter-by-5-meter squares in the horizontal plane. The first (left) square is positioned to your left-rear side, and you start this square by moving left while keeping the drone's nose aligned with the direction of motion along each side. After completing the left square, continue directly into the second (right) square, which is positioned to your right-front side and starts with motion toward the north. While flying the right square, maintain the same 5-meter side lengths and altitude, but use the drone to examine the surrounding area by keeping its head oriented perpendicular to the direction of travel on each side, as if scanning the environment sideways while following the square path.
Take off and climb up 5 m. Fly a 5 m square by moving forward, left, backward, and right. Then fly a 5 m square by moving forward, right, backward, and left.	Perform a basic inspection pattern around a small equipment zone. First, take off and climb straight up 5 meters in the world frame to reach a safe inspection height. Then fly a 5-meter square path in the world frame by moving 5 meters forward, 5 meters left, 5 meters backward, and 5 meters right, returning to the starting point above the equipment. After that, repeat a 5-meter square scan in the world frame but this time move 5 meters forward, 5 meters right, 5 meters backward, and 5 meters left, again returning to the same hovering point.
Take off and climb up 5 m. Fly a 5 m square (forward/right/back/left). Then shift 5 m south, and fly the same square pattern again.	Inspect a rooftop section of a warehouse. In the world frame, take off and climb up 5 meters to reach inspection altitude, then fly a 5-meter square path by moving 5 meters forward, 5 meters right, 5 meters backward, and 5 meters left to scan the first area. After finishing this square, shift 5 meters south in the world frame to align over the next section of the roof, and repeat the same 5-meter square path with the same forward, right, back, and left movements to complete the second area scan.
Take off and climb up 5 m. Fly a 5 m square (forward/right/back/left). And fly the same square pattern again. Make sure the drone is oriented to the flying direction in each square.	The drone is going to perform a simple flight check in an open test field. First, take off and climb up 5 meters in the world frame to reach a safe inspection height. Then have the drone fly a square path with 5-meter sides in the world frame by moving forward, then right, then back, then left, keeping its heading aligned with the current flying direction along each side of the square. After completing this square once, repeat the same 5-meter square pattern in the same way, again making sure the drone's orientation always matches the direction of travel on each leg.

Take off and fly up 5 meters. You should fly in a square pattern with 5-meter sides by moving forward, right, backward, and left along the world axis. Then fly the same square pattern again. Make sure the drone is oriented opposite to the flying direction.	Inspect a rooftop ventilation area with repeated coverage at a fixed height. In the world frame, take off and ascend 5 meters to reach the inspection altitude. At this height, fly a square patrol route with 5-meter sides by moving forward, then right, then backward, then left in the world frame, keeping the drone's heading always opposite to its current direction of motion during this square. After completing the first square, repeat the same square pattern a second time at the same altitude, again ensuring the drone stays oriented opposite to its flying direction for each leg of the path.
Take off and fly up 5 meters, then turn to face south. Next, turn to face east. From your current position, fly in a square with 5-meter sides while continuously facing east. The movement pattern should follow this sequence: north, east, south, and west in the world axis.	We just repaired a drone, and now conduct an outdoor positioning and heading check for a drone. First, take off and climb 5 meters in the world frame to reach a safe testing height, then rotate in place to face south, and then rotate again in place to face east to confirm heading control. From this fixed starting point, fly a 5-meter-side square path in the world frame while always keeping the drone's nose pointing east, following straight segments in this order: move north 5 meters, then east 5 meters, then south 5 meters, and finally west 5 meters to return to the starting position.
Take off and climb up 5 m. Fly a 5 m square (forward/right/back/left). Then shift 5 m north, and fly the same square pattern again. The drone will examine the second square area, where the drone should orientate perpendicular to the moving direction on each side of the second square.	You are going to survey in two neighboring grid cells. First, take off and climb 5 meters in the world frame. From there, fly a 5-meter square path by moving 5 meters forward, then 5 meters right, 5 meters back, and 5 meters left in the world frame to complete the first area scan. Next, shift 5 meters north in the world frame to reach the second grid cell, and repeat the same 5-meter square pattern (forward, right, back, left). During this second square, keep the drone's heading perpendicular to its motion along each side so that it is rotated 90 degrees relative to the direction of travel while it examines this second area.
Take off and fly up 5 meters. You will examine a square area. You should fly in a square pattern with 5-meter sides by moving forward, left, backward, and right. To examine the square area, the drone should orientate perpendicular to the moving direction on each side of the square.	Carry out a short aerial inspection over a small field. First, take off and climb vertically 5 meters in the world frame to reach a safe inspection height. Then, at this fixed altitude, survey a square area with 5-meter sides by flying in a square path: move 5 meters forward, then 5 meters left, then 5 meters backward, and finally 5 meters right in the world frame to return to the starting point. While flying along each side, keep the drone's heading perpendicular to the current flight direction so the camera looks sideways relative to the movement for better coverage of the area.

Take off and fly up 5 meters. Fly in a square with 5-meter sides. The movement pattern should follow this sequence: forward, right, backward, and left along the world axis. Then descend 5 meters. Next, fly a second square that is symmetric with respect to the north-south(X) axis. You will examine the two square areas, and the drone should orientate perpendicular to the moving direction on each side of the squares.	Conduct an automated inspection of two adjacent rooftop zones. First, in the world frame, take off and ascend 5 meters to the inspection height. Then survey the first zone by flying a square path with 5-meter sides, moving in sequence: forward, right, backward, and left, all defined in world axes. After completing this loop, descend 5 meters in the world frame to return to the starting altitude. Next, perform a second square survey at the same scale, with the square path mirrored about the north-south (X) axis in the world frame, so it is symmetric to the first one. Throughout both square paths, keep the drone's heading oriented perpendicular to its direction of motion along each side to simulate a side-looking inspection payload.
Take off and fly up 5 meters. Fly a figure of 8 on a flat, horizontal plane with each side of 5 meters. The first square of 8 is on your left-front side, and the right square is on your right-rear side. You should begin with the first square by flying left. The right square should start from moving south. Make sure the drone is oriented in the flying direction for all squares.	You are responsible for conducting an inspection flight around a small neighborhood. First, take off and climb vertically 5 meters in the world frame to reach a safe inspection height. Then, at that constant altitude, trace a horizontal figure 8 pattern with two adjoining 5-meter-wide squares in the world frame: fly the first 5-meter square to your left-front side, starting by moving left, keeping the drone's nose always pointed along the direction of motion; then continue directly into the second 5-meter square on your right-rear side, starting that square by moving south in the world frame, again keeping the drone oriented along the direction of travel for every segment of both squares.
Take off and fly up 5 meters. Fly a figure of 8 on a flat, horizontal plane with each side of 5 meters. The first square of 8 is on your left-front side, and the right square is on your right-rear side. You should begin with the first square by flying left. The right square should start from moving south. Additionally, for the left square, the drone is oriented in the flying direction; for the right square, the drone should examine the area while flying the right square pattern. A good strategy for examining the inward square area is to let the drone orientate perpendicular to the moving direction on each side of the right square.	You should control the drone to do a low-altitude inspection of a small construction site. First, take off and ascend vertically 5 meters in the world frame. Then, at that height, survey two adjacent 5 m*5 m square zones on a flat horizontal plane in a figure 8 path: the first square lies to your left front and the second to your right rear. Start by flying the left-front square, beginning with a 5-meter segment to the left and completing the full square while keeping the drone's nose pointed along the direction of travel on each side. After finishing that square, continue into the right-rear square, starting its path with a segment heading south and flying the same 5-meter-per-side square. During this right-rear square, inspect the interior of the square by keeping the drone's camera pointed toward the inside area: on each side, orient the drone so its body is perpendicular to the direction of motion while it follows the right-rear square pattern.

Table 6: Application Scenarios in Advanced Task Dataset