
FlySearch:

Exploring how vision-language models explore

Adam Pardy^{1,2,3} Dominik Matuszek^{1,2} Mateusz Przebieracz² Marek Cygan^{4,5}
Bartosz Zieliński² Maciej Wolczyk¹

Abstract

The real world is messy and unstructured. Uncovering critical information often requires active, goal-driven exploration. It remains to be seen whether Vision-Language Models (VLMs), which recently emerged as a popular zero-shot tool in many difficult tasks, can operate effectively in such conditions. In this paper, we answer this question by introducing FlySearch, a 3D, outdoor, photorealistic environment for searching and navigating to objects in complex scenes. We define three sets of scenarios with varying difficulty and observe that state-of-the-art VLMs cannot reliably solve even the simplest exploration tasks, with the gap to human performance increasing as the tasks get harder. We identify a set of central causes, ranging from vision hallucination, through context misunderstanding, to task planning failures, and we show that some of them can be addressed by finetuning. We publicly release the benchmark, scenarios, and the underlying codebase.

1 Introduction

Vision-Language Models (VLMs) have rapidly emerged as state-of-the-art performers in tasks ranging from image captioning [32, 69] to robotics [7, 44]. However, real-world decision-making requires curiosity, adaptability, and a goal-oriented mindset. Nevertheless, the ability of VLMs to operate in realistic, open-ended environments remains largely untested. In this paper, we propose a benchmark to understand and enhance the exploratory capabilities of VLMs. We draw inspiration from the field of Object Navigation (ObjectNav), which focuses on creating embodied agents capable of finding a specific object in a simulated environment and navigating to it. The object may not be visible from the agent’s initial perspective, meaning the agent must perform a careful search to locate it.

There are several significant differences between our benchmark FlySearch and existing counterparts. First, we measure the exploration capabilities of VLMs themselves rather than analyzing more complex systems built on them. Understanding these capabilities is important because gathering information is a crucial aspect of the emerging agentic systems [45]. Second, while most ObjectNav benchmarks take place indoors, we focus on finding objects in a large outdoor space using an Unmanned Aerial Vehicle (UAV). This allows us to examine how VLMs explore large, diverse, and unstructured areas and how they control the altitude to change the search granularity. Finally, we focus on a zero-shot open-ended search setting, i.e., a tested method should not make any prior assumptions about the testing environment, the categories of objects, or the *search process itself*. For example, methods that use detectors of specific objects are out of the scope of our research.

¹IDEAS NCBR; ²Jagiellonian University, Faculty of Mathematics and Computer Science; ³Jagiellonian University, Doctoral School of Exact and Natural Sciences; ⁴University of Warsaw; ⁵Nomagic.
Correspondence to: Adam Pardy adam.pardyl@doctoral.uj.edu.pl

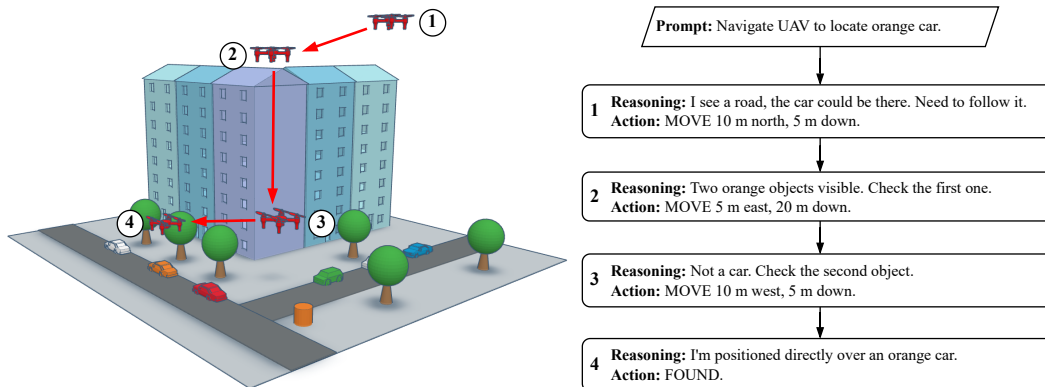


Figure 1: **FlySearch** is a benchmark that evaluates exploration skills using vision-language reasoning. To complete each assessment scenario, a model must locate an object specified in natural language. The agent controls an Unmanned Aerial Vehicle (UAV) by observing images obtained from successive locations of the UAV and providing text commands describing the next move.

FlySearch¹ aims to evaluate the extent to which VLMs can autonomously reason and explore, as well as characterize the limitations of their current capabilities. In FlySearch, the agent (tested method) controls a UAV flying over urban or natural environments to search for objects of interest, such as specific cars, fires, lost people waiting to be rescued, or piles of garbage (see Figure 1). It is built from scratch using the realistic Unreal Engine 5 [15], widely used for video games, providing dynamic 3D scenes with many assets. Moreover, procedural generation allows the generation of an unlimited number of scenarios with varying environmental characteristics, such as time of day, forest density, and UAV launch altitude. The benchmark runs efficiently in headless mode on both modern consumer-grade GPUs (NVIDIA RTX) and standard deep learning clusters (A100/H100), lowering the entry barrier for researchers and practitioners.

FlySearch consists of 3 standardized scenario sets with varying levels of difficulty. **FS-1** tests basic perception and navigation skills, **FS-Anomaly-1** additionally probes the capability of agents to understand the context of the environment, while **FS-2** requires executing a consistent exploration strategy involving a large number of steps to find the object of interest. We evaluate 3 closed-weight and 6 open-weight models on these scenarios and compare the results to scores obtained by humans. We find that VLMs strongly underperform humans on both FS-1 and FS-2. Additionally, while the performance drop between FS-1 and FS-2 is relatively low for humans ($\approx 9\%$), it is massive for VLMs ($\approx 90\%$). Therefore, despite possessing basic navigation and visual comprehension skills, current VLM models fail to form and execute proper exploration strategies, even after GRPO-based fine-tuning.

Our contributions can be summarized as follows:

- We release two high-fidelity outdoor environments built with Unreal Engine 5, enabling realistic and scalable evaluation of embodied agents in complex, unstructured settings.
- We define a suite of object-based exploration challenges designed to isolate and measure the exploration capabilities of VLMs and humans in open-world scenarios.
- We benchmark several popular VLMs in a zero-shot setting and identify consistent failure modes across vision, grounding, and reasoning. Our analysis reveals that these limitations persist even with fine-tuning, suggesting fundamental gaps in current VLM architectures.

2 Related work

ObjectNav. Our environment is focused on Object-Goal Navigation (ObjectNav or ObjectGoal) task, where the goal is to navigate to a specific object type in a given environment [3, 6]. Using this environment, we instantiate challenges that tie into Language-Driven Zero-Shot ObjectNav [12, 17, 39] tasks, as we expect tested methods to be able to perform search for an arbitrary text-based

¹<https://github.com/gmum/FlySearch>

Table 1: **Other benchmarks:** We compare FlySearch with other benchmarks focused on VLM evaluation and ObjectNav challenges. We denote cases of partial criterion satisfaction with ✓.

Environment	Photorealistic	Outdoor	Exploration-focused	3D	Focus on VLM eval
Habitat Nav. Challenge [66]	✓	✗	✓	✓	✗
RoboTHOR [11]	✗	✗	✓	✓	✗
AgentBench [35]	✗	✗	✗	✗	✓
LMAct [51]	✗	✗	✗	✗	✓
SmartPlay [63]	✗	✓	✗	✗	✓
BALROG [45]	✗	✓	✗	✗	✓
VisualAgentBench [36]	✓	✓	✗	✓	✓
OpenEQA [40]	✓	✗	✗	✓	✓
FlySearch (ours)	✓	✓	✓	✓	✓

object description. Currently, Habitat-Sim [52, 66], AI2-THOR [11, 25], Gibson Env [64] are the most commonly used environments for the ObjectNav task [57]. All of these environments are indoor-focused, while in FlySearch the agent is embodied within a UAV in an outdoor scenario.

VLN. There also exists a related broad field of vision-and-language navigation (VLN), mainly concerned with embodied agents following natural language instructions that specify a route from point A to point B [70]. We note the existence of outdoor UAV-centric environments for VLN tasks, such as AerialVLN [34], TRAVEL [61] or CityNav [28]. AerialVLN is an example of a VLN task with high-level UAV control and a simulated environment, which follows the above definition. TRAVEL aims to make drone-based VLN more low-level by modifying the agent’s action space for direct UAV control. CityNav introduces a VLN environment based on scans of real-life cities from SensatUrban [21] (instead of simulating them). These papers focus on instruction following in navigation (i.e., VLN), while our benchmark puts emphasis on finding objects without further instructions (i.e., ObjectNav).

AirSim and OUTDOOR. Microsoft AirSim [54] is an Unreal Engine 4 plugin that can be used to create UAV simulators, which can also be used for ObjectNav tasks in an outdoor setting. However, to the best of our knowledge, only OUTDOOR [65] considers such application of this software, as ObjectNav is generally dominated by indoor environments. However, even though OUTDOOR considers using UAVs, the action space is still 2-dimensional (horizontal movement only), while FlySearch has a 3-dimensional exploration space (the UAV can also move up and down). This emphasises dynamic altitude control to manage uncertainty and avoid redundant exploration, as the level of visual detail and the size of the visible area varies depending on the altitude. As such, success in FlySearch requires an efficient, emergent search strategy, where the agent must reason about where the object is likely to be, when looking from high altitude. Overall, this is a much more complex problem, requiring pre-existing knowledge of the real world to understand the contextual cues to exploit a wide field of view at high altitudes and fly low only in promising areas.

AVE. Additionally, the problem of exploration is considered from a different perspective in Active Visual Exploration (AVE) tasks [46, 47, 53], where a stationary agent is provided with partial observations derived from a large static image, simulating a limited field of view.

VLM benchmarks. Vision question-answering (VQA) datasets that contain pairs of images and textual questions are among the most popular approaches to assess the capabilities and limitations of VLMs. There are many examples of VQA-oriented benchmarks, such as [19, 23, 37, 41, 56, 68]. However, to fully evaluate VLM’s performance, benchmarks that can measure their performance in practical tasks are needed [22, 33]. Recently, benchmarks that require interaction with an environment have gained popularity in the context of evaluating the capabilities of different VLMs. For example, BALROG [45], VisualAgentBench [36] and LMAct [51] evaluate VLM-based agents on several different environments with visual and textual representations. MineDojo [16] can be used to test different VLM-based agents in an interactive, 3D environment based on the popular *Minecraft* game. Other examples of benchmarks include SmartPlay [63] and AgentBench [35], although these are limited only to the textual modality (with images being "encoded" into natural language). There are also examples of embodied question-answering benchmarks for VLMs, such as OpenEQA [40]. We differ from other benchmarks by specifically focusing on gauging VLMs in the ObjectNav, which allows us to evaluate exploration capabilities of VLMs with a high degree of independence from their other capabilities.

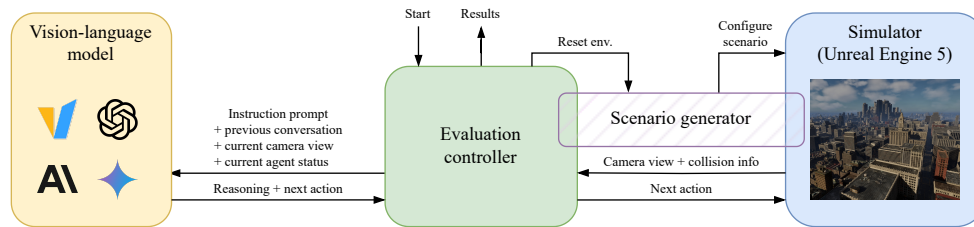


Figure 2: **Evaluation pipeline:** FlySearch consists of three parts besides the vision-language model. The simulator renders near-photorealistic views of a large open-world map and handles basic physics such as collisions. The evaluation controller handles the communication between the evaluated vision-language model and the simulator and performs the evaluation. The scenario generator (functionally integrated with the controller and simulator) procedurally generates new evaluation scenarios.

VLMs in ObjectNav. Recently, we have seen a surge in applications of VLM foundation models in ObjectNav [9, 10, 12, 26, 67] and related tasks [38, 62, 72]. However, we note that these papers use VLMs as tools to create more intricate methods and they do not aim to measure the exploration capabilities of VLMs themselves, which are still largely unknown [57]. In this paper, we attempt to gauge VLMs’ performance by treating FlySearch as a VLM benchmark meant to compare different models and to understand their shortcomings.

3 FlySearch

The goal of FlySearch is to evaluate the visual-spatial reasoning and information-gathering abilities of vision-language models. To achieve this, the model is given control of a simulated multirotor unmanned aerial vehicle equipped with a camera and tasked with finding an object described in natural language, see Figure 1.

3.1 Evaluation task

Environment. The evaluation environment is a square outdoor area consisting of a fragment of a photorealistic procedurally generated map. The agent starts in the center of the area, at a randomly chosen altitude. Somewhere within the fragment is a target object. The agent must locate it within a limited number of steps and report success.

Starting prompt. The model is given a detailed prompt describing its task, including a brief textual description of the object to be located, e.g. *a red pickup truck*. The prompt also describes the communication format, including how to format the responses. We allow the model to preface each of its responses with a description of its reasoning, effectively allowing a chain of reasoning. We provide the prompt template in Appendix G.

Observation. At each exploration step, we provide the agent with a 500×500 pixel RGB image from the simulated UAV camera. To simplify the task, the camera is always facing the ground. The camera image is overlaid with a grid of coordinates to help the model understand movement directions and distances [29]. Additionally, we provide the agent with its height above the ground. In case of FS-2, we also provide the agent with the image specifying how searched object should look like from above; that is done to focus FS-2 more on search. All data except the images is provided in XML format.

Action. We assume that the simulated UAV is equipped with an autopilot system. Therefore, the agent does not need to provide any low-level control signals. Instead, it can focus on exploration by providing simple text commands with the tag `<action>(X, Y, Z)</action>`, where each of the coordinates represents a relative position change in meters in the corresponding direction. We introduce a collision avoidance system that stops the movement if an obstacle is detected within a 0.5 meter radius of the camera position, or if it tries to move out of the fly zone. When the agent decides that it has completed the task, it should respond with the FOUND text and end the exploration. If at any point the model response cannot be parsed, the episode is terminated.



Figure 3: **Environments:** Our benchmark consists of two types of evaluation environments, forest and city. For each environment, we can generate an infinite number of procedurally generated test scenarios. The top row shows a preview of the environment, exhibiting the visual fidelity of the simulation. Below are top-down views of objects, matching the perspective of the agent.

Metrics. An episode is considered successfully completed (i.e., the object is found) if the object’s center point is within the agent’s camera field of view, the agent’s position is at most 10 m above the highest point belonging to the object, and the agent returns the FOUND action. This metric is related to the one defined in [6], although we use the altitude difference and field of view instead of the Euclidean distance, as it is easier to estimate for human testers and VLMs – one only needs to make sure that the altitude is not higher than 10 m and the object is visible. We provide details on the implementation of the success criterion in Appendix F.

3.2 Evaluation pipeline

The FlySearch system consists of two main components: a simulator based on Unreal Engine 5 and a controller module. The controller is responsible for scenario generation, communication between benchmarked VLMs and the simulator, and result aggregation, as seen in Figure 2.

Simulator. To ensure realistic image input for the evaluated tasks, we chose Unreal Engine 5 as the simulation engine. It provides near-photorealistic graphics through real-time ray tracing and dynamic global illumination, while also supporting large, detailed open worlds. The platform is compatible with all major operating systems, and has an open-source codebase, enabling customization for machine learning applications. Additionally, its procedural content generation facilitates environment randomization by integrating parts of the scenario generator directly into the server, allowing us to place tens of thousands of object meshes in seconds. We also leveraged Unreal’s extensive online marketplace of free assets to create evaluation scenarios. As such, these assets can be easily used to further extend the benchmark with new environments, scenarios, or objects of interest.

The simulator can be run using any modern consumer-grade graphics card as well as deep learning dedicated solutions (provided Vulkan is supported), and the engine runs in headless/offscreen mode (without a monitor). As such, we found it to perform well on standard computing clusters. Communication between the Unreal Engine simulator and the evaluation controller is handled via standard TCP/IP networking. The simulator-side implementation is provided as a native Unreal Engine plugin. We build upon the UnrealCV project [50], extending its functionality to allow full use of the aforementioned Unreal Engine 5 features.

Evaluation controller. The final component of FlySearch is the evaluation controller, implemented in Python. This module oversees the entire lifecycle of the benchmarking process, including setting up scenarios and calculating performance metrics. It also handles communication between the simulator

and the evaluated vision-language model (VLM). Details of the prompts and message templates used in FlySearch are provided in Appendix G. FlySearch supports multiple VLMs, as described in Section 4.1. Additional models can be integrated easily by adding simple adapter code to the controller module or using the open-source vLLM inference server [27].

3.3 Evaluation environments

The benchmark consists of two distinct evaluation environment types, a forest and a city, see Figure 3. Each of them has its own set of target objects to find. **Forest environment** is based on the “*Electric Dreams Environment*” Unreal Engine product demo [14]. It consists of sparse forest scenery, including randomly placed rock formations. The map is procedurally generated entirely at runtime by the scenario generator. Moreover, all vegetation on the map is subject to wind changes. **City environment** is a large, modern, American-style city, based on the “*City sample*” Unreal Engine demo [13]. The city layout is a large, semi-procedurally generated map of roughly 4×4 km. New maps can be generated with the tools provided at build time. Furthermore, random distractor assets (parked cars and walking pedestrians) are spawned by the scenario generator at runtime.

4 Evaluation

FlySearch can be used to generate scenarios suited to the user’s needs, e.g., finding cars at night in the city or locating stranded people in a very dense forest. However, to facilitate further research and enable fair comparisons, we propose three standardized challenges, i.e., sets of reproducible scenes that can be used to evaluate agents under the same conditions. Although standardized, it’s important to highlight that these challenges are not static datasets, as the environment will dynamically react to the actions of the agent.

FS-1 contains 400 scenes of finding particular objects in the city and forest environments. The agent is explicitly instructed to find an object with a unique description in at most 10 actions. Certain distracting objects may appear, e.g., if we ask the model to find a yellow sports car, there might be cars of other colors or types in the scene. The goal of this challenge is to measure the general search capabilities of the model across a wide array of objects that might be encountered in various UAV applications, e.g., search-and-rescue missions and fire detection. The search area is limited to $400 \times 400 \times 120$ m, and the starting altitude to between 30 and 100 m, and the agent may not fly outside its field of view. We ensure, that the target object is within the field of view of the starting position and its center is not obscured by hard obstacles (it may not be distinguishable due to distance). The agent has to find one of the following:

- in the city: road construction works, crowd, large trash pile, fire, vehicle (variable type).
- in the forest: campsite, trash pile, person, forest fire, building.

In both scenarios, all target objects are positioned on the ground, in semantically correct locations, e.g., a car will be placed in a parking spot, not on the roof. All objects have multiple variations, randomly selected on scenario generation.

FS-Anomaly-1 is a set of 200 scenes in the city and the forest, where the agent is instructed to find an object that seems out of place, e.g., a giraffe in the city or a UFO in the forest. The goal of this challenge is to measure both the search capabilities of the models as well as their knowledge about what is and is not expected in certain environments. All other settings follow those of the previously mentioned FS-1. Anomalies are placed on the ground level. The anomalous objects to be located are:

- in the forest: UFO (flying saucer), small airplane, helicopter, large dinosaur, airliner,
- in the city: UFO (flying saucer), small airplane, helicopter, medium dinosaur, tank, giraffe.

FS-2 consists of 200 additional harder scenarios in the city environment. Base setting is the same as in FS-1; however, starting altitude range is raised to between 100 and 125 m and the search area is limited to (-starting altitude, +starting altitude) range in X and Y axis. Moreover, dynamic scene lightning is enabled, simulating different times of day and the maximum allowed altitude is 300 m. Most importantly, we allow the object to be obscured by obstacles (but still visible from the sky) and we allow the model to move beyond its field of view at each step to check models navigation capabilities.

Table 2: **Benchmark results:** Success rates ($\pm$ standard errors) of the evaluated models for FS-1 and FS-2 challenges. We observe that, overall, Gemini 2.0 Flash outperforms all evaluated models. Notably, small open-source models largely fail to solve the test scenarios, while larger models such as the Pixtral 124B achieve better performance. Note that the last row shows the model fine-tuned on the Forest environment (but not the specific FS-1 scenarios), which might lead to overfitting. To signal this, we mark the corresponding result gray.

Model	FS-1			FS-2
	Overall (%)	Forest (%)	City (%)	Overall (%)
Human (untrained)	–	–	66.7 ± 4.5	60.8 ± 6.9
GPT-4o	39.5 ± 2.4	45.5 ± 3.5	33.5 ± 3.3	3.5 ± 0.9
Claude 3.5 Sonnet	41.2 ± 2.5	52.0 ± 3.5	30.5 ± 3.3	6.5 ± 1.2
Gemini 2.0 flash	42.0 ± 2.5	42.5 ± 3.5	41.5 ± 3.5	6.0 ± 1.1
Phi 3.5 vision	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	–
InternVL-2.5 8B MPO	2.0 ± 0.7	2.5 ± 1.1	1.5 ± 0.9	–
Llava-Interleave-7b	0.8 ± 0.4	0.0 ± 0.0	1.5 ± 0.9	–
Qwen2.5-VL 7B	3.8 ± 1.0	6.0 ± 1.7	1.5 ± 0.9	0.0 ± 0.0
Qwen2-VL 72B	17.2 ± 1.9	16.5 ± 2.6	18.0 ± 2.7	–
Llava-Onevision 72b	9.5 ± 1.5	12.5 ± 2.3	6.5 ± 1.7	–
Pixtral-Large	29.8 ± 2.3	38.0 ± 3.4	21.5 ± 2.9	3.0 ± 0.8
Qwen2.5-VL 7B, GRPO on Forest	–	57.0 ± 3.5	27.0 ± 3.1	0.0 ± 0.0

4.1 Baselines

We evaluate a range of popular models. We select three proprietary models: OpenAI GPT-4o (2024-08-06 release) [2, 24], Anthropic Claude 3.5 Sonnet [4], and Google Gemini 2.0 flash (experimental) [18, 58]. Furthermore, we select four small open-weight models, with below 11B parameters: Phi-3.5-vision [1], InternVL2.5-8B-MPO [60], Llava-Interleave-Qwen-7B-dpo-hf [31], and Qwen2.5-VL 7B [5]. Finally, we select three more open-weight models with more than 11B parameters: Qwen2-VL-72B-Instruct [59], Llava-Onevision-Qwen2-72B-ov-hf [30], and Pixtral-Large-Instruct-2411 124B [43]. All models were selected based on their ability to process a full evaluation run, keeping all steps in context. During selection, we tested and rejected Llama-3.2 [42]. Although it architecturally supports handling multiple images, the publicly available model fails to form coherent responses when there is more than one image in the context.

Human study. Furthermore, we provide a human baseline for FS-1 City and FS-2 based on a user study of respectively 111 and 51 samples. The study was conducted using an online service, where participants were provided with the benchmark prompt and had to perform the same actions as the VLM. We provide more details about human baselines in Appendix D.

4.2 Results

FS-1. Table 2 contains the main aggregated results of our study. We find that the state-of-the-art VLMs achieve significantly worse results than non-trained humans on FS-1. While humans score 67% on average, the best-performing Gemini 2.0 manages to find the object in 42% of cases, with Claude and GPT-4o closely following. Large open-weight models fall behind significantly, with Pixtral, the best-performing model in this category, losing 10 percentage points on average to the proprietary models. Finally, the small open-weight models do not show any signal at all, as none of them exceeded 4%. We find that their poor performance can be largely attributed to their inability to follow instructions. Figure 4 shows that small models often do not claim that they have found the object even if it is within range at the end of the episode. As such, we exclude the small models from further analysis. We provide additional results, including measuring the impact of the action representation and the grid overlay, in Appendix H.

FS-2. Strikingly, the situation looks much different on FS-2. While on FS-1 humans outperformed best VLMs by 60%, on FS-2 this number is closer to 835%. We attribute this gap to the lack of systematic exploration abilities in VLMs. Since in FS-1 the object should be visible in the initial

Table 3: **FS-Anomaly-1 results:** Success rates ($\pm$ standard errors) of the evaluated models. Full results are in Appendix H.

Model	FS-Anomaly-1
	Overall (%)
GPT-4o	27.0 ± 3.1
Claude 3.5 Sonnet	27.5 ± 3.2
Gemini 2.0 flash	35.5 ± 3.4
Phi 3.5 vision	0.0 ± 0.0
InternVL-2.5 8B MPO	3.5 ± 1.3
Llava-Interleave-7b	0.0 ± 0.0
Qwen2.5-VL 7B	2.8 ± 1.2
Qwen2-VL-72B	7.5 ± 1.9
Llava-Onevision 72b	8.5 ± 2.0
Pixtral-Large	15.0 ± 2.5

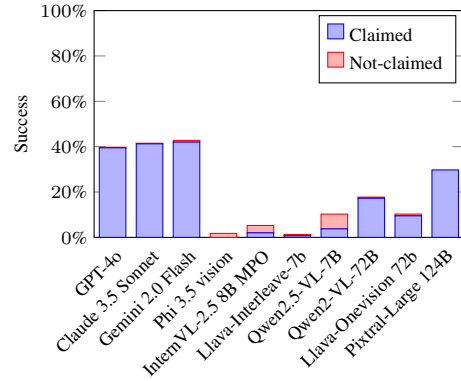


Figure 4: **Not claimed successes in FS-1:** In this figure we compare the number of cases, where the agent located the object, but failed to claim the FOUND action. We observe, that small models often fail to format the text output to report success.

Table 4: Performance of Gemini 2.0 Flash and Pixtral-Large in ablation studies. Surprisingly, increasing the number of steps might lead to worse performance. Additionally, explicitly specifying the object type in FS-Anomaly-1 improves the results as the task becomes easier.

Setting		City		Forest		Overall	
		Gemini	Pixtral	Gemini	Pixtral	Gemini	Pixtral
FS-1	5 steps limit	34.5%	15.5%	41.5%	33.5%	38.0%	24.5%
	10 steps limit (baseline)	41.5%	21.5%	42.5%	38.0%	42.0%	29.8%
	20 steps limit	33.5%	13.5%	45.5%	36.0%	39.5%	24.8%
FS-Anomaly-1	Searching for an anomaly (baseline)	25.0%	4.0%	46.0%	26.0%	35.5%	15.0%
	Searching for explicit object types	34.0%	7.0%	59.0%	34.0%	46.5%	20.5%

frame, it requires mostly object recognition and spatial reasoning abilities. On the other hand, since the object might not be within the field of view in FS-2, it requires the agent to implement a strategy that extensively explores the environment over multiple timesteps. While humans intuitively start following the streets in search of the object in question, VLMs mostly wander aimlessly in random directions. The problem is exacerbated due to difficulties with handling long contexts, see the episode length ablation in a later paragraph.

Fine-tuning results. We find that although some of these shortcomings can be addressed through simple fine-tuning, the problems with systematic exploration are more fundamental. We finetune Qwen2.5-VL-7B [5] using GRPO [55] on Forest environment in an offline mode, using a synthetic set of randomly generated flight trajectories, see details in Appendix E. The resulting model, presented in the last rows in Table 2, vastly improves upon the base model’s performance on FS-1 City scenario, boosting the score by 14 times, from roughly 1.5% to 21.5%. However, the fine-tuning does not impact the results on FS-2 at all, where we still never see any successes.

Qualitative analysis. We perform further qualitative analysis of the larger models, see Figure 6 and Appendix J for example trajectories. By manually analyzing failed exploration trajectories on FS-1, we observe that even the most advanced models struggle with spatial reasoning. For instance, when a model loses sight of an object, it often backtracks its moves or starts hallucinating rather than moving toward the object’s last known location. In case of FS-2, these issues are aggravated by the additional need of carrying out a systematic search. For example, GPT-4o often flies to the ground and hallucinates the existence of a searched object, whilst not performing any reasonable exploration pattern at all. Moreover, all models fail to handle collisions properly, often attempting to redo the same action when it previously resulted in hitting an object. In one of the trajectories, Gemini states *"I am unable to navigate without hitting the building. I guess I will give up."*

Object-specific analysis. In Figure 5 we zoom in on the specific object classes that appear in FS-1. As expected, classes with large objects, such as buildings have a higher success rate than those with

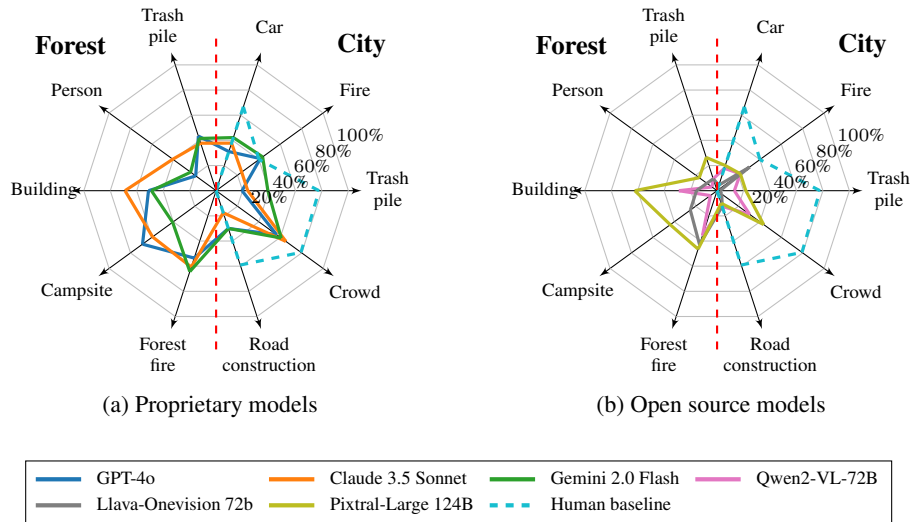


Figure 5: **Success rate per class:** In this figure, we show the model success rate per each target class for both the Forest and the City environments. Classes with large visible objects such as *Building* for Forest or *Crowd* for City are significantly easier for most models to locate. On the other hand, classes that are difficult to distinguish from the background, such as both garbage classes, are only located by more advanced models. Human baseline for City is marked with dashed line.

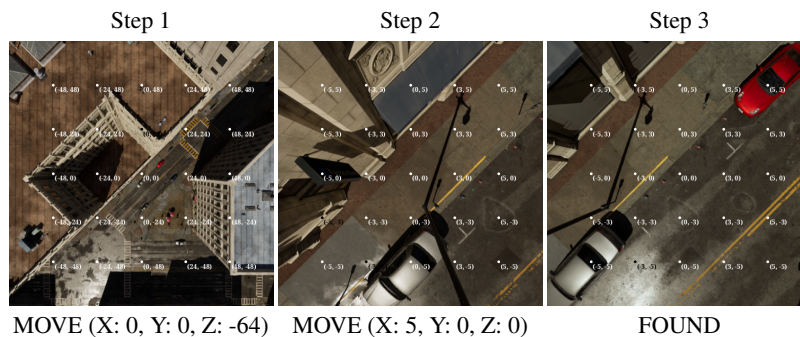


Figure 6: **Example of a successful trajectory in FS-1:** GPT-4o navigates to *red sports car* object by first descending and then moving to the right. The first row shows the model's visual inputs, and the second actions it has taken. Note the presence of the grid overlay on images. Best viewed zoomed in.

smaller figures, such as single persons. However, even finding a building poses a significant challenge since the best-performing Claude 3.5 Sonnet does not manage to find approximately 30% of objects in this class. Interestingly, VLMs tend to be more successful at finding trash piles in the forest rather than in the city, where they are more visible from a distance. Lastly, the road construction site class is one of the hardest classes, even though the name provides a clue as to where the model should look.

FS-Anomaly-1. The results for FS-Anomaly-1, presented in Table 3 are on average significantly lower than in FS-1, suggesting that the models struggle with the additional challenge of figuring out which object is out of place. Indeed, as we confirm on a subset of the models, the performance substantially increases once we explicitly name the anomaly object to be found, see Appendix H. Without explicit instructions, more often than not, VLMs find one of many typical, although visually distinct objects to be out of place, while ignoring obvious anomalies. For example, in one of the city scenarios, all closed-source VLMs wrongly identified a *yellow taxi* to be the anomaly, ignoring a *tank* standing next to it. Moreover, we find that the models sometimes tend to misidentify the anomaly objects as something more expected in a given situation, e.g., one of the models assumed that a giraffe walking around city streets is a dog.

Impact of the number of steps. Finally, we study the impact of changing the length of each episode in FS-1, i.e., the number of actions that can be taken before the trajectory automatically ends in failure. We find that reducing the number of steps from the baseline 10 to 5 reduces the results by 10% (4pp) for Gemini and 17% (5pp) for Pixtral. More interestingly, we also observe performance deterioration as we increase the step limit to 20 – Gemini’s performance falls by 6% (2.5pp) and Pixtral’s by 17% (5pp). Breaking these results by category, we find that increasing the number of steps only leads to significant deterioration in the visually cluttered City environment. We discover that models tend to fail when expected to reason and gather information over longer timeframes, which we also observed in FS-2. See more details in Table 4 and Appendix H.

5 Conclusion

In this paper, we introduce FlySearch, a dynamic benchmark designed for evaluating exploration capabilities of VLMs. Navigating three-dimensional environments and finding objects of interest are everyday real-world tasks that remain underrepresented in VLM benchmarking. To address this gap, FlySearch leverages Unreal Engine 5, a highly realistic video game engine to procedurally generate scenarios of searching for objects in urban and natural environments. Using the three standardized challenges, FS-1, FS-Anomaly-1, and FS-2, we show that VLMs underperform compared to human baseline, especially when it comes to more complex exploration tasks. At the same time, this study has certain limitations that offer interesting directions for future work. In this paper, we purposefully avoid testing more sophisticated ObjectNav methods [9, 26, 67], since our main focus lies in understanding pure VLM capabilities. At the same time, checking their performance in FlySearch could bring interesting insights to the field. Additionally, we use a simple prompting technique, and one could possibly get better results out of VLMs by leveraging few-shot learning [8, 48] or prompt optimization tools [49, 73].

Acknowledgments

This paper has been supported by the Horizon Europe Programme (HORIZONCL4-2022-HUMAN-02) under the project "ELIAS: European Lighthouse of AI for Sustainability", GA no. 101120237. This research was funded by National Science Centre, Poland (grant no. 2023/50/E/ST6/00469 and Sonata Bis grant no 2024/54/E/ST6/00388). The research was supported by a grant from the Faculty of Mathematics and Computer Science under the Strategic Programme Excellence Initiative at Jagiellonian University. We gratefully acknowledge Polish high-performance computing infrastructure PLGrid (HPC Center: ACK Cyfronet AGH) for providing computer facilities and support within computational grant no. PLG/2024/017483. Some experiments were performed on servers purchased with funds from the Priority Research Area (Artificial Intelligence Computing Center Core Facility) under the Strategic Programme Excellence Initiative at Jagiellonian University.

References

- [1] M. Abdin, J. Aneja, H. Awadalla, et al. Phi-3 technical report: A highly capable language model locally on your phone. Technical Report MSR-TR-2024-12, Microsoft, August 2024.
- [2] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Al-tenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report, 2024.
- [3] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018.
- [4] Anthropic. Introducing claude 3.5 sonnet. <https://www.anthropic.com/news/claude-3-5-sonnet>, Jun 2024.
- [5] S. Bai, K. Chen, X. Liu, J. Wang, W. Ge, S. Song, K. Dang, P. Wang, S. Wang, J. Tang, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025.
- [6] D. Batra, A. Gokaslan, A. Kembhavi, O. Maksymets, R. Mottaghi, M. Savva, A. Toshev, and E. Wijmans. Objectnav revisited: On evaluation of embodied agents navigating to objects. *arXiv preprint arXiv:2006.13171*, 2020.

- [7] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. π_0 : A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.
- [8] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.
- [9] W. Cai, S. Huang, G. Cheng, Y. Long, P. Gao, C. Sun, and H. Dong. Bridging zero-shot object navigation and foundation models through pixel-guided navigation skill. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5228–5234, 2024.
- [10] Y. Cai, X. He, M. Wang, H. Guo, W.-Y. Yau, and C. Lv. Cl-cotnav: Closed-loop hierarchical chain-of-thought for zero-shot object-goal navigation with vision-language models. *arXiv preprint arXiv:2504.09000*, 2025.
- [11] M. Deitke, W. Han, A. Herrasti, A. Kembhavi, E. Kolve, R. Mottaghi, J. Salvador, D. Schwenk, E. VanderBilt, M. Wallingford, L. Weihs, M. Yatskar, and A. Farhadi. RoboTHOR: An Open Simulation-to-Real Embodied AI Platform. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3161–3171, Los Alamitos, CA, USA, Jun 2020. IEEE Computer Society.
- [12] V. S. Dorbala, J. F. Mullen, and D. Manocha. Can an embodied agent find your “cat-shaped mug”? Llm-based zero-shot object navigation. *IEEE Robotics and Automation Letters*, 9(5):4083–4090, May 2024.
- [13] Epic Games. City sample. an overview of the features used to create the city sample learning example. <https://dev.epicgames.com/documentation/en-us/unreal-engine/city-sample-project-unreal-engine-demonstration>, Nov 2024.
- [14] Epic Games. Electric dreams environment in unreal engine | unreal engine 5.5 documentation. <https://dev.epicgames.com/documentation/en-us/unreal-engine/electric-dreams-environment-in-unreal-engine>, Nov 2024.
- [15] Epic Games. Unreal Engine 5.5. <https://unrealengine.com>, Nov 2024.
- [16] L. Fan, G. Wang, Y. Jiang, A. Mandelkar, Y. Yang, H. Zhu, A. Tang, D.-A. Huang, Y. Zhu, and A. Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.
- [17] S. Y. Gadre, M. Wortsman, G. Ilharco, L. Schmidt, and S. Song. Cows on pasture: Baselines and benchmarks for language-driven zero-shot object navigation. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 23171–23181, 2023.
- [18] Google. Introducing gemini 2.0: Our new ai model for the agentic era. <https://blog.google/technology/google-deepmind/google-gemini-ai-update-december-2024/>, Dec 2024.
- [19] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6325–6334, 2017.
- [20] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.
- [21] Q. Hu, B. Yang, S. Khalid, W. Xiao, N. Trigoni, and A. Markham. Sensaturban: Learning semantics from urban-scale photogrammetric point clouds. *International Journal of Computer Vision*, 130(2):316–343, 2022.
- [22] J. Huang and J. Zhang. A survey on evaluation of multimodal large language models. *arXiv preprint arXiv:2408.15769*, 2024.

- [23] D. A. Hudson and C. D. Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6693–6702, 2019.
- [24] A. Hurst, A. Lerer, A. P. Goucher, A. Perelman, A. Ramesh, A. Clark, A. Ostrow, A. Welihinda, A. Hayes, A. Radford, et al. GPT-4o System Card. *arXiv preprint arXiv:2410.21276*, 2024.
- [25] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, M. Deitke, K. Ehsani, D. Gordon, Y. Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.
- [26] Y. Kuang, H. Lin, and M. Jiang. Openfmnav: Towards open-set zero-shot object navigation via vision-language foundation models. *arXiv preprint arXiv:2402.10670*, 2024.
- [27] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. Gonzalez, H. Zhang, and I. Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP '23*, page 611–626, New York, NY, USA, 2023. Association for Computing Machinery.
- [28] J. Lee, T. Miyanishi, S. Kurita, K. Sakamoto, D. Azuma, Y. Matsuo, and N. Inoue. City-nav: Language-goal aerial navigation dataset with geographic information. *arXiv preprint arXiv:2406.14240*, 2024.
- [29] X. Lei, Z. Yang, X. Chen, P. Li, and Y. Liu. Scaffolding coordinates to promote vision-language coordination in large multi-modal models. *arXiv preprint arXiv:2402.12058*, 2024.
- [30] B. Li, Y. Zhang, D. Guo, R. Zhang, F. Li, H. Zhang, K. Zhang, P. Zhang, Y. Li, Z. Liu, and C. Li. LLaVA-onevision: Easy visual task transfer. *Transactions on Machine Learning Research*, 2025.
- [31] F. Li, R. Zhang, H. Zhang, Y. Zhang, B. Li, W. Li, Z. Ma, and C. Li. LLaVA-NeXT-Interleave: Tackling multi-image, video, and 3d in large multimodal models. *arXiv preprint arXiv:2407.07895*, 2024.
- [32] J. Li, D. Li, C. Xiong, and S. Hoi. BLIP: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 12888–12900. PMLR, 17–23 Jul 2022.
- [33] L. Li, G. Chen, H. Shi, J. Xiao, and L. Chen. A survey on multimodal benchmarks: In the era of large ai models. *arXiv preprint arXiv:2409.18142*, 2024.
- [34] S. Liu, H. Zhang, Y. Qi, P. Wang, Y. Zhang, and Q. Wu. AerialVLN: Vision-and-Language Navigation for UAVs. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15338–15348, Los Alamitos, CA, USA, Oct 2023. IEEE Computer Society.
- [35] X. Liu, H. Yu, H. Zhang, Y. Xu, X. Lei, H. Lai, Y. Gu, H. Ding, K. Men, K. Yang, S. Zhang, X. Deng, A. Zeng, Z. Du, C. Zhang, S. Shen, T. Zhang, Y. Su, H. Sun, M. Huang, Y. Dong, and J. Tang. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv: 2308.03688*, 2023.
- [36] X. Liu, T. Zhang, Y. Gu, I. L. Iong, S. XiXuan, et al. VisualAgentBench: Towards large multimodal models as visual foundation agents. In *The Thirteenth International Conference on Learning Representations*, 2025.
- [37] Y. Liu, H. Duan, Y. Zhang, B. Li, S. Zhang, W. Zhao, Y. Yuan, J. Wang, C. He, Z. Liu, et al. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, pages 216–233. Springer, 2025.
- [38] Y. Long, W. Cai, H. Wang, G. Zhan, and H. Dong. InstructNav: Zero-shot system for generic instruction navigation in unexplored environment. In *8th Annual Conference on Robot Learning*, 2024.

- [39] A. Majumdar, G. Aggarwal, B. Devnani, J. Hoffman, and D. Batra. ZSON: zero-shot object-goal navigation using multimodal goal embeddings. In *Proceedings of the 36th International Conference on Neural Information Processing Systems, NIPS '22*, Red Hook, NY, USA, 2022. Curran Associates Inc.
- [40] A. Majumdar, A. Ajay, X. Zhang, P. Putta, S. Yenamandra, M. Henaff, S. Silwal, P. Mcvay, O. Maksymets, S. Arnaud, et al. OpenEQA: Embodied question answering in the era of foundation models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16488–16498, 2024.
- [41] K. Marino, M. Rastegari, A. Farhadi, and R. Mottaghi. Ok-vqa: A visual question answering benchmark requiring external knowledge. In *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, pages 3195–3204, 2019.
- [42] MetaAI. Llama 3.2: Revolutionizing edge ai and vision with open, customizable models. <https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/>, 2024.
- [43] Mistral. Pixtral large. <https://mistral.ai/news/pixtral-large/>, Jan 2025.
- [44] S. Nasiriany, F. Xia, W. Yu, T. Xiao, J. Liang, I. Dasgupta, A. Xie, D. Driess, A. Wahid, Z. Xu, et al. Pivot: Iterative visual prompting elicits actionable knowledge for vlms. In *International Conference on Machine Learning*, pages 37321–37341. PMLR, 2024.
- [45] D. Paglieri, B. Cupiał, S. Coward, U. Piterbarg, M. Wolczyk, A. Khan, E. Pignatelli, Ł. Kuściński, L. Pinto, R. Fergus, J. N. Foerster, J. Parker-Holder, and T. Rocktäschel. BALROG: Benchmarking agentic LLM and VLM reasoning on games. In *The Thirteenth International Conference on Learning Representations*, 2025.
- [46] A. Pardyl, G. Rypeść, G. Kurzejamski, B. Zieliński, and T. Trzciński. Active visual exploration based on attention-map entropy. In E. Elkind, editor, *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 1303–1311, 8 2023. Main Track.
- [47] A. Pardyl, M. Wronka, M. Wolczyk, K. Adamczewski, T. Trzciński, and B. Zieliński. Adaglimpse: Active visual exploration with arbitrary glimpse position and scale. In *Computer Vision – ECCV 2024*. Springer Nature Switzerland, 2024. Main Track.
- [48] A. Parnami and M. Lee. Learning from few examples: A summary of approaches to few-shot learning. *arXiv preprint arXiv:2203.04291*, 2022.
- [49] R. Pryzant, D. Iter, J. Li, Y. T. Lee, C. Zhu, and M. Zeng. Automatic prompt optimization with "gradient descent" and beam search. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.
- [50] W. Qiu, F. Zhong, Y. Zhang, Z. X. Siyuan Qiao, T. S. Kim, Y. Wang, and A. Yuille. UnrealCV: Virtual worlds for computer vision. *ACM Multimedia Open Source Software Competition*, 2017.
- [51] A. Ruoss, F. Pardo, H. Chan, B. Li, V. Mnih, and T. Genewein. LMAct: A benchmark for in-context imitation learning with long multimodal demonstrations. In *Forty-second International Conference on Machine Learning*, 2025.
- [52] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9339–9347, 2019.
- [53] S. Seifi, A. Jha, and T. Tuytelaars. Glimpse-Attend-and-Explore: Self-Attention for Active Visual Exploration. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16117–16126, Los Alamitos, CA, USA, Oct 2021. IEEE Computer Society.
- [54] S. Shah, D. Dey, C. Lovett, and A. Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In M. Hutter and R. Siegwart, editors, *Field and Service Robotics*, pages 621–635, Cham, 2018. Springer International Publishing.

- [55] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. Li, Y. Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.
- [56] A. Singh, V. Natarajan, M. Shah, Y. Jiang, X. Chen, D. Batra, D. Parikh, and M. Rohrbach. Towards VQA models that can read. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [57] J. Sun, J. Wu, Z. Ji, and Y.-K. Lai. A survey of object goal navigation. *IEEE Transactions on Automation Science and Engineering*, 22:2292–2308, 2025.
- [58] G. Team, P. Georgiev, V. I. Lei, R. Burnell, L. Bai, A. Gulati, G. Tanzer, D. Vincent, Z. Pan, S. Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.
- [59] P. Wang, S. Bai, S. Tan, S. Wang, Z. Fan, J. Bai, K. Chen, X. Liu, J. Wang, W. Ge, Y. Fan, K. Dang, M. Du, X. Ren, R. Men, D. Liu, C. Zhou, J. Zhou, and J. Lin. Qwen2-VL: Enhancing vision-language model’s perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.
- [60] W. Wang, Z. Chen, W. Wang, Y. Cao, Y. Liu, Z. Gao, J. Zhu, X. Zhu, L. Lu, Y. Qiao, and J. Dai. Enhancing the reasoning ability of multimodal large language models via mixed preference optimization. *arXiv preprint arXiv:2411.10442*, 2024.
- [61] X. Wang, D. Yang, Z. Wang, H. Kwan, J. Chen, W. Wu, H. Li, Y. Liao, and S. Liu. Towards realistic uav vision-language navigation: Platform, benchmark, and methodology. *arXiv preprint arXiv:2410.07087*, 2024.
- [62] P. Wu, Y. Mu, K. Zhou, J. Ma, J. Chen, and C. Liu. Camon: Cooperative agents for multi-object navigation with llm-based conversations. *arXiv preprint arXiv:2407.00632*, 2024.
- [63] Y. Wu, X. Tang, T. Mitchell, and Y. Li. Smartplay : A benchmark for LLMs as intelligent agents. In *The Twelfth International Conference on Learning Representations*, 2024.
- [64] F. Xia, A. R. Zamir, Z. He, A. Sax, J. Malik, and S. Savarese. Gibson env: Real-world perception for embodied agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [65] Q. Xie, T. Zhang, K. Xu, M. Johnson-Roberson, and Y. Bisk. Reasoning about the unseen for efficient outdoor object navigation. *arXiv preprint arXiv:2309.10103*, 2023.
- [66] K. Yadav, J. Krantz, R. Ramrakhya, S. K. Ramakrishnan, J. Yang, A. Wang, J. Turner, A. Gokaslan, V.-P. Berges, R. Mootaghi, O. Maksymets, A. X. Chang, M. Savva, A. Clegg, D. S. Chaplot, and D. Batra. Habitat challenge 2023. <https://aihabitat.org/challenge/2023/>, 2023.
- [67] B. Yu, H. Kasaei, and M. Cao. L3MVN: Leveraging large language models for visual target navigation. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3554–3560, 2023.
- [68] W. Yu, Z. Yang, L. Li, J. Wang, K. Lin, Z. Liu, X. Wang, and L. Wang. MM-Vet: evaluating large multimodal models for integrated capabilities. In *Proceedings of the 41st International Conference on Machine Learning, ICML’24*. JMLR.org, 2024.
- [69] P. Zhang, X. Li, X. Hu, J. Yang, L. Zhang, L. Wang, Y. Choi, and J. Gao. VinVL: Revisiting visual representations in vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5579–5588, June 2021.
- [70] Y. Zhang, Z. Ma, J. Li, Y. Qiao, Z. Wang, J. Chai, Q. Wu, M. Bansal, and P. Kordjamshidi. Vision-and-language navigation today and tomorrow: A survey in the era of foundation models. *Transactions on Machine Learning Research*, 2024. Survey Certification.
- [71] Y. Zhao, J. Huang, J. Hu, X. Wang, Y. Mao, D. Zhang, Z. Jiang, Z. Wu, B. Ai, A. Wang, W. Zhou, and Y. Chen. SWIFT: A scalable lightweight infrastructure for fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 29733–29735, Apr. 2025.

- [72] G. Zhou, Y. Hong, and Q. Wu. NavGPT: Explicit reasoning in vision-and-language navigation with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 7641–7649, 2024.
- [73] Y. Zhou, A. I. Muresanu, Z. Han, K. Paster, S. Pitis, H. Chan, and J. Ba. Large language models are human-level prompt engineers. In *The Eleventh International Conference on Learning Representations*, 2022.

A Impact Statement

In this paper, we focus on the abstract problem of visual exploration – how to interact with the environment to locate the objects of interest. To evaluate this ability in VLMs, we propose a benchmark designed around flying a UAV to find objects in urban and natural environments. This is relevant to real-world applications with positive societal impact, such as search-and-rescue missions, forest fire detection, or personal assistance. However, autonomous UAVs also pose risks as they can be misused by bad agents for surveillance or military operations. We implore users to use their best judgment for the use of the benchmark. Our benchmark is not designed for surveillance or military applications. We urge other researchers to ensure that their benchmarks (including FlySearch derivatives) and models are not directly or indirectly optimized for malicious objectives. We encourage researchers to evaluate their systems for potential biases or unintended behaviors that could lead to misuse. Finally, we would like to highlight the importance of regulatory oversight and the need for clear ethical guidelines in the deployment of autonomous UAVs.

B Funding Transparency Statement

Funding. All financial activities supporting the submitted work are listed in the Acknowledgments section of the main paper.

Competing Interests. At the time of the publication, Maciej Wołczyk is working at Google. However, he did the majority of the work on this paper while at IDEAS NCBR and has not been involved in benchmarking Google models. The authors declare that they have no competing interests and no financial relationships with any entity that could be perceived as influencing the content of this work.

C Benchmark specification

In this section we present the parameters of benchmark scenario generation and evaluation details.

C.1 Scenario parameters

Table S1: Parameters in scenario generation: In this table, we present value ranges for variables that have an impact on the search trajectory. We sample from them uniformly at random while creating a scenario. FS-Anomaly-1 has identical value ranges to FS-1, except for the searched object type and searched object asset. Furthermore, in FS-1 and FS-Anomaly-1 there will be a clear line of sight from the starting position of the agent to the object. In FS-2, we relax this condition and only validate that the object is not under an obstacle (e.g. is not hidden under a bridge, but can be behind a building). Lastly, objects can be placed anywhere in the forest environment and in one of 31852 semantically correct locations in the city environment.

Parameter	FS-1		FS-2
	Value range (Forest)	Value range (City)	Value range (City)
Seed	$\mathbb{N} \cap [0, 1000000000]$	$\mathbb{N} \cap [0, 1000000000]$	$\mathbb{N} \cap [0, 1000000000]$
Agent's starting height (h) [m]	$\mathbb{N} \cap [30, 100]$	$\mathbb{N} \cap [30, 100]$	$\mathbb{N} \cap [100, 125]$
Agent's starting position offset [m]	$[-0.5h, 0.5h]$	$[-0.5h, 0.5h]$	$[-0.95h, 0.95h]$
Searched object type	{campsite, trash, person, fire, building}	{construction works, crowd, trash, fire, car}	
Searched object coordinates	Anywhere on the map	38152 possible placements	
Searched object asset	Dependent on the object type	Dependent on the object type	
Sun elevation angle	$[10, 90]^\circ$ over horizon	45° over horizon	$[10, 90]^\circ$ over horizon
Sun azimuth angle	$[0, 360]^\circ$	110°	$[0, 360]^\circ$
Tree density	$[0.0, 0.3]$	N/A	N/A
Rock density	$[0.0, 0.1]$	N/A	N/A
Object visibility	Visible from agent's starting position		Not under an obstacle

We present all variables that can be used to generate a new scenario in Table S1, such as whether the searched object should be visible from the UAV's initial location. In Table S2, we describe additional parameters that do not impact the scenario generation, but govern details of evaluation – such as:

- Search area bounds – a rectangle centered on agent’s initial position, describing bounds inside of which agent can move. Note that this only serves as a limitation on agent’s movement and does not impact the scenario generation process,
- Maximum altitude – if an agent’s action would bring it above that threshold, it is considered invalid and the agent is asked to issue a new instruction,
- Whether agent can move beyond its *current* view – in FS-1 and FS-Anomaly-1 the agent is prevented from leaving visible area, discouraging hallucination,
- Number of available actions,
- Number of possible *consecutive* retries if agent’s action is considered invalid and needs to be redone,
- Whether the agent should also receive an image showcasing how the target object should look like from above.

Table S2: **Additional parameters:** In this table, we present configurations used during evaluations in FS-1, FS-Anomaly-1 and FS-2. Number of possible retries in case of invalid action was introduced to prevent LLMs from "hanging" by constantly performing invalid actions. As such, this limit was not present while evaluating human baselines. Similarly, we allow humans to fly out of their current view even in FS-1. We note that FS-1 and FS-Anomaly-1 use the exact same configuration, while more search-oriented FS-2 uses a slightly different one. In this table, h denotes starting height of the UAV.

Parameter	FS-1 / FS-Anomaly-1	FS-2
Search area bounds [m]	400×400	$2h \times 2h$
Maximum altitude [m]	120	300
Agent can move beyond its current view	No (Yes for humans)	Yes
Number of available actions	10	20
Number of possible retries if action is invalid	5 (∞ for humans)	5 (∞ for humans)
Object type specification modality	Text	Text + Image

C.2 Target details

The targets in the City environment are as follows:

- Road construction works – a construction zone at the side of a road,
- Crowd – a group of over 30 randomly generated people standing close together,
- Large trash pile – a random pile of trash (bags, car tires, barrels, metal sheets, etc.),
- Fire – a burning car or pile of trash with fire and smoke,
- Car – a randomized car or truck with a specific color (we provide the agent with the color and type).

While in the Forest the goal is to find:

- Campsite – a randomly generated set of camping equipment, including at least one tent,
- Trash pile – a pile of car tires, barrels, metal scrap, and other waste (different than in the forest environment),
- Person – a human laying flat on the ground, representing an injured hiker,
- Forest fire – a burning forest area, emitting a large cloud of smoke,
- Building – an abandoned building in various styles.

D Human baselines

In order to evaluate human performance on FlySearch we provide a web-based interface for human testers. The interface is designed to be consistent with VLM evaluation procedure, while allowing for efficient human interaction. Therefore, instead of using XML formatted text for communication, the interface uses standard HTML forms and buttons. In Fig. S1 the welcome screen is presented.

The user is shown the model prompt (with XML formatting instructions omitted) and can select the between FS-1 and FS-2. In Fig. S2 and Fig. S3 the communication screens for FS-1 and FS-2 respectively are shown. On each step the evaluator has to either fill in X, Y, Z action coordinates and press the MOVE button, or if they believe they have fulfilled the task criterion they have to press the FOUND button. Finally, after clicking FOUND or exhausting possible moves the user is notified if they have succeeded or failed the scenario. During the entire evaluation, users can report any issues with a built-in bug report form.

Source code of the interface and associated back-end server is included in FlySearch codebase. The study was conducted with participation of human manual software testers (employees). The study did not pose any risks to participants. No Personally Identifiable Information was gathered in the process. The study was conducted according to institutional regulations on data gathering and processing and was subject to internal approval process.

FlySearch demo Your name Mode **FS-1** ▼ **CONNECT TO SERVER**

Your task
A similar prompt is given to the AI agent

You are in command of a UAV, tasked with finding a target.

You should fly **BELOW** 10 meters above the object and then reply with "FOUND". Being lower (closer to the object) (like 9, 8, or less meters) is good, being higher than that (like 11, 12, or more meters) is bad.

You may not be able to see the object in the first image, so you need to perform a careful search. Your performance will be evaluated based on whether the object was at most 10 meters below the drone when you replied with "FOUND". The object **MUST** be in your field of view when you reply with "FOUND". You must be centered on the object.

There is a grid overlaid on each image you are presented with. It is meant to (roughly) communicate which point will be in drone's center of vision if you move in that direction. Note that height of the drone is not represented in the grid.

To move the drone in a certain direction, use the following format: (x, y, z). For example, if you want to fly to the place denoted as (10, 10) on the grid without changing the altitude, you should reply with (10, 10, 0).

x and y are the coordinates on the grid, and z is the altitude difference. For example, (0, 0, -10) means that you are moving 10 meters down. This is especially important, since you need to get close to the object in question. tag should contain the move you are making.

If you find the object, fly below 10 meters relative to it and reply with "FOUND". Remember, it must be in your field of view when you reply with "FOUND" and you must be 10 meters above it or closer. Being too far away is not acceptable.

You shouldn't move into coordinates that are outside of your view. Otherwise, you may hit something which is not ideal. You can make a limited number of moves, dependent on the scenario. Your altitude cannot exceed 300 meters.

The search area is limited to what would be visible from the starting position if there were no buildings or obstacles. The object is within this area. You may not fly outside of it.

[To start a new game fill in your username and press the connect button in the top right corner.](#)

REPORT ISSUE

Figure S1: **Human study – start screen:** Screenshot of the FlySearch human interface welcome screen, containing instructions derived from the benchmark prompt. The screen also contains a field for the participant identifier (nickname), a scenario generator switch (FS-1/FS-2), connection button and bug report button.

E Fine-tuning details

To provide a reasonable baseline score for fine-tuning VLMs for spatial reasoning we train the Qwen VL 2.5 7B [5] model on the Forest environment and evaluate it on the City environment in both FS-1 and FS-2. The Qwen VL 2.5 7B model is selected as it is the best performing model in the *small open-source model* category in our evaluation. Since the City and Forest environments share almost no graphical assets and are vastly different visually, training the model on Forest allows for a fair comparison with other models on the City environment. That is, the model cannot learn to visually recognize relevant objects and their placement in the evaluated scenario.

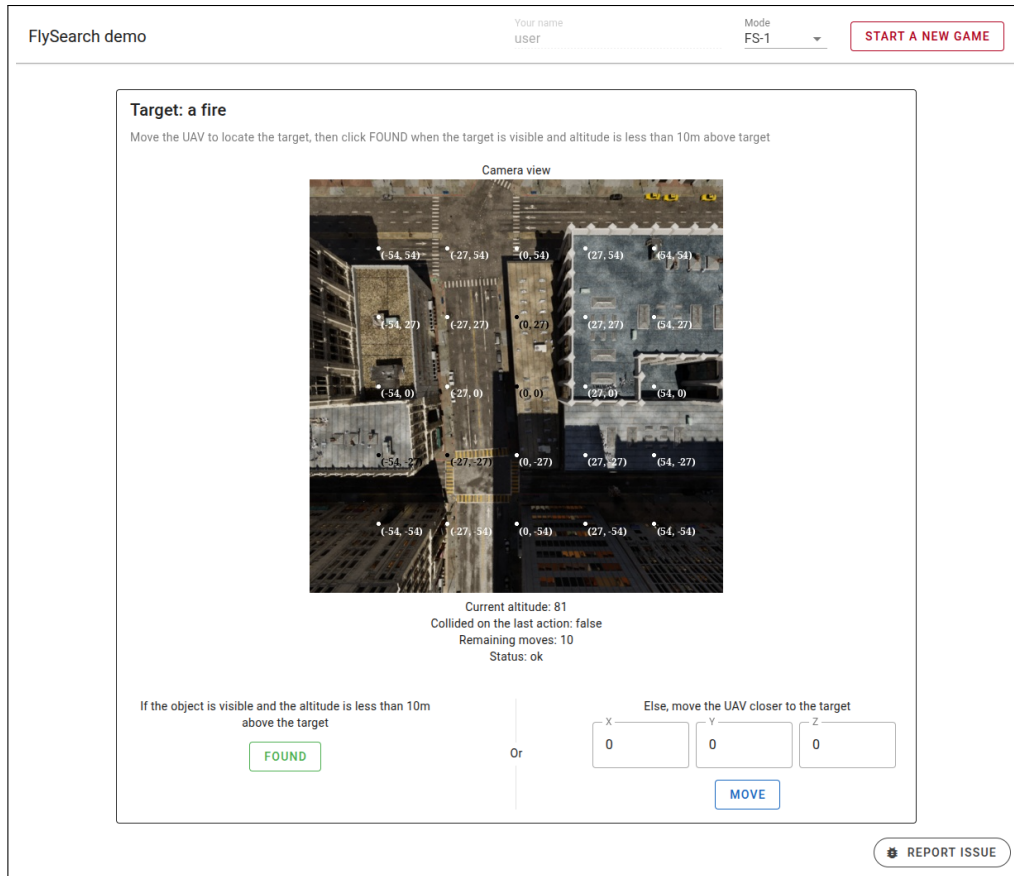


Figure S2: **Human study – FS-1 screen:** Screenshot of the FlySearch human interface FS-1 screen. The page contains the target description, agent camera view, altitude, object and boundary collision information, and action controls. The image is overlaid with grid coordinates, exactly as the image provided to the evaluated VLM. All state and action elements are the same as those provided to the model, but for the formatting (a form instead of XML formatted text).

Using the Forest environment, we generate 6750 unique exploration scenarios. In each scenario, we pre-record a flight trajectory from the starting point to the target. The trajectory is created by sampling 30 evenly spaced steps over a straight line. To each flight step a random position offset in range of $[-10, 10]$ m, and capture camera views using our simulator. Finally, we augment the dataset by generating 10 new trajectories from each captured episode, randomly dropping some of the flight steps. Resulting trajectories have between 1 and 10 steps, depending on the distance between start and target. Each trajectory is further cut at random step and formatted as a conversation to be completed by the VLM, resulting in 67500 training samples.

Standard supervised fine-tuning does not provide sufficient improvement in results (11.1% on FS-1 City, compared to 1.5% of the base model). This is likely due to the fact, that each scenario can be solved in multiple ways, not just the ideal trajectory. Therefore, we apply GRPO fine-tuning [55] in step-wise, offline manner. That is, using the pre-generated training dataset we train the model to predict one next action. We define the reward function for the step with the following pseudo-code:

Listing 1: Reward for GRPO fine-tuning (pseudo-code).

```
def reward(model_output):
    if model_output is not parsable:
        return 0

    reasoning, action = parse(model_output)

    reasoning_reward = min(1, len(reasoning) / 100)
```

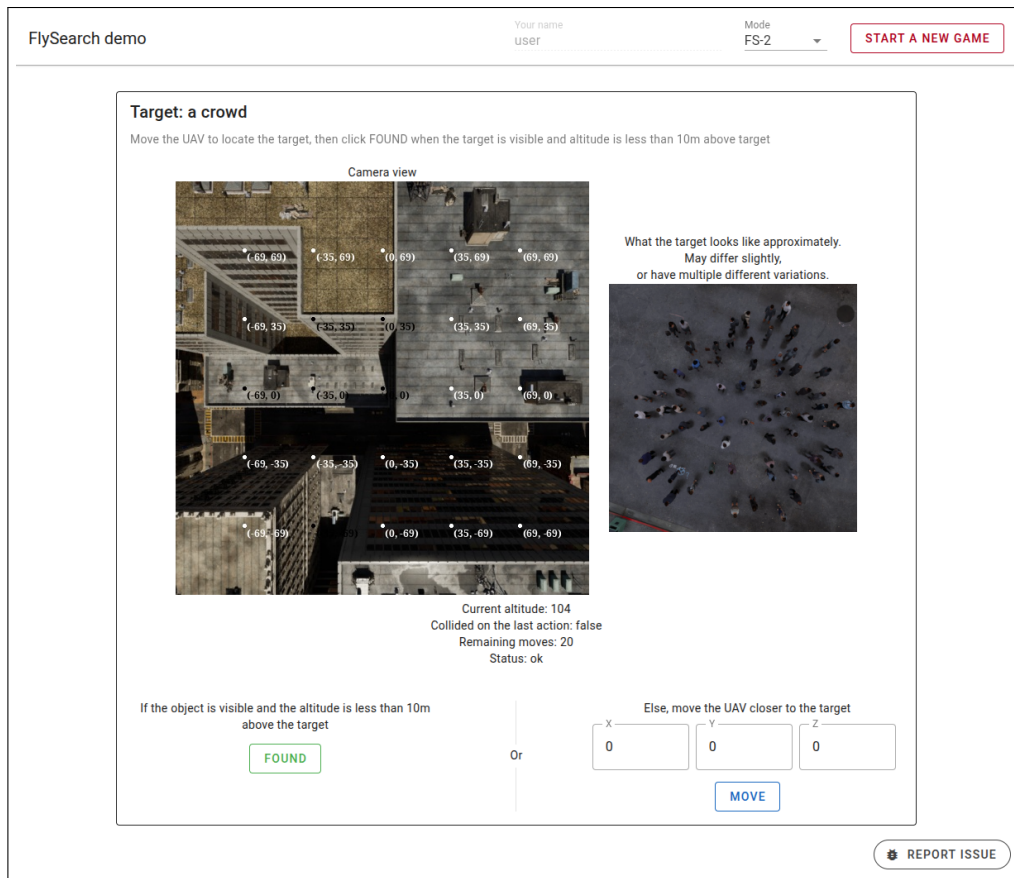


Figure S3: **Human study – FS-2 screen:** Screenshot of the FlySearch human interface FS-2 screen. The page contains the same elements as for FS-1 plus an additional visual description of the target (a neutral background and lightning picture, not an actual image of the searched object).

```

if action == 'FOUND':
    if target_located:
        return 1 + reasoning_reward
    else:
        return 0 + reasoning_reward

action_reward = (current_distance - next_distance) / current_distance
action_reward = max(min(action_reward, 1), 0) # clip between 0 and 1

if target_located:
    # action should be FOUND, decrease reward for further moves
    mod = max(0., (altitude - targed_height compare)) / 10
    action_reward = (mod ** .5) * action_reward

return reasoning_reward + action_reward

```

Therefore, the model is incentivized to both provide at least 100 tokens of reasoning output and to move closer to the target in each step. When the model can respond with FOUND action the reward for further moves towards the target is decreased. Finally, if the agent responds with FOUND it receives a binary reward based on the success.

We perform the GRPO fine-tuning using the Swift library [71], and use LoRA [20] to speed up the training. Moreover, we freeze the vision encoder part of the model to prevent overfitting on visual features. Bellow are the full parameters of the training:

Listing 2: GRPO fine-tuning command.

```
NPROC_PER_NODE=2 CUDA_VISIBLE_DEVICES=0,1,2,3 swift rlhf \
--rlhf_type grpo \
--model Qwen/Qwen2.5-VL-7B-Instruct \
--train_type lora \
--dataset $DATA_PATH \
--num_train_epochs 1 \
--per_device_train_batch_size 8 \
--per_device_eval_batch_size 8 \
--learning_rate 1e-5 \
--gradient_accumulation_steps 1 \
--eval_steps 100 \
--save_steps 100 \
--save_total_limit 2 \
--logging_steps 5 \
--dataloader_num_workers 4 \
--attn_impl flash_attn \
--use-hf \
--torch_dtype bfloat16 \
--deepspeed zero2 \
--target_modules all-linear lm_head \
--lora_alpha 32 \
--lora_rank 8 \
--external_plugins rewardplugin.py \
--reward_funcs fly_search \
--max_completion_length 1024 \
--warmup_ratio 0.05 \
--num_generations 16 \
--dataset_num_proc 4 \
--temperature 0.9 \
--log_completions true \
--use_vllm true \
--vllm_gpu_memory_utilization 0.9 \
--vllm_limit_mm_per_prompt '{"image": 10}' \
--num_infer_workers 2 \
--async_generate true \
--seed 42 \
--split_dataset_ratio 0
```

The entire process takes several hours using 4 NVIDIA H100 GPUs. After fine-tuning the model is evaluated with standard FS-1 City and FS-2 settings, achieving 27.0% and 0.0% accordingly. Moreover, it achieves 57.0% accuracy on the Forest environment it was trained upon.

F Success criterion implementation

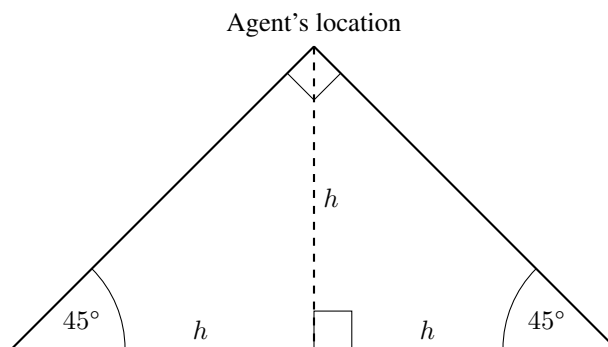


Figure S4: **Success criterion illustration:** The searched object's center must be inside of the camera's cone of view. We assume the camera's field of view is 90° .

For a trajectory to be considered successful, at the end of it the agent must be located less than 10 meters above the object's highest point. Furthermore, the object must be seen from the agent's location.

To check whether the agent can see the object, we calculate agent's cone of view and check whether object's center is inside of the cone of view. This calculation is straightforward, because camera's field of view is 90° . It easily follows that agent at altitude h sees an area of $2h \times 2h$ meters. We provide a drawing to illustrate that fact in Fig. S4.

In some cases, this implementation may cause misclassifications as a failure, due to the agent „seeing“ an object in a human sense (e.g. a significant part of it's edges), but not having object's center in its cone of view. To avoid this issue, we specify in the prompt that agent should be *centered* on the object.

G Prompt templates

In this section we present the prompt templates for all benchmark tasks.

G.1 Prompt template for FS-1 and FS-Anomaly-1

Listing 3: Prompt template for FS-1 and FS-Anomaly-1.

```
<Context>
  You are in command of a UAV, tasked with finding {TARGET}.
</Context>

<Objective>
  You should fly BELOW 10 meters above the object and then reply with "FOUND".
  Being lower (closer to the object) (like 9, 8, or less meters) is good, being
  higher than that (like 11, 12, or more meters) is bad.

  You may not be able to see the object in the first image, so you need to perform
  a careful search. Your performance will be evaluated based on whether the
  object was at most 10 meters below the drone when you replied with "FOUND". The
  object MUST be in your field of view when you reply with "FOUND". You must be
  centered on the object.
</Objective>

<Coordinates>
  There is a grid overlaid on each image you are presented with. It is meant to (
  roughly) communicate which point will be in drone's center of vision if you
  move in that direction. Note that height of the drone is not represented in the
  grid.
</Coordinates>

<Controls>
  <Action space>
    To move the drone in a certain direction, use the following format:
    <Action>(x, y, z)</Action>. For example, if you want to fly to the place
    denoted as (10, 10) on the grid without changing the altitude, you should reply
    with <Action>(10, 10, 0)</Action>.

    x and y are the coordinates on the grid, and z is the altitude difference.
    For example, <Action>(0, 0, -10)</Action> means that you are moving 10 meters
    down. This is especially important, since you need to get close to the object
    in question.

  </Action space>

  <Formatting>

    Your each response should contain XML <Reasoning> tag and <Action> tag.
    <Reasoning> tag should contain your reasoning for the move you are making.
```

<Action> tag should contain the move you are making.

If you find the object, fly below 10 meters relative to it and reply with "FOUND". Remember, it must be in your field of view when you reply with "FOUND" and you must be 10 meters above it or closer. Being too far away is not acceptable.

For example:

<Reasoning>This yellow point might be the object in question. I need to go lower to check for that. If it's not the object in question, I will continue the search. I will also slightly go to the north.</Reasoning>

<Action>(5, 0, -30)</Action>

</Formatting>

<Limitations>

You shouldn't move into coordinates that are outside of your view. Otherwise, you may hit something which is not ideal.

You can make at most 9 moves. Your altitude cannot exceed 120 meters. Your search area is 400x400m from the drone's starting position.

</Limitations>

</Controls>

G.2 Prompt template for FS-2

Listing 4: Prompt template for FS-2.

<Context>

You are in command of a UAV, tasked with finding {TARGET}.

</Context>

<Objective>

You should fly BELOW 10 meters above the object and then reply with "FOUND". Being lower (closer to the object) (like 9, 8, or less meters) is good, being higher than that (like 11, 12, or more meters) is bad.

You may not be able to see the object in the first image, so you need to perform a careful search. Your performance will be evaluated based on whether the object was at most 10 meters below the drone when you replied with "FOUND". The object MUST be in your field of view when you reply with "FOUND". You must be centered on the object.

</Objective>

<Coordinates>

There is a grid overlaid on each image you are presented with. It is meant to (roughly) communicate which point will be in drone's center of vision if you move in that direction. Note that height of the drone is not represented in the grid.

</Coordinates>

<Controls>

<Action space>

To move the drone in a certain direction, use the following format: <Action>(x, y, z)</Action>. For example, if you want to fly to the place denoted as (10, 10) on the grid without changing the altitude, you should reply with <Action>(10, 10, 0)</Action>.

x and y are the coordinates on the grid, and z is the altitude difference. For example, <Action>(0, 0, -10)</Action> means that you are moving 10 meters down. This is especially important, since you need to get close to the object in question.

</Action space>

<Formatting>

Your each response should contain XML <Reasoning> tag and <Action> tag.
<Reasoning> tag should contain your reasoning for the move you are making.
<Action> tag should contain the move you are making.

If you find the object, fly below 10 meters relative to it and reply with "FOUND". Remember, it must be in your field of view when you reply with "FOUND" and you must be 10 meters above it or closer. Being too far away is not acceptable.

For example:

<Reasoning>This yellow point might be the object in question. I need to go lower to check for that. If it's not the object in question, I will continue the search. I will also slightly go to the north.</Reasoning>
<Action>(5, 0, -30)</Action>

</Formatting>

<Limitations>

You shouldn't move into coordinates that are outside of your view. Otherwise, you may hit something which is not ideal.
You can make at most {glimpses - 1} moves. Your altitude cannot exceed 300 meters.

The search area is limited to what would be visible from the starting position if there were no buildings or obstacles. The object is within this area. You may not fly outside of it.

</Limitations>

</Controls>

Furthermore, along with the textual description of the searched object we pass the image, showcasing how it should look like from above, with a following annotation:

Listing 5: Annotation of searched object's image in FS-2.

The object you're looking for is similar to this. This is NOT the drone's current view.

H Additional Experiments

In this section we present additional experimental results, evaluating design choices of our benchmark.

Table S3: Performance of Gemini 2.0 Flash and Pixtral-Large on ablations.

	Setting	City		Forest	
		Gemini	Pixtral	Gemini	Pixtral
FS-1	Baseline	41.5%	21.5%	42.5%	38.0%
	Compass actions	17.5%	21.0%	17.5%	22.0%
	No grid overlay	17.0%	15.5%	31.5%	20.0%
FS-Anomaly	Baseline	25.0%	4.0%	46.0%	26.0%
	Anomaly with ID	34.0%	7.0%	59.0%	34.0%

Impact of the action space. We test how changing the action format affects performance by replacing the default Cartesian (x, y, z) movements with compass-style commands that specify a direction (north, south, east, west, up, down) and distance. This type of action space is common in previous works. As shown in Table S3, this change significantly degrades performance for both

Table S4: **FS-Anomaly-1 results:** Success rates ($\pm$ standard errors) of the evaluated models.

Model	FS-Anomaly-1		
	Overall (%)	Forest (%)	City (%)
GPT-4o	27.0 ± 3.1	39.0 ± 4.9	15.0 ± 3.6
Claude 3.5 Sonnet	27.5 ± 3.2	37.0 ± 4.9	18.0 ± 3.9
Gemini 2.0 flash	35.5 ± 3.4	46.0 ± 5.0	25.0 ± 4.4
Phi 3.5 vision	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
InternVL-2.5 8B MPO	3.5 ± 1.3	6.0 ± 2.4	1.0 ± 1.0
Llava-Interleave-7b	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
Qwen2.5-VL 7B	2.8 ± 1.2	3.7 ± 2.1	2.0 ± 1.4
Qwen2-VL-72B	7.5 ± 1.9	10.0 ± 3.0	5.0 ± 2.2
Llava-Onevision 72b	8.5 ± 2.0	11.0 ± 3.1	6.0 ± 2.4
Pixtral-Large	15.0 ± 2.5	26.0 ± 4.4	4.0 ± 2.0

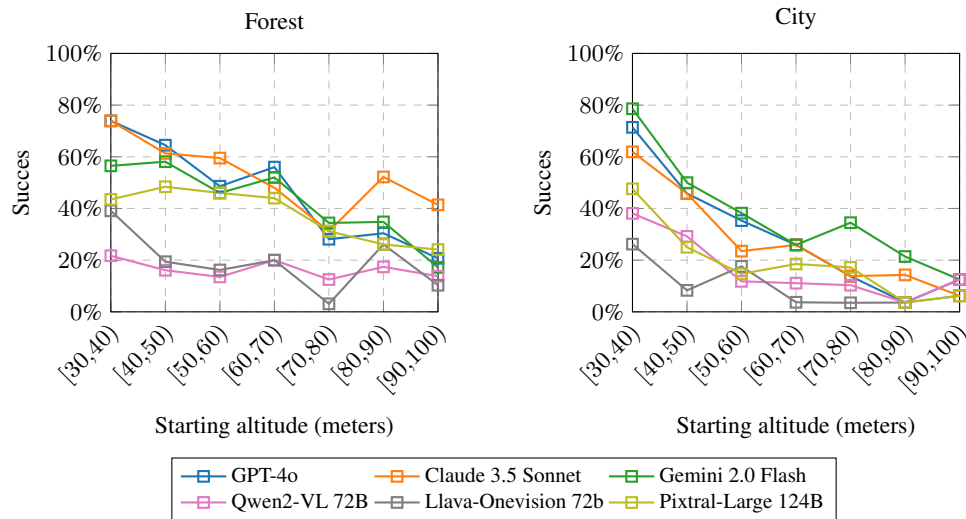


Figure S5: **Success rate per starting altitude interval:** In this figure, we plot the success rates of the models as a function of the initial altitude interval for both the Forest and the City environments in the FS-1 challenge. The performance drop that comes with increasing altitude is more pronounced for the City environment.

Gemini 2.0 Flash and Pixtral-Large. Gemini’s accuracy drops from 42.5% to 17.5% in Forest and from 41.5% to 17.5% in City. Pixtral falls from 38% to 22% in Forest, with little change in City (21.5% to 21%). These results highlight the importance of a flexible, continuous action space. Compass-style movement restricts fine-grained control and appears to limit the models’ ability to effectively search.

Impact of the grid overlay. To assess the importance of the grid overlay on each glimpse, we run an ablation where glimpses are shown without the grid. As shown in Table S3, removing the grid leads to a notable drop in performance for both Gemini 2.0 Flash and Pixtral-Large. Gemini’s success rate falls from 42.5% to 31.5% in Forest, and more dramatically in City—from 41.5% to 17%. Pixtral shows a similar trend, dropping from 38% to 20% in Forest and from 21.5% to 15.5% in City. These results suggest that the grid plays a key role in helping models maintain spatial awareness, especially in more complex environments. Without it, both localization and goal-oriented planning appear to suffer. This result is consistent with previous works on vision-language coordination [29].

Impact of hidden vs explicit anomaly categories. In the FS-Anomaly setting, the model must identify an object that doesn’t belong in the surrounding environment. In the base version, the

anomaly category is implicit, requiring the model to infer what is out of place. In this ablation, we explicitly tell the model what object to look for.

As shown in Table S3, providing the anomaly identity significantly improves performance. Gemini's accuracy increases from 46% to 59% in Forest and from 25% to 34% in City. Pixtral also improves, from 26% to 34% in Forest and from 4% to 7% in City.

This result suggests that models often struggle with anomaly detection due to semantic ambiguity rather than visual perception. Explicitly stating the anomaly class helps disambiguate the task and leads to more focused and successful exploration.

FS-Anomaly-1 full results. We provide full results of the FS-Anomaly-1 benchmark with distinction between Forest and City environment performance in Table S4.

Impact of the starting height. Finally, we verify how the results vary depending on the starting height. As seen in Figure S5, the success rate decreases as the starting altitude (and therefore distance to the target) increases. This is particularly significant in the City environment, where the success rate for the best models drops by half, likely due to the high number of distracting objects. In contrast, in the generally easier Forest environment, this decline is less pronounced and occurs mainly in the upper half of the altitude range.

I Sim2real gap

The main design goal of FlySearch is the evaluation of VLM exploration and 3D spatial reasoning capabilities, with UAV object navigation serving as a realistic assessment task. However, we acknowledge the fact that FlySearch can serve as a platform for testing methods designed specifically for UAV control, including non-VLM solutions. Therefore, in this section, we discuss in depth the differences between FlySearch and real-life UAV control, underlining issues that were simplified or omitted in order for the problem to be addressable with existing foundational VLMs.

Environment. FlySearch uses Unreal Engine 5 to provide near-photorealistic simulation of highly complex environments, with both procedurally generated and handcrafted content, using a vast set of high-resolution assets and generation rules. This setup allows for world simulation on par with the latest video games. However, the real world presents far greater complexity and unpredictability than any video game. This might lead to discrepancies between the behavior of VLM in simulated and real-world environments.

World dynamics. The real world contains a vast number of dynamic elements (moving cars, walking people, flying birds, etc.) that cannot be fully simulated even with modern video game engines. Moreover, FlySearch omits some of the flight-related dynamics, such as sudden gusts of wind or abrupt weather changes, that were deemed too complex for current foundational VLMs.

Sensor simulation. Our benchmark limits sensor inputs to a downward-facing camera and altitude reading with 1 m accuracy. The camera sensor is simulated using real-time ray-tracing capabilities of the engine, accounting for camera auto-exposure, depth of view, light direction (time of the day), fixed light cloud cover, and atmospheric haze. This enables an overall realistic simulation of a modern digital camera, but simplifies the weather element and omits any data loss or low-light condition emulation. Furthermore, real-life platforms may feature multiple cameras, LIDAR/RADAR, GPS units, IMU, and other sensor modalities that are not simulated in FlySearch.

Target objects. The searched object is stationary (may be animated but remains in the same location), unique for the scenario (visually similar but semantically different objects may exist on the map), and visible from the top. In real-world situations, the object can move freely in the environment, including moving inside buildings, and the description of the object provided to the agent can match multiple objects. It may require more complex search strategies and the cooperation of multiple agents.

UAV control. FlySearch uses high-level relative movement actions with 1 m resolution. This can be implemented in real-life UAVs using existing autopilot software such as PX4 Autopilot or DroneKit, which can accommodate wind changes and drift. However, a more direct control of the platform

could speed up the exploration process in practice and make it less dependent on GPS or inertial navigation systems. At the same time, this makes the task more complex for both VLM models and human evaluators.

Other simplifications. The benchmark limits agent interaction with the world to up to 20 steps due to the model context length limitations of most open-source VLMs. In real-life problems, the location of an object in a large environment may require significantly more observations. In addition, real-world UAVs are subject to hardware and software failures, none of which are simulated in FlySearch. In future work, when VLMs can solve our toughest challenge set, we plan to address some of the above limitations, making our benchmark more realistic.

J Example trajectories

The sample trajectories of different VLMs on FS-1 and FS-2 scenarios are provided in a separate file in the supplementary material, as well as at: <https://github.com/gmum/FlySearch/blob/main/docs/example-trajectories.pdf>

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [Yes]

Justification: The main claims in the abstract and introduction accurately reflect the paper's contributions and scope, as they consistently present the research objectives, methodology, and key findings, aligning with the detailed content and conclusions provided in the main body of the paper.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: The limitations are discussed in Section 5.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.

- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: There are no theoretical results in the paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Experiments are described in the main body of the paper. Moreover, there are full details in the appendix section. Full source code is provided.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [\[Yes\]](#)

Justification: The data and code are publicly available.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [\[Yes\]](#)

Justification: The full details are presented in the core paper, the appendix, or available in the code.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.

- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: The main empirical results are accompanied by standard errors.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The paper provides sufficient information about the type of compute resources used.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: The research does conform with the Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [\[Yes\]](#)

Justification: The societal impacts are presented in Appendix A due to space restrictions.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [\[NA\]](#)

Justification: The benchmark poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [\[Yes\]](#)

Justification: The creators and original owners of assets that are used in the paper are properly credited.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [\[Yes\]](#)

Justification: New assets are documented and the documentation is provided alongside them.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [\[Yes\]](#)

Justification: The instructions given to human subjects are the same as those given to the models in the benchmark. Full details are presented in Appendix D.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [Yes]

Justification: The paper includes benchmark scores achieved by human manual software testers (employees). As clarified in Appendix D, the study does not pose any risk to participants. The study was conducted according to institutional regulations on data gathering and processing and was subject to internal approval process.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The LLMs are only used for editing and as a test subject.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.