See, Point, Fly: A Learning-Free VLM Framework for Universal Unmanned Aerial Navigation

Chih Yao Hu[2*] **Yang-Sen Lin**[1*] **Yuna Lee**[1] **Chih-Hai Su**[1] **Jie-Ying Lee**[1]
Shr-Ruei Tsai[1] **Chin-Yang Lin**[1] **Kuan-Wen Chen**[1] **Tsung-Wei Ke**[2] **Yu-Lun Liu**[1]
[1] National Yang Ming Chiao Tung University, [2] National Taiwan University

https://spf-web.pages.dev

Abstract: We present See, Point, Fly (SPF), a training-free aerial vision-and-language navigation (AVLN) framework built atop vision-language models (VLMs). SPF is capable of navigating to any goal based on any type of free-form instructions in any kind of environment. In contrast to existing VLM-based approaches that treat action prediction as a text generation task, our key insight is to consider action prediction for AVLN as a 2D spatial grounding task. SPF harness VLMs to decompose vague language instructions into iterative annotation of 2D waypoints on the input image. Along with the predicted traveling distance, SPF transforms predicted 2D waypoints into 3D displacement vectors as action commands for UAVs. Moreover, SPF also adaptively adjusts the traveling distance to facilitate more efficient navigation. Notably, SPF performs navigation in a closed-loop control manner, enabling UAVs to follow dynamic targets in dynamic environments. SPF sets a new state of the art in DRL simulation benchmark, outperforming the previous best method by an absolute margin of 63%. In extensive real-world evaluations, SPF outperforms strong baselines by a large margin. We also conduct comprehensive ablation studies to highlight the effectiveness of our design choice. Lastly, SPF shows remarkable generalization to different VLMs.

Keywords: Vision-Language Models, Zero-shot UAV Navigation, 2D-to-3D Waypoint Prompting

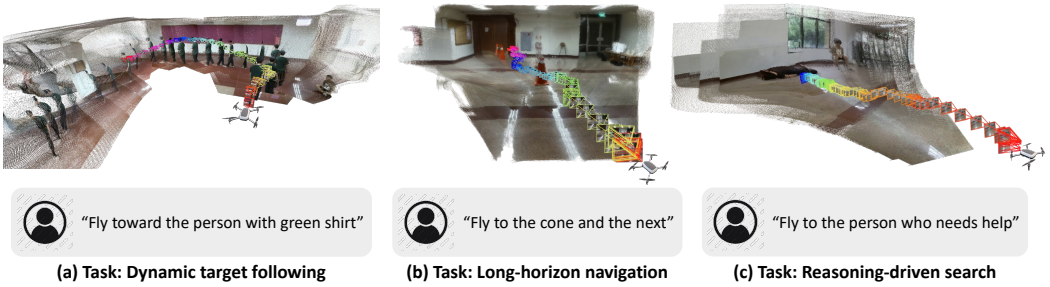
1 Introduction

The rapid development of unmanned aerial vehicles (UAVs) has revolutionized applications from environmental monitoring to security patrol. However, autonomous UAV navigation remains challenging due to requirements for strong visual reasoning in unstructured environments, language understanding for user instructions, and high-level task planning with low-level action control [1]. These autonomous UAV navigation tasks are often framed as aerial vision-and-language (AVLN) tasks [2, 3].

The autonomous UAV navigation tasks are commonly framed as aerial vision-and-language (AVLN) tasks [2, 3]. Conventional methods primalily adopt end-to-end policy learning frameworks which consist of a text and vision encoder that maps language instructions and visual observations into latent representations, followed by a policy head that converts these representations into UAV actions [4, 5, 6, 7, 8, 9, 10]. The entire models are trained on a curated set of expert demonstrations [11, 12, 13, 14]. However, due to the limited scale and diversity of the training data, these methods fail to generalize to unseen environments or task instructions. In contrast, recent works explore a training-free direction that directly converts Vison Large Language Models (VLM) into

*The first two authors contributed equally

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.



(a) Task: Dynamic target following (b) Task: Long-horizon navigation (c) Task: Reasoning-driven search

Figure 1: **Zero-shot language-guided UAV control.** (a) The UAV continually replans to keep pace with a moving person. (b) The UAV chains multiple goals across the hall. (c) The UAV locates the person on the ground and navigates around obstacles. Coloured 3D boxes mark successive camera viewpoints, revealing the UAV's full flight trajectory over the reconstructed point cloud. All waypoints are generated directly by the vision-language model, with *no* task-specific training.

AVLN policies [15, 16, 17, 18]. As VLMs are trained on large-scale internet data, these models have demonstrated not only rich common-sense knowledge of the world, strong capabilities in visual/language understanding, reasoning and planning, but also, strong generalization to novel environments and tasks [19, 20, 21].

How to repurpose VLMs that generate texts into embodied agents that generate physical actions has attracted increasing interest in robotics [22, 23, 24, 25], while the research direction is still underexplored in AVLN. Existing VLM-based approaches to AVLN build atop a direct solution, that considers action prediction simply as a text-generation task. VLMs are prompoted to output either continuous actions [16] or pre-defined skills [15, 17, 18], in terms of texts. Despite the simplicity of these methods, they have two obvious problems: (1) embodied agents need to execute fine-grained actions, while texts are not suitable to represent high-precision floating numbers; (2) these VLMs have not been trained on aerial navigation data to predict 3D actions for navigation. In contrast, our key insight is to consider action prediction for AVLN as a 2D spatial grounding task. Instead of predicting 3D actions directly, we harnesses VLMs to annotate 2D waypoints [26, 27, 28, 29] on the image, which do not require any domain knowledge of AVLN but general spatial understanding [30, 31]. As these 2D waypoints are grounded in the visual scene, they inherently contain precise action information. These 2D waypoints can then be transformed into 3D actions using the camera information.

Notably, we do not introduce the concept of predicting 2D waypoints for action selection—similar ideas have been explored in both robot manipulation and navigation [28, 26, 27, 20]. For example, RT-Trajectory [26] leverages VLMs to directly label 2D waypoints on the image, which are then used by a separately trained policy network to predict corresponding actions. PIVOT [28], in contrast, samples multiple candidate actions as 2D waypoints and employs a VLM to select the most appropriate one for execution. In this work, we build on this general idea and adapt it to the AVLN setting. Our method requires no additional neural network training, yet it significantly outperforms PIVOT, which is also a training-free approach.

We introduce See, Point, Fly (SPF), a novel VLM-based AVLN framework that navigates to any goal based on any free-form instructions in any environment. At the core of our method is a VLM [20] that conditions on the current scene and language instructions, and outputs the 2D waypoints in terms of pixel locations. These 2D waypoints are unprojected into unit-length 3D positions based on the camera parameters. These 3D positions denote the relative 3D actions to the current UAV location. To enhance the navigation speed, we propose an adaptive controller module that adjust the scale of the actions based on the distance between the UAV and the target. Since our method naturally enables closed-loop control of the UAV, as shown in Fig. 1, UAVs are capable of following dynamic targets. Moreover, building atop VLMs, our method can easily tackle long-horizon and even ambiguous task instructions in a zero-shot manner.

We test SPF on a simulation and a real-world benchmark. Our method outperforms prior state-of-the-art, TypeFly [15] by a large margin. We show that our method works well across a wide range of tasks, including long-horizon, abstract, and dynamic navigation tasks. We also conduct an extensive ablation study to validate the effectiveness of each design choice.

In summary, our contributions are: (1) We propose a state-of-the-art AVLN framework that generalizes to novel scenes and free-form instructions; (2) We set a new state-of-the-art in the DRL simulator [32] simulation benchmark, outperforming prior SOTAs with a margin of 63% in success rate; (3) We set a new state of the art in the real-world benchmark, outperforming prior SOTAs with a margin of 82% in success rate.

2 Related Work

End-to-end policy learning in UAV navigation. The goal of policy learning is to train a model that outputs control actions for UAVs. Policy learning for UAV navigation can be broadly categorized into imitation learning (IL) [33] and reinforcement learning (RL) [34]. The objective of RL is to maximize cumulative rewards through interaction with the environment. These methods have achieved strong performance in drone racing [35, 36, 37], collision avoidance [38] and optimal quadrotor control [39, 40, 41, 42, 43]. Recent work has also explored NeRF-based environments for validating autonomous navigation policies [44], providing realistic simulation environments for training and testing. However, RL often struggles with tasks involving long temporal horizons and sparse reward signals, and have shown limited success in navigation tasks.

On the other hand, the objective of IL is to maximize the likelihood of the actions from expert demonstrations [11, 12, 13, 14]. Prior works focus on exploring effective policy architectures for navigation. GSMN [4] proposes to construct intermediate map representations inside the policy, to facilitate action predictions. CIFF [5] utilizes a mask generator to annotate the goal location on the image, followed by recurrent neural network to predict the corresponding UAV actions. LLMIR [6] and AVDN [8] instead build policies based on conditional transformers. Recent advances in robotic control have also demonstrated the effectiveness of diffusion-based methods for precise manipulation tasks [45], suggesting potential applications in UAV control. Notably, due to the limited capacity of language encoders inside these methods, they are incapable of handling free-form instructions in recent AVLN benchmarks [2, 3, 9, 46]. To enhance language understanding, recent works propose to fine-tune large language models as navigation policies [7, 10].

While these end-to-end learning frameworks show good evaluation performance in similar settings as training data, due to the limited scale and diversity of the training data, these methods fail to generalize to unseen environments or task instructions. We instead explore a training-free alternative that deploy VLMs for AVLN in a zero-shot manner.

Vison language models for training-free UAV Navigation. Converting VLMs, originally designed for text generation, into embodied agents that output action controls has drawn increasing attention. A direct solution is to prompt VLMs to generate UAV actions in textual forms. For instance, [16] proposes to construct semantic map representation that localizes task-related objects in the bird's-eye aerial map, with VLMs. Prompted with the map representations, VLMs output the corresponding 2D actions to reach the target on the map. In stead of outputting continuous actions, TypeFly [15], UAV-VLA [17], Flex [47] and GeoNav [17] prompt VLMs to generate discrete actions, selected from a predefined set of navigation skills. While both paradigms simplify the interface between language models and control systems, they restrict the UAV's action space, often leading to suboptimal motion trajectories and reduced control precision. In stark contrast, our SPF considers action prediction as a 2D spatial grounding task. We utilize recent VLMs' [20] strong capabilities in affordance annotation, prompting VLMs to label 2D waypoints [26, 27, 28, 29] on the image. Transforming these 2D points into 3D actions with the camera information results in more effective UAV control.

3 Method

We formulate UAV navigation as an iterative target-reaching process in 3D space. At each timestep t, the system processes the current visual observation $I_t \in \mathbb{R}^{H \times W \times 3}$ along with a natural language instruction ℓ to determine the next motion. Formally, we define a policy $\pi(\cdot \mid \ell, I_t)$ that maps the observation-instruction pair to a 3D motion command $m_t \in \mathcal{A}$, where the action space $\mathcal{A} \subseteq \mathbb{R}^3$ represents feasible displacement vectors.

Figure 2: **Pipeline overview.** A camera frame and user instructions enter a frozen vision-language model, which returns a structured JSON with a 2D waypoint and any obstacle boxes. An Action-to-Control layer converts this output into low-level velocity commands (yaw, throttle, pitch) that steer the UAV. The loop repeats until the task is completed.

We leverage vision-language models (VLMs) to implement the policy π, transforming complex and vague navigation nature language instructions into sequences of interpretable waypoint decisions. This approach decomposes the navigation task into discrete spatial reasoning steps that can be efficiently converted into UAV control signals, while remaining robust to diverse environments and instruction types.

As illustrated in Fig. 2, our system runs an iterative perception-action loop with three stages: (1) Given ℓ and I_t, we use the VLM G to produce a structured spatial understanding, 2D waypoints and moving step sizes (Sec. 3.1), (2) We transform the predicted 2D waypoint and step size into a 3D displacement vector, yielding executable low-level commands m_t (Sec. 3.2 and Sec. 3.3), and (3) A lightweight reactive controller continuously updates the observation, replans using the VLM, and executes the resulting motion commands in a closed-loop manner (Sec. 3.4).

By outsourcing high-level spatial reasoning to the VLM and employing a lightweight geometric controller, our method achieves robust zero-shot UAV navigation directly from language—without relying on skill libraries, external depth sensors, policy optimization, or model training.

3.1 VLM-based Obstacle-Aware Action Planning

We frame the first stage of our method as a structured visual grounding task, where a VLM G processes an egocentric UAV camera observation $I_t \in \mathbb{R}^{H\times W\times 3}$ alongside a natural language instruction ℓ specifying the desired UAV task. Conditioned on this input, the VLM outputs a probability distribution $P_G(w \mid \ell, I_t)$ over candidate waypoint plans $w \in \mathcal{W}$, where $\mathcal{W}$ represents the discrete space of feasible spatial waypoint sequences. We define the intermediate spatial plan O_t as the most likely waypoint sequence under this distribution:

$$O_t = \arg\max_{w\in\mathcal{W}} P_G(w \mid \ell, I_t). \tag{1}$$

The output $O_t = \{u, v, d_{\text{VLM}}\}$ specifies a 3D navigation target in image space, where (u, v) are pixel coordinates and $d_{\text{VLM}} \in \{1, 2, \ldots, L\}$ is a discretized depth label. Importantly, d_{VLM} represents the VLM's prediction of intended travel distance along the UAV's forward direction (positive y-axis in body frame), rather than a sensored depth measurement.

When obstacle-avoidance mode is activated, the VLM is further constrained to generate waypoints that guide the UAV toward the goal while avoiding intersection with detected object bounding boxes, promoting safe navigation through cluttered environments. By formulating UAV control through this visual grounding approach, we transform complex spatial reasoning into a computationally efficient task that enables robust, zero-shot, obstacle-aware navigation without requiring iterative optimization or exhaustive low-level action sampling.

3.2 Adaptive Travel Distance Scaling

Although VLMs can infer high-level spatial plans from visual inputs, they often lack a precise understanding of real-world 3D geometry and UAV navigation intuition possessed by human pilots. Consequently, motion commands derived directly from VLM outputs may result in overly aggressive or unsafe movements, particularly in cluttered or constrained environments.

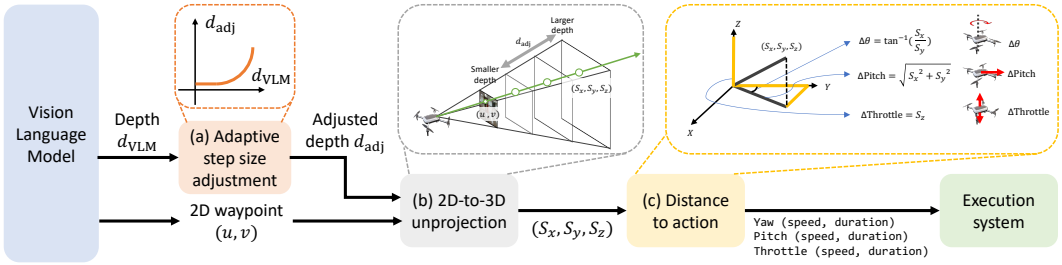


Figure 3: **Control-geometry details of our VLM-driven flight loop.** A frozen vision-language model first predicts a 2D waypoint (u, v) and a discrete depth cue d_{VLM}. (a) A nonlinear scaling curve converts d_{VLM} into an adaptive step size d_{adj}, letting the UAV take larger strides in open space and smaller ones near obstacles. (b) The pair (u, v, d_{adj}) is unprojected through the pin-hole model to a 3D displacement vector (S_x, S_y, S_z) in the UAV's body frame. (c) This vector is decomposed into control primitives: yaw $\Delta\theta = \tan^{-1}(S_x/S_y)$, pitch $\Delta\text{Pitch} = \sqrt{S_x{}^2 + S_y{}^2}$, and throttle $\Delta\text{Throttle} = S_z$. These quantities are sent as timed velocity commands by the execution layer. The perception, planning, and control cycle repeats until the language instruction is fulfilled.

To address this limitation, as shown in Fig. 3 (a), we employ a non-linear scaling curve that converts the discrete depth label d_{VLM} into an adjusted step size d_{adj}:

$$d_{\text{adj}} = \max\left(d_{\min},\ s \times \left(\frac{d_{\text{VLM}}}{L}\right)^{p}\right), \tag{2}$$

where s is a global scaling factor, p controls the nonlinearity of the scaling curve, and $d_{\min}$ specifies a lower bound on the step size to ensure safety.

This adaptive scaling approach enables the UAV to take larger steps in open areas while executing smaller, more cautious movements near targets and obstacles. The UAV can thus adapt its trajectory naturally to scene geometry without requiring explicit 3D maps or external depth sensors. This capability is particularly valuable for lightweight UAVs where onboard perception and strict latency constraints limit the feasibility of deploying traditional depth-sensing hardware.

3.3 Policy Mapping from Image Space to 3D Actions

Given the structured VLM output $O_t = \{u, v, d_{\text{adj}}\}$, our system transforms this image-space waypoint into executable 3D motion commands. This transformation defines the core of our reactive policy, enabling the UAV to navigate toward visually grounded targets using only RGB inputs.

As depicted in Fig. 3 (b), we unproject the predicted 2D waypoint (u, v) together with the adjusted depth d_{adj} through a pin-hole camera model to obtain a 3D displacement vector (S_x, S_y, S_z), which is later decomposed into yaw, pitch and throttle commands.

To compute the desired 3D displacement vector (S_x, S_y, S_z), the angular projection of the pixel location onto the camera's field of view is used:

$$S_x = u \cdot d_{\text{adj}} \cdot \tan(\alpha), \quad S_y = d_{\text{adj}}, \quad S_z = v \cdot d_{\text{adj}} \cdot \tan(\beta), \tag{3}$$

where α and β are the camera's horizontal and vertical half field-of-view angles, respectively. The forward motion S_y is aligned with the UAV's body-frame y-axis.

3.4 Reactive Control Loop Execution

Operating within a closed-loop control framework, desired 3D displacements are decomposed into UAV control primitives: pitch, yaw, and throttle, as illustrated in Fig. 3 (c). Each control primitive is converted into a velocity-duration pair, where the duration is derived from the magnitude of the required adjustment and a predefined constant speed. Commands are enqueued into an execution queue and sent to the UAV with temporal synchronization, allowing for smooth, responsive, and low-latency control through continuous correction. This approach enables efficient adaptation to dynamic environments without requiring complex trajectory optimization. For more technical details, please refer to the supplementary material.

4 Experimental Results

Experimental Setup. We evaluated our approach in both simulated and real-world environments. For simulation, we employed the high-fidelity DRL simulator [32], which serves as a standard benchmark from the Drone Racing League competition and effectively bridges the simulation-to-real gap through accurate physics modeling and realistic sensor simulation. For real-world validation, we implemented our system on a DJI Tello EDU drone platform, controlled through the Python SDK using low-level `rc` velocity commands. We conducted extensive tests across various indoor environments (office spaces, corridors, living areas) and outdoor settings (parks, campus walkways) with different lighting conditions, obstacle densities, and visual complexities to thoroughly assess real-world performance.

Metrics. We evaluated performance using two metrics: **Success Rate (SR)**, the percentage of trials where the drone reached its target without collisions, and **Completion Time**, measuring duration from movement initiation to task completion. These metrics together assess both reliability and efficiency across diverse navigation scenarios.

Task Categories. Our evaluation framework includes 6 distinct task categories designed to assess the robustness and versatility of VLM-guided UAV control across diverse navigation scenarios: (1) **Navigation:** Navigating to specified static targets or objects / locations in the real-world. (2) **Obstacle Avoidance:** Reaching designated targets while avoiding static and dynamic obstacles. (3) **Long Horizon:** Multi-stage navigation sequences requiring sustained performance and compositional planning across extended spatial and temporal scales. (4) **Reasoning:** Tasks requiring contextual interpretation, spatial inference, and environmental understanding beyond literal instruction following. (5) **Search:** Target localization tasks where targets initially lie outside the UAV's field of view. (6) **Follow:** Identifying and tracking real-world objects or people.

We design a total of 23 tasks for simulation and 11 tasks for real-world evaluation, across task categories. Each task was executed 5 times per method to account for execution variability. Performance metrics were aggregated by category to assess domain-specific capabilities. Complete task specifications and evaluation protocols are detailed in the supplementary material.

Baselines. We benchmark our approach against three representative methods for language-guided UAV control: (1) **TypeFly** [15]: A language-driven approach that uses GPT-4 to interpret natural language commands and select appropriate actions from a predefined skill library. While effective for known tasks, this method's reliance on a fixed action space fundamentally limits its zero-shot generalization capabilities; (2) **PIVOT** [28]: A visual-language approach that overlays candidate 2D waypoints on the input image as visual prompts, from which a VLM selects the most appropriate waypoint for navigation. This approach requires pre-generating and evaluating multiple candidate paths rather than directly predicting optimal waypoints; (3) **Plain VLM**: An ablation of our method that directly prompts a VLM to predict drone actions in textual form without our proposed structured output formulation, spatial transformation, or adaptive depth scaling techniques.

We used the publicly released implementation for TypeFly, while PIVOT and Plain VLM were re-implemented following their published methodologies to ensure fair comparison within our evaluation framework.

4.1 Performance Evaluation

We demonstrate the effectiveness of our method with the quantitative results shown in Table 1. In simulation, our approach achieves **93.9%** average success rate, significantly outperforming PIVOT (28.7%) and TypeFly (0.9%, limited by its predefined skill library). In particular, our framework excels in complex scenarios that require spatial reasoning and planning, such as obstacle avoidance (92% vs. 16% for PIVOT), long-horizon tasks (92% vs. 28% for PIVOT) and search tasks (92% vs. 36% for PIVOT).

Table 1: **Success rate (%) comparison across task categories.** Our framework significantly outperforms TypeFly [15] and PIVOT [28] baselines in both high-fidelity simulation and real-world DJI Tello experiments. We achieve 93.9% and 92.7% overall success rates in simulation and real-world settings, respectively. Note that Search tasks were exclusively evaluated in simulation, while Follow tasks were only tested in real-world settings due to environment constraints.

Method	Navigation	Obstacle Avoid	Long Horizon	Reasoning	Search / Follow	Overall Avg.
Simulation						
TypeFly [15]	1/25	0/25	0/25	0/15	0/25	0.9%
PIVOT [28]	11/25	4/25	7/25	2/15	9/25	28.7%
SPF (Ours)	**25/25**	**23/25**	**23/25**	**14/15**	**23/25**	**93.9%**
Real-world						
TypeFly [15]	1/5	3/10	5/10	2/20	2/10	23.6%
PIVOT [28]	0/5	1/10	0/10	2/20	0/10	5.5%
SPF (Ours)	**5/5**	**7/10**	**9/10**	**20/20**	**10/10**	**92.7%**

(a) Obstacle avoidance (b) Target identification (c) Pattern searching

Figure 4: **Qualitative comparison of flight trajectories in the simulator.** Trajectory of our method is colored in **green**, PIVOT [28] in **blue**, and TypeFly [15] in **purple**. The absence of a colored path indicates the baseline failed to issue any fly command. Full videos are included in the supplementary materials.

Real-world experiments confirmed our method's effectiveness with a **92.7%** average success rate. In contrast, TypeFly struggled with object recognition and language understanding, while PIVOT performed poorly in real-world settings, demonstrating the advantages of our structured visual grounding approach. We evaluated completion time across 5 representative real-world tasks including obstacle avoidance, long horizon, reasoning, and follow categories. As shown in Fig. 6, SPF not only successfully completed all tasks where both baselines often failed, but also achieved faster completion times. These results demonstrate our method's superior efficiency and reliability in diverse scenarios.

We present qualitative results in simulation (Fig. 4) and in the real-world (Fig. 5). Our results suggest that our SPF is more effective in generating smooth navigation trajectories, avoiding obstacles, and reaching the target than TypeFly and PIVOT.

4.2 Ablations

We conducted an ablation study to evaluate the effectiveness of each model component in simulation. Our study includes five simulated tasks and three real-world tasks across different categories. The results are presented in Table 2.

Structured Prompting and Grounding. We compared three VLM-based action prediction approaches: our method (prompting VLM to label 2D waypoints on images), plain VLM (predicting actions as text) and PIVOT (selecting from candidate 2D points on images). Our approach significantly outperforms alternatives with a success rate of 100% versus just 7% for plain VLM and 40% for PIVOT on navigation tasks, demonstrating the effectiveness of our structured visual grounding formulation.

VLM. Our method performs robustly across multiple VLMs: Gemini 2.5 Pro, Gemini 2.0 Flash, and GPT-4.1 all achieved 100% success rate; Claude 3.7 Sonnet and Llama 4 Maverick reached 93.3%; and even Gemini 2.0 Flash-Lite achieved 87%. This demonstrates our framework's effective generalization across vision-language models of varying capabilities.

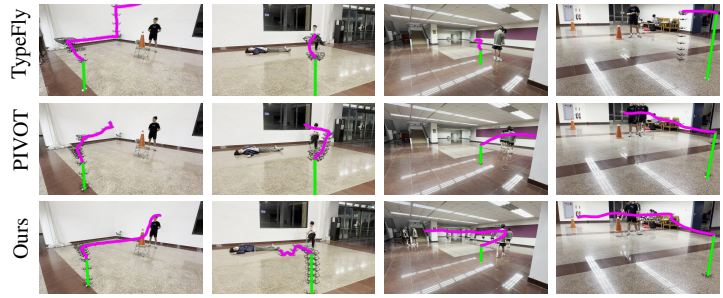


Figure 5: **Qualitative comparison of flight trajectories in the real-world.** Trajectory of our method compared to other baselines in the real-world testing. Take off trajectory is colored in **green** and task trajectory in **magenta**. Please refer to the supplementary materials for full videos.

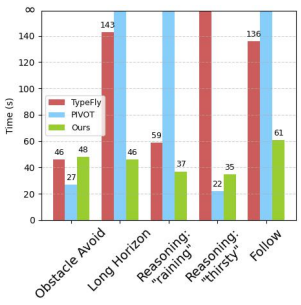


Figure 6: **Completion time by task.** Our approach achieves faster completion times across most tasks, particularly excelling in complex scenarios. Bars capped at ∞ indicate baseline failures.

Table 2: **Ablation of prompting strategies and VLM backbones in simulation.** 2D→3D waypoint prompting (SPF) lifts SR from 40% (PIVOT) to 87% with Flash-Lite and hits 100% on stronger VLMs, whereas plain text generation scores only 7%.

	Action prediction	VLM model	SR (%)
Plain VLM	Text Generation	Gemini 2.0 Flash[48]	7
PIVOT [28]	Visual Prompting	Gemini 2.0 Flash[48]	40
SPF (Ours)	2D Waypoint Labeling	Gemini 2.0 Flash-Lite [48]	**87**
		Gemini 2.0 Flash [48]	**100**
		Gemini 2.5 Pro [48]	**100**
		GPT-4.1 [49]	**100**
		Claude 3.7 Sonnet [50]	**93.3**
		Llama 4 Maverick [51]	**93.3**

Table 3: **Adaptive step-size controller cuts completion time while preserving success.** Across two representative tasks, switching from a fixed step to our adaptive scaling halves flight duration and raises the success ratio to 5/5.

Task	Step	Compl. time	SR
"Fly to the cones and the next."	Fixed	61s	5/5
	Adaptive	**28s**	**5/5**
"I'm thirsty. Find something that can help me."	Fixed	50.25s	4/5
	Adaptive	**35.20s**	**5/5**
"It's raining. Head to the comfiest chair that will keep you dry."	Fixed	47s	5/5
	Adaptive	**30s**	**5/5**

Adaptive Travel Distance Scaling. Our method significantly speeds up the travel time using the proposed adaptive distance scaling. It maintains navigation performance, while reducing the average completion time from 50.25 to 35.20 seconds. The results are presented in Table 3. We refer to the supplementary material for more details of the experimental setup.

Our VLM-Integrated Approach. Our approach generates bounding boxes directly from the Vision-Language Model (VLM) in a single pass, enabling zero-shot generalization and low latency. This offers critical advantages over specialized detectors limited by fixed vocabularies.

Table 4: **Design trade-off for obstacle avoidance.**

Method	Latency	Accuracy (%)	Generalization
Ours (VLM-integrated)	1.077s	88.8	Zero-shot (any object)
+ External Detector (YOLOv8n) [52]	1.726s	72.2	Limited to known classes

Conclusion We presented SPF, a training-free framework that repurposes frozen vision-language models for universal UAV navigation. By casting action prediction as 2D waypoint grounding, then geometrically lifting these points to 3D displacements, our method sidesteps task-specific data collection and policy optimization. A lightweight adaptive controller closes the perception-action loop, yielding smooth flights despite second-level VLM latency. Across 23 simulated and 11 real-world tasks, SPF achieved **93.9%** and **92.7%** success rates, respectively, substantially outperforming TypeFly and PIVOT while remaining model-agnostic and hardware-friendly.

Limitations. Despite promising results, our system has limitations. VLM inaccuracies (hallucinations and misinterpretations) can occur, and grounding precision may decrease for small or distant targets. The adaptive step heuristic provides implicit depth but can be imprecise. Performance can be sensitive to prompt phrasing. Reactivity to highly dynamic obstacles is limited by the VLM inference latency (≈1-3s). Finally, VLM-generated search patterns are not guaranteed to be optimal. These limitations highlight avenues for future work, including improving perception robustness, improving grounding mechanisms, reducing system latency for better reactivity, exploring VLM fine-tuning, and developing more sophisticated exploration strategies.

Acknowledgments

This research was funded by the National Science and Technology Council, Taiwan, under Grants NSTC 113-2628-E-A49-023- and 111-2628-E-A49-018-MY4. The authors are grateful to Google, NVIDIA, and MediaTek Inc. for their generous donations. Yu-Lun Liu acknowledges the Yushan Young Fellow Program by the MOE in Taiwan.

References

[1] Y. Chang, Y. Cheng, U. Manzoor, and J. Murray. A review of uav autonomous navigation in gps-denied environments. *Robotics and Autonomous Systems*, 170:104533, 2023.

[2] S. Liu, H. Zhang, Y. Qi, P. Wang, Y. Zhang, and Q. Wu. Aerialvln: Vision-and-language navigation for uavs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15384–15394, 2023.

[3] J. Lee, T. Miyanishi, S. Kurita, K. Sakamoto, D. Azuma, Y. Matsuo, and N. Inoue. Citynav: Language-goal aerial navigation dataset with geographic information. *arXiv preprint arXiv:2406.14240*, 2024.

[4] V. Blukis, N. Brukhim, A. Bennett, R. A. Knepper, and Y. Artzi. Following high-level navigation instructions on a simulated quadcopter with imitation learning. *arXiv preprint arXiv:1806.00047*, 2018.

[5] D. Misra, A. Bennett, V. Blukis, E. Niklasson, M. Shatkhin, and Y. Artzi. Mapping instructions to actions in 3d environments with visual goal prediction. *arXiv preprint arXiv:1809.00786*, 2018.

[6] Z. Chen, J. Li, F. Fukumoto, P. Liu, and Y. Suzuki. Vision-language navigation for quadcopters with conditional transformer and prompt-based text rephraser. In *Proceedings of the 5th ACM International Conference on Multimedia in Asia*, pages 1–7, 2023.

[7] Y. Liu, F. Yao, Y. Yue, G. Xu, X. Sun, and K. Fu. Navagent: Multi-scale urban street view fusion for uav embodied vision-and-language navigation. *arXiv preprint arXiv:2411.08579*, 2024.

[8] Y. Fan, W. Chen, T. Jiang, C. Zhou, Y. Zhang, and X. E. Wang. Aerial vision-and-dialog navigation. *arXiv preprint arXiv:2205.12219*, 2022.

[9] 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.

[10] A. Lykov, V. Serpiva, M. H. Khan, O. Sautenkov, A. Myshlyaev, G. Tadevosyan, Y. Yaqoot, and D. Tsetserukou. Cognitivedrone: A vla model and evaluation benchmark for real-time cognitive task solving and reasoning in uavs. *arXiv preprint arXiv:2503.01378*, 2025.

[11] A. Giusti, J. Guzzi, D. C. Cireşan, F.-L. He, J. P. Rodríguez, F. Fontana, M. Faessler, C. Forster, J. Schmidhuber, G. Di Caro, et al. A machine learning approach to visual perception of forest trails for mobile robots. *IEEE Robotics and Automation Letters*, 1(2):661–667, 2015.

[12] N. Smolyanskiy, A. Kamenev, J. Smith, and S. Birchfield. Toward low-flying autonomous mav trail navigation using deep neural networks for environmental awareness. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4241–4247. IEEE, 2017.

[13] A. Loquercio, A. I. Maqueda, C. R. Del-Blanco, and D. Scaramuzza. Dronet: Learning to fly by driving. *IEEE Robotics and Automation Letters*, 3(2):1088–1095, 2018.

[14] I. Bozcan and E. Kayacan. Au-air: A multi-modal unmanned aerial vehicle dataset for low altitude traffic surveillance. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8504–8510. IEEE, 2020.

[15] G. Chen, X. Yu, N. Ling, and L. Zhong. Typefly: Flying drones with large language model. *arXiv preprint arXiv:2312.14950*, 2023.

[16] Y. Gao, Z. Wang, L. Jing, D. Wang, X. Li, and B. Zhao. Aerial vision-and-language navigation via semantic-topo-metric representation guided llm reasoning. *arXiv preprint arXiv:2410.08500*, 2024.

[17] H. Xu, Y. Hu, C. Gao, Z. Zhu, Y. Zhao, Y. Li, and Q. Yin. Geonav: Empowering mllms with explicit geospatial reasoning abilities for language-goal aerial navigation. *arXiv preprint arXiv:2504.09587*, 2025.

[18] O. Sautenkov, Y. Yaqoot, A. Lykov, M. A. Mustafa, G. Tadevosyan, A. Akhmetkazy, M. A. Cabrera, M. Martynov, S. Karaf, and D. Tsetserukou. Uav-vla: Vision-language-action system for large scale aerial mission generation. *arXiv preprint arXiv:2501.05014*, 2025.

[19] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[20] G. R. Team, S. Abeyruwan, J. Ainslie, J.-B. Alayrac, M. G. Arenas, T. Armstrong, A. Balakrishna, R. Baruch, M. Bauza, M. Blokzijl, et al. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

[21] A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[22] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE, 2023.

[23] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530. IEEE, 2023.

[24] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

[25] K. Lin, C. Agia, T. Migimatsu, M. Pavone, and J. Bohg. Text2motion: From natural language instructions to feasible plans. *Autonomous Robots*, 47(8):1345–1365, 2023.

[26] J. Gu, S. Kirmani, P. Wohlhart, Y. Lu, M. G. Arenas, K. Rao, W. Yu, C. Fu, K. Gopalakrishnan, Z. Xu, et al. Rt-trajectory: Robotic task generalization via hindsight trajectory sketches. *arXiv preprint arXiv:2311.01977*, 2023.

[27] F. Liu, K. Fang, P. Abbeel, and S. Levine. Moka: Open-world robotic manipulation through mark-based visual prompting. *arXiv preprint arXiv:2403.03174*, 2024.

[28] 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. *arXiv preprint arXiv:2402.07872*, 2024.

[29] J.-J. Hwang, R. Xu, H. Lin, W.-C. Hung, J. Ji, K. Choi, D. Huang, T. He, P. Covington, B. Sapp, et al. Emma: End-to-end multimodal model for autonomous driving. *arXiv preprint arXiv:2410.23262*, 2024.

[30] B. Chen, Z. Xu, S. Kirmani, B. Ichter, D. Sadigh, L. Guibas, and F. Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14455–14465, 2024.

[31] A.-C. Cheng, H. Yin, Y. Fu, Q. Guo, R. Yang, J. Kautz, X. Wang, and S. Liu. Spatialrgpt: Grounded spatial reasoning in vision language models. *arXiv preprint arXiv:2406.01584*, 2024.

[32] Drone Racing League. DRL Simulator, 2024. URL https://www.drl.io/drlsim. [Computer software].

[33] D. A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988.

[34] R. S. Sutton, A. G. Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[35] E. Kaufmann, L. Bauersfeld, A. Loquercio, M. Müller, V. Koltun, and D. Scaramuzza. Champion-level drone racing using deep reinforcement learning. *Nature*, 620(7976):982–987, 2023.

[36] Y. Song, A. Romero, M. Müller, V. Koltun, and D. Scaramuzza. Reaching the limit in autonomous racing: Optimal control versus reinforcement learning. *Science Robotics*, 8(82): eadg1462, 2023.

[37] R. Ferede, C. De Wagter, D. Izzo, and G. C. De Croon. End-to-end reinforcement learning for time-optimal quadcopter flight. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6172–6177. IEEE, 2024.

[38] K. Kang, S. Belkhale, G. Kahn, P. Abbeel, and S. Levine. Generalization through simulation: Integrating simulated and real data into deep reinforcement learning for vision-based autonomous flight. In *2019 international conference on robotics and automation (ICRA)*, pages 6008–6014. IEEE, 2019.

[39] E. Kaufmann, A. Loquercio, R. Ranftl, M. Müller, V. Koltun, and D. Scaramuzza. Deep drone acrobatics. *arXiv preprint arXiv:2006.05768*, 2020.

[40] A. Molchanov, T. Chen, W. Hönig, J. A. Preiss, N. Ayanian, and G. S. Sukhatme. Sim-to-(multi)-real: Transfer of low-level robust control policies to multiple quadrotors. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 59–66. IEEE, 2019.

[41] J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter. Control of a quadrotor with reinforcement learning. *IEEE Robotics and Automation Letters*, 2(4):2096–2103, 2017.

[42] M. O'Connell, G. Shi, X. Shi, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung. Neural-fly enables rapid learning for agile flight in strong winds. *Science Robotics*, 7(66): eabm6597, 2022.

[43] G. Shi, X. Shi, M. O'Connell, R. Yu, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung. Neural lander: Stable drone landing control using learned dynamics. In *2019 international conference on robotics and automation (icra)*, pages 9784–9790. IEEE, 2019.

[44] M.-Y. Shen, C.-C. Hsu, H.-Y. Hou, Y.-C. Huang, W.-F. Sun, C.-C. Chang, Y.-L. Liu, and C.-Y. Lee. Driveenv-nerf: Exploration of a nerf-based autonomous driving environment for real-world performance validation. *arXiv preprint arXiv:2403.15791*, 2024.

[45] S.-W. Guo, T.-C. Hsiao, Y.-L. Liu, and C.-Y. Lee. Precise pick-and-place using score-based diffusion networks. In *IROS*, 2024.

[46] Y. Gao, C. Li, Z. You, J. Liu, Z. Li, P. Chen, Q. Chen, Z. Tang, L. Wang, P. Yang, et al. Openfly: A versatile toolchain and large-scale benchmark for aerial vision-language navigation. *arXiv preprint arXiv:2502.18041*, 2025.

[47] M. Chahine, A. Quach, A. Maalouf, T.-H. Wang, and D. Rus. Flex: End-to-end text-instructed visual navigation from foundation model features, 2024.

[48] Google DeepMind. Gemini, 2025. URL https://deepmind.google/technologies/gemini/.

[49] OpenAI. Gpt-4.1, 2025. URL https://openai.com/index/gpt-4-1/.

[50] Anthropic. Claude sonnet 3.7, 2024. URL https://www.anthropic.com/claude/sonnet.

[51] Meta. Llama 4, 2024. URL https://www.llama.com/models/llama-4/.

[52] G. Jocher, A. Chaurasia, and J. Qiu. Ultralytics yolov8, 2023. URL https://github.com/ultralytics/ultralytics. [Computer software].