VLN on the Fly: An Onboard Vision-Language Navigation Stack for Aerial Robots

Marco S. Tayar[†, *], Felipe Tommaselli[*], Gianluca Capezutto[*], Pedro Antonio Rabelo Saraiva[*], Pedro H. V. de Freitas, Lucas Kido, Guilherme Sonego, Ricardo V. Godoy, and Marcelo Becker
University of São Paulo (USP), Brazil

[*]These authors contributed equally to this work. [†] Corresponding author: marcotayar@usp.br

Abstract

Running vision-language navigation fully onboard an aerial robot is hard, since grounding, planning, and control must share limited compute and a single-stage error is difficult to isolate in flight. End-to-end aerial policies fuse these stages into one network, giving up the observability and safety checks a modular stack keeps available. We propose VLN on the Fly, an onboard stack that keeps grounding, planning, and control as separate, inspectable stages. A quantized VLM grounds an instruction to a coarse image cell, depth lifts it to a 3D goal, a fast B-spline planner returns a feasible trajectory, and a pretrained reinforcement learning policy tracks it to motor commands across quadrotors. Across 15 onboard flights over three everyday referents in a controlled indoor volume, the stack reaches the target in 13 of 15 trials with 5.72 cm mean goal error and 39.3% average GPU utilization. In 6 additional cluttered-environment trials, the stack tracks collision-free trajectories under onboard perception gating.

1 Introduction

Aerial robots are increasingly being deployed in human-centered indoor environments, including warehouses, offices, inspection sites, and residential spaces. Indoor deployment introduces GPS denial, cluttered geometry, narrow passages, and layouts built around people or ground vehicles rather than autonomous flight. Operation under such conditions depends on the platform's robustness and the ability of the underlying task-completion models to generalize beyond a fixed environment. Language-conditioned control is promising in this setting because natural-language instructions allow a human to specify task intent directly in a new scenario, without re-engineering the system for each task.

Vision-Language-Action (VLA) models have been used to address this need, with a single end-to-end policy trained to map raw observations and instructions directly to control commands. Applied to navigation, this class is commonly referred to as Vision-Language Navigation (VLN) [1–4], and recent aerial works such as AerialVLA [5], AutoFly [6], and Singer [7] point in promising directions. However, the compositional generalization behind the success of language models has proven harder to achieve in physical autonomy [8]. For aerial robots, each new instruction must be grounded not only in semantic context but also in feasible motion under vehicle dynamics and safety constraints. When grounding, spatial reasoning, and control are encoded inside a single end-to-end policy, limited observability makes failures harder to localize and safety constraints harder to inspect or enforce through standard planning mechanisms such as collision checks, feasibility tests, and recovery behaviors [9].

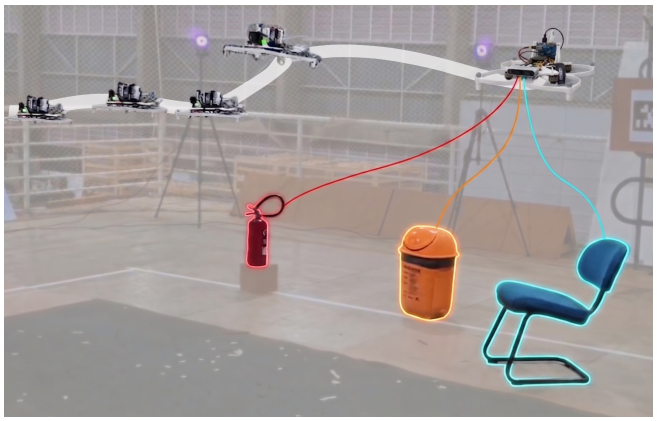

Figure 1: **VLN on the Fly**, an onboard vision-language navigation stack that grounds language instructions and plans and executes trajectories toward named referents, here a fire extinguisher, a trash bin, or a chair.

Recent literature shows that decoupling these models enables desirable behaviors in navigation stacks [10, 11], since separate modules remain observable, reusable path-planning safety guards remain available, and changes to one component of the stack do not require retraining the entire policy. For a decoupled approach, integrating the hierarchical stack, particularly with the flight controller's inner modules, remains a valid concern underexplored in the literature. Concretely, we address three concerns: (i) safety, (ii) hierarchical complexity, and (iii) onboard compute, since running a local Vision-Language Model (VLM) together with a full stack onboard is inherently challenging.

In this work, we propose **VLN on the Fly**, an onboard stack for navigation on aerial vehicles (Fig. 1). We embed a quantized VLM (Qwen-3.5-2B [12]) for coarse grid grounding, where the model identifies the image region containing the target object from the language instruction and visual scene. The selected region is combined with depth information to estimate a 3D goal, which is passed to EGO-Planner [13] for B-spline trajectory generation. To overcome the low-level complexity of the hierarchical pipeline, we leverage the pretrained RAPTOR policy [14], which maps a position setpoint and the proprioceptive data directly to motor outputs. To maintain safety across this hierarchy, we validate the projected goal against the operating bounds before planning, while the planner's occupancy map supports collision-aware trajectory generation. We investigate these questions through system design, real-flight evaluation, and per-module characterization. Our contributions are:

- An onboard decoupled aerial VLN stack combining a quantized VLM for open-vocabulary grounding, EGO-Planner for 3D B-spline planning, and pretrained RAPTOR for low-level control, without end-to-end training or offboard compute.
- A lightweight safety supervisor, using a finite-state machine to reject projected goals outside the operating bounds and to gate trajectory execution based on planner feasibility from the onboard occupancy map.
- A real-flight feasibility evaluation of the full onboard stack on an open-vocabulary object-goal task over three referents, and a per-stage failure attribution enabled by the modular design.

2 Related Work

Aerial VLN has largely been end-to-end, a single network mapping instructions and egocentric observations to actions, from the AerialVLN benchmark [15] to recent open-vocabulary models [6, 7], with VLA-AN [16] carrying such a policy fully onboard. This coupling forfeits the observability and path-planning safety guards of a conventional stack and ties the policy to one platform. In contrast, VLN on the Fly keeps language grounding, metric planning, and low-level control as separate stages, enabling onboard VLN while preserving inspectable interfaces.

Decoupled VLNs split semantic reasoning from control to recover those guards. DualVLN [11] separates high and low-level execution, while See-Point-Fly [17] and Fly0 [18] ground an instruction to an image point, the latter projects it to a 3D target with depth for a geometric planner, the same mid-stack we adopt. AirHunt [19] keeps the slow VLM from bottlenecking the planner, and OnFly [20] runs the VLM fully onboard on our class of embedded computers, confirming the recipe is both effective and onboard-capable. We adopt the same decoupling principle, but replace pixel-level grounding with coarse-grid grounding and connect the VLM-to-planner interface to a learned flight policy.

Reinforcement Learning Policies map vehicle state directly to actuator commands at the high control rates required for agile flight, around 100 Hz, as shown in champion-level drone racing [21] or autonomous inspection [22]. However, many learned controllers specialize in a single airframe and require renewed system identification or retraining after hardware changes. RAPTOR [14] addresses this issue with a foundation policy that adapts across quadrotors zero-shot. We use RAPTOR as the low-level stage of the decoupled VLN stack, feeding position setpoints directly from the planner and avoiding a hand-tuned position-attitude cascade.

3 System Design

Our framework presents a modular, hierarchical, and fully onboard VLN architecture with three processing stages coordinated by a safety supervisor (Fig. 2). The grounding stage converts the instruction and current observation into a 3D goal, EGO-Planner generates a feasible trajectory, and RAPTOR tracks the resulting setpoints. Explicit goals and setpoints keep the interfaces observable, while the supervisor validates candidate goals and gates trajectory execution.

3.1 VLM-based Goal Grounding

We use Qwen-3.5-2B quantized to INT4 [12], reducing memory use enough for onboard inference. At each VLM query, the model takes the egocentric RGB frame and the instruction, then returns one of nine named regions from a 3×3 image grid together with a confidence score, or reports that the target is not visible. The grid is defined only in the text prompt and is not rendered onto the image. The returned region is mapped to the corresponding cell during post-processing. Restricting the output to nine region names removes pixel coordinates, making the result easier to check.

For the returned cell, the median valid depth d_{cell} is assigned to its center pixel $(u_{\text{cell}}, v_{\text{cell}})$. The corresponding point in the camera frame is obtained through the back-projection in (1):

$$\mathbf{p}_{\text{camera}} = d_{\text{cell}} \begin{bmatrix} (u_{\text{cell}} - c_x)/f_x \\ (v_{\text{cell}} - c_y)/f_y \\ 1 \end{bmatrix}, \tag{1}$$

where f_x and f_y are the camera focal lengths, and (c_x, c_y) is the principal point. The current pose then transforms the point into the map frame to obtain the planner goal. Using the cell median reduces sensitivity to isolated invalid or noisy depth readings. If the cell contains no valid depth within the sensor range, the goal is rejected.

The goal altitude is constrained to a narrow band around the current flight altitude, keeping the goal within a safe vertical range and avoiding climbs or descents caused by depth noise. Between VLM queries, the planner and policy continue running at their own rates.

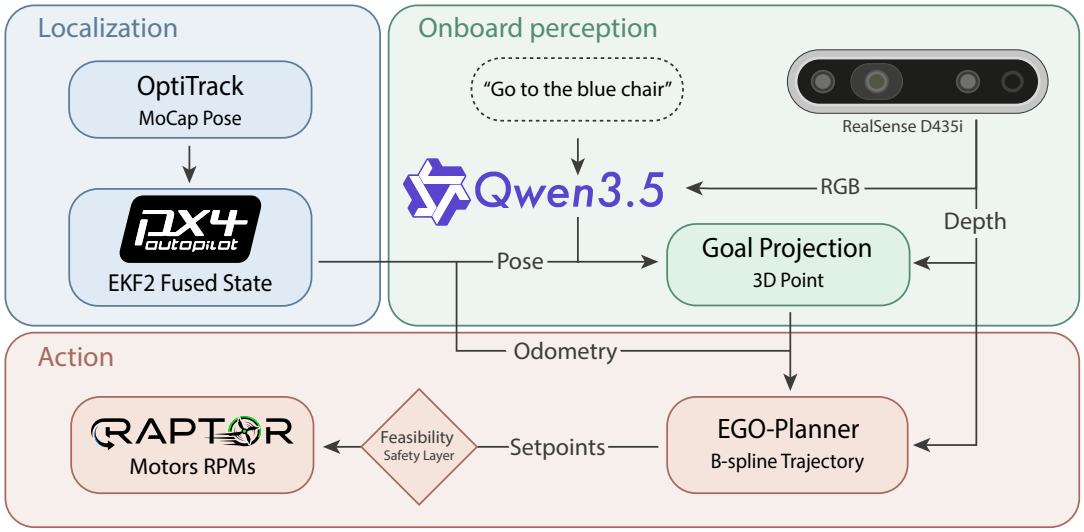

Figure 2: **Overview of the VLN on the Fly.** The RGB-D camera and language instruction are processed onboard by Qwen3.5 to project a 3D goal, EGO-Planner generates a feasible B-spline trajectory, and RAPTOR tracks the resulting setpoints as motor commands. OptiTrack pose is fused by EKF2 for localization in the experimental setup, while the safety layer gates motion.

3.2 *Safety Supervisor*

The safety supervisor controls the handoff between VLM grounding and EGO-Planner. At take-off, the initial vehicle position defines the center of a bounded operating volume. A candidate goal (x_g, y_g, z_g) is accepted only when

$$|x_g - x_0| \le b_x, \qquad |y_g - y_0| \le b_y, \qquad |z_g - z_0| \le b_z, \quad (2)$$

where (x_0, y_0, z_0) denotes the initial vehicle position and b_x, b_y, and b_z define the configured limits along each axis. Goals violating (2) are rejected before planning.

Before goal publication, the supervisor requires three consecutive predictions for the same grid region, each above the configured confidence threshold. Only predictions with a valid projected goal enter the consensus process. After acceptance, one goal remains active until EGO-Planner reports completion, rejection, or failure. Nearby proposals and rapid updates are suppressed to prevent unstable commands between VLM queries.

3.3 *3D Planning*

The validated 3D goal is passed to EGO-Planner [13], which returns a smooth, dynamically feasible trajectory. The planner does not require a Euclidean Signed Distance Field (ESDF), allowing local, on-demand collision evaluation without constructing a full distance field. The lower computational cost allows EGO-Planner to run alongside the VLM on the same onboard computer. EGO-Planner represents trajectories as uniform B-splines and plans natively in 3D.

The B-spline convex-hull property allows velocity, acceleration, and collision constraints to be applied through the control points. The corresponding limits are configured for each airframe, defining how the platform's motion limits are passed to the planner. Local support also confines each control-point update to a short segment of the trajectory.

After goal validation, EGO-Planner checks the trajectory's feasibility against a local occupancy map built from the depth camera and the current pose. The map is entered into the collision cost before the optimized B-spline is sampled into position setpoints. Replanning refreshes the setpoints.

3.4 *Low-level Policy*

A conventional flight-control stack maps planner setpoints to motor commands through position, velocity, attitude, and rate controllers, each tuned to the drone's mass, inertia, motor constants, and thrust limits. RAPTOR [14] replaces the cascade with a pretrained policy that maps the setpoint-relative state directly to actuator commands.

RAPTOR receives the position setpoints from Section 3.3 relative to the current vehicle state. A lightweight tracking bridge samples the planned path as a moving reference and provides position setpoints with velocity feedforward. After replanning, the reference resumes from the path point nearest the current vehicle position, reducing discontinuities between successive trajectories. Updating the stream after replanning ensures the policy tracks the latest trajectory. RAPTOR handles low-level control through a network with 2,084 parameters and three recurrent layers, running directly on the flight controller without retraining for our airframe.

4 Experimental Results

In this section, we evaluate VLN on the Fly as an onboard aerial VLN stack for open-vocabulary object-goal navigation. We first describe the platform and experimental setup, then evaluate goal-reaching performance across 15 real flights and

Figure 3: **Experimental Setup.** **(a)** Controlled indoor flight volume for the open-vocabulary object-goal trials, with the quadrotor, marked operating bounds, and everyday referents. **(b)** Cluttered setup for the path-planning trials (Sec. 4.3), with a table, chair, tool case, and a freestanding vertical panel placed between the take-off pose and the trash-bin referent.

profile onboard VLM latency. We next analyze the perception design through grid-resolution, depth-readout, and referent-discrimination studies. Finally, we test the complete stack in cluttered scenes to assess goal validation, obstacle-aware planning, and trajectory tracking.

We deploy VLN on the Fly on MIRA [23], an open-source quadrotor equipped with an Intel RealSense D435i RGB-D camera, an onboard Jetson Orin NX, and a Pixhawk 6C flight controller. An OptiTrack PrimeX 41 motion-capture system provides pose estimates used as odometry.

4.1 *Performance Analysis*

Trials are conducted as shown in Fig. 3, with three everyday referents: a trash bin, a chair, and a fire extinguisher. For each referent, we run the full stack 5 times, for a total of 15 onboard flights from the same take-off pose. Each flight receives a single instruction naming the target. The scene layout remains fixed during each run.

As reported in Table 1, our method reached the commanded referent in 13 of 15 flights (87% success rate, with a Wilson 95% confidence interval of 0.62–0.96). The trash bin succeeded in all five runs, ending directly above the referent every time (zero goal error) even though the executed paths curved on the way in. The chair and fire extinguisher each failed once, and in both cases the vehicle approached the correct object but stopped short of the 20 cm radius once the target left the camera view, preventing goal refinement.

Table 1: Open-Vocabulary Object-Goal Navigation

Referent	**Success rate ↑**	**Goal err. (cm) ↓**	**Track err. (cm) ↓**	**Avg. GPU (%)**
Trash bin	1.00	0.00	34.96	42.2
Chair	0.80	10.07	26.13	37.3
Fire extinguisher	0.80	7.08	37.64	38.4
Overall	**0.87**	**5.72**	**32.91**	**39.3**

The qualitative run in Fig. 4 exposes the main interfaces of the stack during a successful run. The VLM output is correct at the coarse-region level, supporting our choice to avoid pixel-level grounding. However, the main residual error appears downstream, as the limited RGB-D range and local inflation in the occupancy map can bias the planner away from a straight approach, as seen in the curved projected trajectory.

Beyond the GPU utilization of Table 1, we profile the grounding stage directly from the onboard inference records (Table 2, N=1865 queries). On the Jetson Orin NX the quantized 2B VLM completes a grounding query in a median of 0.79 s (95th percentile 0.88 s), which comfortably sustains the 0.5 Hz query rate and leaves the planner and policy to run uninterrupted in between. Time-to-first-token, which covers vision encoding and prompt prefill, dominates and is nearly constant at 0.60 s, the total time varies only with the short region output, so latency is stable across queries.

Table 2: Onboard VLM Inference Timing

Stage	**Median**	**95th pct.**	**Max**
Time-to-first-token	0.60 s	0.66 s	0.72 s
Total inference	0.79 s	0.88 s	1.12 s

4.2 *Perception Ablation Study*

To isolate the effect of otherwise untracked hyperparameters and to motivate our design choice, Table 3 presents an offline analysis of the perception layer. We replayed the logged onboard depth frames using the same cell-selection and projection pipeline as in flight. In addition to the valid-goal rate against the ground-truth position, we report the inter-frame goal-selection jitter.

A 1×1 grid provides stable depth estimates but no image-plane localization and therefore cannot spatially distinguish multiple referents. Among grids that preserve spatial localization, 3×3 achieves a valid-goal rate of 1.00 with substantially lower jitter than 5×5, motivating its adoption as our default configuration.

Taking the grid to its limit, a single pixel, removes the region abstraction and makes explicit why we ground at the cell level (bottom of Table 3). Back-projecting the goal from

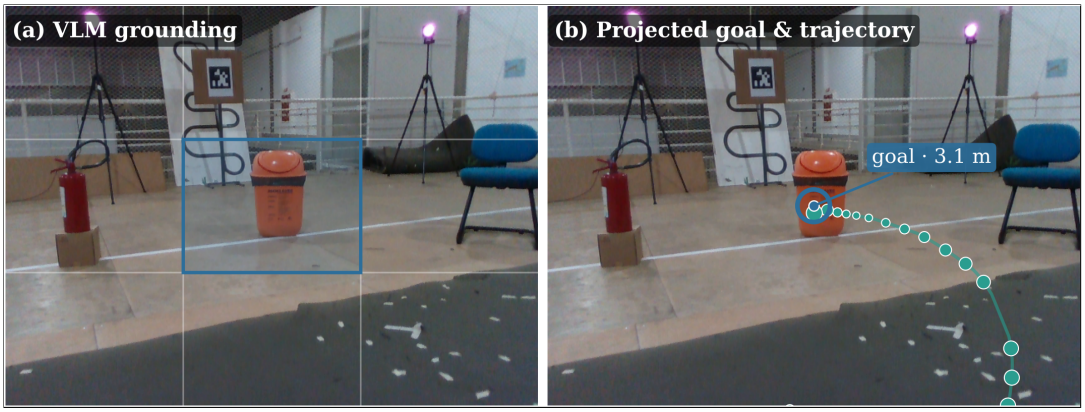

Figure 4: **Qualitative Grounding and Planning Result.** A single onboard frame from a representative successful run for the instruction *"go to the trash bin"*, with the fire extinguisher and chair present as distractors. **(a)** VLM grounding over the image grid: the selected center cell (blue) localizes the referent. **(b)** The same frame as a navigation command: the back-projected 3D goal at the selected cell's ray, and the executed onboard trajectory projected into the camera view.

Table 3: Grounding Resolution Ablation

Region grid	Valid-goal rate ↑	Selection jitter (m) ↓
1×1	0.94	0.07
3×3 (ours)	1.00	0.70
5×5	1.00	1.27
Single pixel	0.15	1.16

Table 4: VLM Region Selection by Prompted Referent

Prompted referent	Not visible	Modal region	Share
Chair (left)	53%	Top-left	86%
Trash bin (center)	44%	Center	90%
Fire extinguisher (absent)	66%	Center	79%

one image pixel yields a valid depth reading in only ∼15% of frames (invalid in 85%), since a lone pixel frequently lands on a low-texture, reflective, or out-of-range surface, and where it is valid, it can read across an object boundary and back-project from the background behind the referent, differing from the cell median by up to 0.8 m. The selected 3×3 cell median avoids these failures.

To further support the open-vocabulary claim, we replayed identical onboard frames through the VLM, varying only the referent name in the prompt (N=1865 queries, Table 4). On frames where the model commits to a region, the two present referents map to disjoint, scene-consistent locations, the chair to the top-left (86%) and the trash bin to the center (90%). The absent referent is correctly reported as not visible in 66% of frames, but when it does commit, it defaults to the center (79%), the trash bin's region. This hallucination mode does not affect the discrimination between present referents, but it suggests trusting not-visible reports over commitments to out-of-scene targets. Although the VLM exhibits a high false-negative rate for present objects (44–53% not visible), the tracking and projection pipeline successfully bridges these intermittent perception gaps.

4.3 Cluttered-Environment Evaluation

The trials above isolate grounding accuracy in a largely open volume. To probe the stack under the cluttered conditions typical of indoor deployment, we ran a small set of additional onboard flights ($n = 6$) toward the trash-bin referent with obstacles (a chair, a tool case, and mocap tripods) placed between the take-off pose and the target. Using the onboard occupancy map, EGO-Planner produced collision-free B-spline trajectories that RAPTOR tracked to the referent. In three of the six runs the stack reached the referent collision-free, ending 0.00–0.183 m from the goal. In the other two the safety layer never released a goal that satisfied its depth and bounds checks, so no trajectory was executed and no collision occurred. In the remaining run (Run 3), the vehicle successfully avoided obstacles but terminated just outside the success radius (0.21 m from the goal) due to high tracking deviation. This demonstrates that the same onboard stack extends from open-volume goal reaching to cluttered navigation without changes, and that the goal-validation interface degrades to a safe no-commit. Fig. 5 shows the planned and flown trajectories for a representative run, and Table 5 reports per-run errors measured from trajectory generation to the first goal-reach. Across the four executed runs the flown path stayed close to the planned B-spline, with a mean tracking error of 17.6 cm (per-run means 10.9–29.9 cm), altitude was held near the commanded band, so the horizontal path carried the navigation while the vehicle navigated the clutter.

5 Limitations

As with many open-vocabulary systems, the generalization of our results is not fully characterized. While experi-

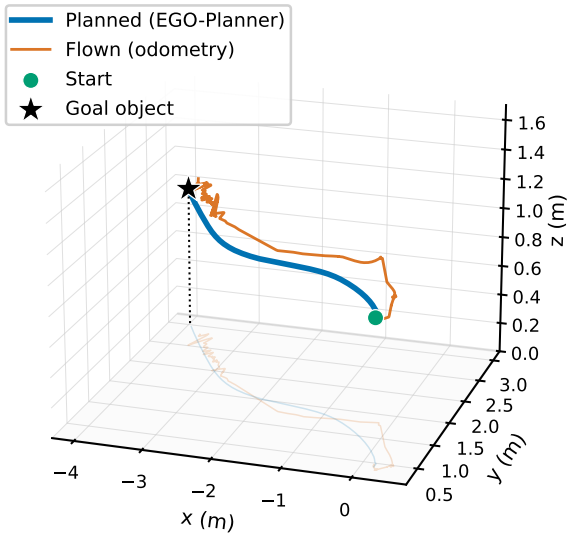

Figure 5: **Planned vs. Flown Trajectory in Clutter.** EGO-Planner's collision-free B-spline (blue) runs from take-off (green) to the goal object (star), and RAPTOR tracks it (orange). Altitude is held near the commanded band, faint floor shadows show the horizontal path. The window spans trajectory generation to the first goal-reach.

ments demonstrate the feasibility of the proposed VLN tasks, gains in modularity and safety come at the cost of integration trade-offs. In particular, the target remains fixed between goal queries, so a grounding error can persist until the next query, an asynchronous design could refresh it sooner. Grounding also operates on the current egocentric frame alone, with no spatial memory, so a referent outside the field of view must first be brought into view before it can be grounded, which bounds the contextual and sequential task families. Finally, depth back-projection proved to be limited in practice by the stereo sensor's range and by invalid depth regions.

To isolate the present contributions, we made two simplifying assumptions that future work could relax: fixed-altitude flight and external odometry. The z-clamp fixes the altitude per environment, so the planner's full 3D capability is not exercised. Moreover, all trials are conducted within a single OptiTrack volume from a known initial pose, without evaluating arbitrary initialization or relocalization.

6 Conclusion

We presented **VLN on the Fly**, an onboard decoupled aerial VLN stack that grounds natural-language instructions into 3D goals with a quantized VLM, plans collision-aware trajectories with EGO-Planner [13], and executes position setpoints through the pretrained RAPTOR policy [14]. The decoupled design preserves intermediate observability, allowing grounding, goal projection, planning, and control errors

Table 5: Goal and Tracking Errors for Cluttered Flights

Run	Goal err. (cm) ↓	Mean track. err. (cm) ↓	p90 track. err. (cm) ↓
1	0.00	14.5	18.6
2	18.3	15.1	26.4
3	20.7	29.9	79.1
4	0.00	10.9	21.2
Mean	**9.75**	**17.6**	**36.3**

to be inspected separately during real flight. Our experiments characterize open-vocabulary object-goal navigation in a controlled indoor volume, demonstrating onboard execution while highlighting remaining sensitivity to depth range, metric projection, and planner-interface effects. Complementary sensing, longer-range depth, and tighter coupling between perception uncertainty and trajectory generation could improve robustness in less constrained environments and provide a clearer path toward longer-horizon aerial VLN.

7 Acknowledgments

This work was supported in part by the São Paulo Research Foundation (FAPESP), Grants #2025/20858-7 and #2025/22381-3; in part by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), Grant 308092/2020-1; and by Petróleo Brasileiro S/A - Petrobras, using resources from the ANP R&D clause, in partnership with the University of São Paulo (USP) and the Fundação de Apoio à Física e à Química (FAFQ), under Cooperation Agreements #2023/00016-6 and #2023/00013-7.

References

[1] Hongyi Zhou, Weiran Liao, Xi Huang, Yucheng Tang, Fabian Otto, Xiaogang Jia, Xinkai Jiang, Simon Hilber, Ge Li, Qian Wang, Ömer Yağmurlu, Nils Blank, Moritz Reuss, and Rudolf Lioutikov. Beast: Efficient tokenization of b-splines encoded action sequences for imitation learning. In *Advances in Neural Information Processing Systems*, volume 38, pages 172934–172959, 2025.

[2] Moritz Reuss, Hongyi Zhou, Marcel Rühle, Ömer Erdinç Yağmurlu, Fabian Otto, and Rudolf Lioutikov. Flower: Democratizing generalist robot policies with efficient vision-language-action flow policies, 2025.

[3] Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. GNM: A general navigation model to drive any robot. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7226–7233, 2023.

[4] Haoran Liu, Weikang Wan, Xiqian Yu, Minghan Li, Jiazhao Zhang, Bo Zhao, Zhibo Chen, Zhongyuan Wang,

Zhizheng Zhang, and He Wang. Navid-4d: Unleashing spatial intelligence in egocentric RGB-D videos for vision-and-language navigation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10607–10615, 2025.

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

[6] Xiaolou Sun, Wufei Si, Wenhui Ni, Yuntian Li, Dongming Wu, Fei Xie, Runwei Guan, He-Yang Xu, Henghui Ding, Yuan Wu, Yutao Yue, Yongming Huang, and Hui Xiong. Autofly: Vision-language-action model for UAV autonomous navigation in the wild, 2026.

[7] Maximilian Adang, JunEn Low, Ola Shorinwa, and Mac Schwager. SINGER: An onboard generalist vision-language navigation policy for drones, 2025.

[8] Yue Zhang, Ziqiao Ma, Jialu Li, Yanyuan Qiao, Zun Wang, Joyce Chai, Qi Wu, Mohit Bansal, and Parisa Kordjamshidi. Vision-and-language navigation today and tomorrow: A survey in the era of foundation models. *Transactions on Machine Learning Research*, 2024.

[9] Joonkyung Kim, Wenxi Chen, Davood Soleymanzadeh, Yi Ding, Xiangbo Gao, Zhengzhong Tu, Ruqi Zhang, Fan Fei, Sushant Veer, Yiwei Lyu, Minghui Zheng, and Yan Gu. Modular safety guardrails are necessary for foundation-model-enabled robots in the real world, 2026.

[10] Abdelrhman Werby, Chenguang Huang, Martin Büchner, Abhinav Valada, and Wolfram Burgard. Hierarchical open-vocabulary 3D scene graphs for language-grounded robot navigation. In *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024*, 2024.

[11] Meng Wei, Chenyang Wan, Jiaqi Peng, Xiqian Yu, Yuqiang Yang, Delin Feng, Wenzhe Cai, Chenming Zhu, Tai Wang, Jiangmiao Pang, and Xihui Liu. Ground slow, move fast: A dual-system foundation model for generalizable vision-and-language navigation, 2025.

[12] Qwen Team. Qwen3.5: Towards native multimodal agents, 2026.

[13] Xin Zhou, Zhepei Wang, Hongkai Ye, Chao Xu, and Fei Gao. EGO-Planner: An ESDF-free gradient-based local planner for quadrotors. *IEEE Robotics and Automation Letters*, 6(2):478–485, 2021.

[14] Jonas Eschmann, Dario Albani, and Giuseppe Loianno. RAPTOR: A foundation policy for quadrotor control. *Science Robotics*, 11(114):eaec1481, 2026.

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

[16] Yuze Wu, Mo Zhu, Xingxing Li, Yuheng Du, Yuxin Fan, Wenjun Li, Zhichao Han, Xin Zhou, and Fei Gao. VLA-AN: An efficient and onboard vision-language-action framework for aerial navigation in complex environments, 2025.

[17] Chih Yao Hu, Yang-Sen Lin, Yuna Lee, Chih-Hai Su, Jie-Ying Lee, Shr-Ruei Tsai, Chin-Yang Lin, Kuan-Wen Chen, Tsung-Wei Ke, and Yu-Lun Liu. See, point, fly: A learning-free VLM framework for universal unmanned aerial navigation. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pages 4697–4708. PMLR, 2025.

[18] Zhenxing Xu, Brikit Lu, Weidong Bao, Zhengqiu Zhu, Junsong Zhang, Hui Yan, Wenhao Lu, and Ji Wang. Fly0: Decoupling semantic grounding from geometric planning for zero-shot aerial navigation, 2026.

[19] Xuecheng Chen, Zongzhuo Liu, Jianfa Ma, Bang Du, Tiantian Zhang, Xueqian Wang, and Boyu Zhou. Airhunt: Bridging VLM semantics and continuous planning for efficient aerial object navigation, 2026.

[20] Guiyong Zheng, Yueting Ban, Mingjie Zhang, Juepeng Zheng, and Boyu Zhou. Onfly: Onboard zero-shot aerial vision-language navigation toward safety and efficiency, 2026.

[21] Elia Kaufmann, Leonard Bauersfeld, Antonio Loquercio, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Champion-level drone racing using deep reinforcement learning. *Nature*, 620(7976):982–987, 2023.

[22] Marco S. Tayar, Lucas K. de Oliveira, Felipe Andrade G. Tommaselli, Juliano D. Negri, Thiago H. Segreto, Ricardo V. Godoy, and Marcelo Becker. Autonomous UAV flight navigation in confined spaces: A reinforcement learning approach. In *2025 Latin American Robotics Symposium (LARS)*, pages 1–6, 2025.

[23] Lucas K. de Oliveira, Felipe A. G. Tommaselli, João Aires Marsicano, Marco S. Tayar, Pedro A. R. Saraiva, Ricardo V. Godoy, and Marcelo Becker. Mira: A modular open-source micro-uav for indoor research, 2026.